AUTOMATED TEST CASE GENERATOR FOR PHISHING PREVENTION USING GENERATIVE GRAMMARS AND DISCRIMINATIVE METHODS

by

Sean Palka
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Information Technology

Committee:

____________________________	Dr. Damon McCoy, Dissertation Director
____________________________	Dr. James Jones, Committee Member
____________________________	Dr. Dana Richards, Committee Member
____________________________	Dr. Lois Tetrick, Committee Member
____________________________	Robert Osgood, Committee Member
____________________________	Dr. Stephen Nash, Senior Associate Dean
____________________________	Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering
Date: ______________________	Fall Semester 2015 George Mason University Fairfax, VA

Automated Test Case Generator for Phishing Prevention Using Generative Grammars and Discriminative Methods

A Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

by

Sean Palka
Master of Science
Marymount University, 2001
Bachelor of Arts
University of Notre Dame, 1998

Director: Damon McCoy, Professor
International Computer Science Institute

Fall Semester 2015
George Mason University
Fairfax, VA

DEDICATION

This research is dedicated to my parents Mary Ann and Dr. Bruce Palka, and my brilliant daughter Malia. It has taken many years, and I am forever grateful for your support while pursuing this degree.

I promise I will only do this once.

ACKNOWLEDGEMENTS

This thesis is the final deliverable from approximately ten years of study at George Mason University. It has been a long, challenging road, balancing both professional and academic schedules and requirements, but I would like to take a moment to acknowledge and thank some of the key people involved with this research. Without them, I would not be where I am today, nor would I be who I am today.

First of all, I would like to thank Dr. Harry Wechsler for his critical involvement with developing the core concept of this thesis, and for providing important feedback and guidance through the process. Dr. Wechsler was one of my professors, a co-author for papers, and was also my academic advisor for a period of time. He was instrumental in my continued presence at George Mason University, and helped vet the ideas presented in this dissertation document.

I would also like to thank Booz Allen Hamilton and my exceptional managers for affording me the flexibility to focus on my research when necessary. Dr. Ron Ritchey, John Pearce, and Jeff Lunglhofer each provided the professional guidance needed to succeed, while making sure I didn’t burn out in the process. I would also like to thank the Booz Allen Hamilton security staff for their assistance and professional courtesy during my research.

Last, but definitely not least, I would like to thank my committee of Dr. Damon McCoy, Dr. Jim Jones, Robert Osgood, Dr. Dana Richards and Dr. Lois Tetrick. As my chair, Dr. McCoy helped me push through to the end, and I could not have done it without him. Likewise, the rest of my committee provided excellent feedback and sanity checks to keep my research on track, and I am forever grateful for their guidance and integrity. I would also like to thank Dr. Duminda Wijisekera and Dr. Mihai Boicu for their critical feedback into this research topic and their guidance regarding aspects of experimental design.

TABLE OF CONTENTS

Page

List of Tables viii

List of Figures ix

List of Equations xi

List of Abbreviations xii

Abstract xiii

Chapter 1: Introduction 1

1.1 Need for Dynamic and Relevant Content 2

1.2 Problem Statement 4

1.3 Fundamental Contributions 5

1.3 Dissertation Outline 7

Chapter 2: Background and Related Work 9

2.1 Phishing Overview 10

2.2 Technical Mitigation Research 12

2.3 People-centric Mitigation Research 15

2.4 Live Phishing Exercises 19

Chapter 3: Ethical Considerations 21

3.1 Human Subjects Testing 21

3.2 Potential for Misuse of Research 23

Chapter 4: Phishing Exercise Design and Best Practices 25

4.1 Determine Level of Notification 25

4.2 Define Exercise Objectives 27

4.3 Define Target List Selection Process 29

4.4 Determine Exercise Schedule and Frequency 32

4.5 Determine Infrastructure Requirements 34

4.6 Determine E-mail Structure and Contents 36

4.7 Develop Rules of Engagement Document 37

Chapter 5: Motivational Study on Content Complexity 39
5.1 Hypothesis 39
5.2 Experimental Design 40
5.3 Results and Analysis 44
5.4 Conclusions 48
Chapter 6: Motivational Study on Content Reuse 49
6.1 Hypothesis 49
6.2 Experimental Design 50
6.3 Results and Analysis 51
6.4 Conclusions 54
Chapter 7: Impact of Content Complexity and Training on Responses 56
7.1 Hypothesis 56
7.2 Experimental Design 57
7.3 Results 60
7.4 Analysis 64
7.4.1 E-mail Complexity 65
7.4.2 Normalized Gain 67
7.4.2 Recidivism 72
7.5 Conclusions 81
Chapter 8: Generating Consistent Semantic Content with Grammars 83
8.1 Generative Grammars 83
8.2 Context-free Grammars in PhishGen 90
8.3 Content Generation Process Overview 94
8.4 Information Flow and Feedback 100
8.5 User Interface Design 104
8.6 Content Generation Strategies 109
Chapter 9: Content Characteristic Simulations 111
9.1 Simulation Environment and System Settings 112
9.2 Learning, Creation, and Diversity Metrics 113
9.3 Hypothesis 116
9.4 Results 117
9.5 Analysis and Conclusion 125

Chapter 10: Evaluation of Effectiveness in Live Exercises 126
10.1 Hypothesis 126
10.2 Experimental Design 130
10.3 Results 131
10.4 Analysis 133
10.5 Conclusions 136
Chapter 11: Improvement of Existing Models 137
11.1 Filter Evasion and Hardening 137
11.2 Hypothesis 140
11.3 Experimental Design 140
11.4 Results 143
11.5 Analysis and Conclusions 144
Chapter 12: Conclusions and Future Work 147
12.1 Fundamental Contributions Overview 147
12.1.1 Live Phishing Exercise Capabilities and Best Practices 148
12.1.2 Content Impact Analysis Results 148
12.1.3 Semantic Consistency Engine 149
12.1.4 Diverse Dataset Generation 149
12.1.5 Model Improvement Using Iterative Feedback Testing 150
12.2 Future Work 150
12.2.1 Codex Generation 150
12.2.2 Additional Phishing Exercises 151
12.2.3 Improved Features and Grammar Rule Formats 152
12.3 Final Remarks 152
References 154

LIST OF TABLES

Table Page
Table 1 Overview of results from the Content Impact Study 45
Table 2 Significant differences in click-through rates using a two-tailed Z test of 2 population proportions 45
Table 3 Significant differences in reporting rates using a two-tailed Z test of 2 population proportions 46
Table 4 Deployment strategy and Click-through rates for Content Reuse Study 53
Table 5 Cohort sizes for the Complexity and Training Study 59
Table 6 Percentage of participants from 32 test populations in the Complexity and Training Study who clicked on links in phishing e-mails across 6 rounds of testing 61
Table 7 Percentage of participants from 32 test populations in the Complexity and Training Study who received training after responding to a phishing e-mail 62
Table 8 Combined click-through values for each test e-mail across all test populations for 6 rounds of exercises 63
Table 9 Efficiency rating for 6 rounds of test e-mails in the Complexity and Training Study using normalized gain analysis. 70
Table 10 Example list of production rules for generating content with PhishGen 92
Table 11 Creation, learning, and diversity results for all content characteristic simulations 119
Table 12 Click-through statistics for Comparison Study 132
Table 13 Selected cohorts for comparison based on e-mail received in specific rounds of testing 132
Table 14 Click-through rates and significant difference for selected cohorts 134
Table 15 Summary of results for SME generated content in selected cohorts 135
Table 16 Summary of results for PhishGen generated content in selected cohorts 135
Table 17 Detection rates for multiple simulations across two test environments 144

LIST OF FIGURES

Figure Page
Figure 1 Focal points for existing research in the communication path of a phishing e-mail 12
Figure 2 An example of phishing e-mail content currently used in a user awareness training module 17
Figure 3 Example of e-mail sent to the “Random” cohort as part of the Content Complexity Study 42
Figure 4 Example of e-mail sent to the “Scam” cohort as part of the Content Complexity Study 42
Figure 5 Example of e-mail sent to the “Link” cohort as part of the Content Complexity Study 43
Figure 6 Example of e-mail sent to the “Secure” cohort as part of the Content Complexity Study 43
Figure 7 Example of e-mail sent to the “Password” cohort as part of the Content Complexity Study 44
Figure 8 Actual phishing e-mail detected in the wild utilizing the link-only strategy used for the “Link” cohort........ 48
Figure 9 Click-through rates for three rounds of the Content Reuse Study showing potentially decreased utility 54
Figure 10 Click-through value ranges for the 28 e-mails used in the Complexity and Training Study 64
Figure 11 Normalize gain over the course of 6 exercises compared to raw click-through rate, indicating a decrease in the effectiveness of phishing attacks 71
Figure 12 Instances of repeated responses in the Complexity and Training Study where e-mail difficulty increased, stayed the same, or decreased 74
Figure 13 Number of trained participants in the Complexity and Training Study after first-time responses across 6 rounds 74
Figure 14 First-time responses and repeated responses for trained and untrained participants in the Complexity and Training Study across 6 rounds 75
Figure 15 Total number of responses per participant in the Complexity and Training Study 77
Figure 16 Interim rounds of testing between responses in the Complexity and Training Study 77
Figure 17 Sample grammar trees showing an example implementation of variable interpolation 85
Figure 18 Example of a simulated phishing e-mail with obvious context errors 88

Figure 19 Example of a simulated phishing e-mail with context-appropriate visuals...... 89
Figure 20 Example of a context-free grammar (CFG) in Chomsky normal form............ 91
Figure 21 An example of one possible expansion of the start state using production rules in our sample grammar 96
Figure 22 Possible changes in the environment and potential areas for feedback during a phishing attack 100
Figure 23 Process overview for generating content in PhishGen 103
Figure 24 Illustration of the web-based user interface for PhishGen 107
Figure 25 E-mail generation procedures in PhishGen 108
Figure 26 Example of two different e-mails generated from the same available production rules in PhishGen 115
Figure 27 Learning results graphs for different threshold values and content generation approaches.. 122
Figure 28 Diversity results graphs for different threshold values and content generation approaches.. 123
Figure 29 Creation results graphs for different threshold values and content generation approaches.. 124
Figure 30 Secure e-mail generated by PhishGen... 127
Figure 31 Order confirmation e-mail generated by PhishGen....................................... 128
Figure 32 Electronic fax e-mail generated by PhishGen 128
Figure 33 Relief fund e-mail generated by PhishGen... 129
Figure 34 Free cruise e-mail generated by PhishGen 129
Figure 35 IQ Test e-mail generated by PhishGen... 130
Figure 36 Normalized Gain for SME and PhishGen e-mails 136
Figure 37 Process overview for using n-gram analysis to update filters and PCFG rule weights 139
Figure 38 PhishGen tracks specific good and bad n-grams which can be used to update filters 145

LIST OF EQUATIONS

Equation Page
Equation 1 Formula for measuring normalized gain 69
Equation 2 Formula for measuring diversity in a set of e-mails..................................... 116

LIST OF ABBREVIATIONS

Uniform Resource Locator	URL
Intrusion Detection System	IDS
Intrusion Prevention System	IPS
Context-free Grammar	CFG
Probabilistic Context-free Grammar	PCFG
Subject Matter Expert	SME
Learning Management System	LMS
Short Message Service	SMS

ABSTRACT

AUTOMATED TEST CASE GENERATOR FOR PHISHING PREVENTION USING GENERATIVE GRAMMARS AND DISCRIMINATIVE METHODS

Sean Palka, Ph.D.

George Mason University, 2015

Dissertation Director: Dr. Damon McCoy

This research details a methodology designed for creating content in support of various phishing prevention tasks including live exercises and detection algorithm research. Our system uses probabilistic context-free grammars (PCFG) and variable interpolation as part of a multi-pass method to create diverse and consistent phishing email content on a scale not achieved in previous research. This system, which we have named PhishGen, is capable of generating a large amount of unique content that can be used in live exercises, or alternatively used to build training datasets for phishing detection methods and filter settings. PhishGen is a web-based application that implements our underlying methodology to provide a user-interface for building and modifying PCFG rules and weights. The system is released as an open-source tool in order to allow access to other researchers. PhishGen has already been used in support of

live commercial phishing exercises and is in the process of being utilized for content development for commercial frameworks.

As part of our research, we present the results of multiple studies supporting our hypothesis regarding the impact of content on phishing exercises. We present a study focusing specifically on how phishing content affects click-through rates, and demonstrate how compelling content generates significantly higher click-through rates when compared to poorly crafted phishing content. We then present the results of a study that investigates whether content maintains its utility when being replayed across a population. The results of these initial motivational studies provided empirical evidence that content generation is a topic worth investigating. Next we present the results of a more thorough study involving the entire population of medium-sized commercial organization, in which we demonstrate again the impact of content-complexity and provide a normalization approach that takes into account differences in phishing e-mails. We then present several studies to test the effectiveness of PhishGen, during which several live phishing exercises were run to demonstrate how our generated content performs compared to phishing e-mails manually crafted by experts. We also present the results of simulations that did not involve live exercises to measure various characteristics of content created by PhishGen. Finally, we demonstrate how PhishGen is able to adapt to previous responses, or lack of responses, to generate more effective e-mails in subsequent exercises, while maintaining a higher level of diversity than existing methods of content generation. We show how this approach can be used to strengthen existing filters by identifying gaps in coverage. In all, over 115,000 test phishing e-mails were sent to over 19,000 participants in the course of our studies, making this one of the largest phishing research initiatives to date.

CHAPTER 1:

INTRODUCTION

On March 17, 2011 RSA Security was the victim of a sophisticated cyber attack that resulted in the compromise of millions of employee records. The compromise undermined the security associated with the company's SecurID tokens, and ultimately resulted in multiple additional attacks against Fortune 500 companies [1]. The breach started with an e-mail that was caught by spam filters and ended up in users' Junk mailboxes, but even then at least one user opened the malicious attachment, providing the attackers with access to internal network resources.

RSA is not alone in the long list of successful phishing campaigns that have become public. In August of 2013, the Syrian Electronic Army, a hacktivist group, was able to compromise the New York Times and Twitter by using a phishing campaign as the initial attack [2]. In November of 2013, Target Corporation was the victim of a breach in which over 70 million customers were impacted, as a result of one of their vendor contacts being targeted with a phishing attack [3][4]. As early as December of 2014, according to most estimates, the Office of Personnel Management was the victim of a breach that resulted in the compromise of sensitive information of over 22.1 million individuals, and was likely the result of initial social engineering attacks that were used to harvest valid user credentials [5][6]. In July of 2015, the Office of the Joint Chiefs was the victim of a sophisticated e-mail phishing campaign that was attributed to state-sponsored adversaries [7][8].

These attacks serve as excellent examples that even highly trained security staff are susceptible to social engineering attacks, with phishing being one of the most effective variants. The attacks succeeded despite the myriad of phishing prevention and detection technologies that were deployed at RSA and the New York Times, or even at the Pentagon, and these are still only a few examples of the many high-profile breaches that continue to occur with phishing as the initial means of compromise. There will undoubtedly be more in the future as well.

According to research at McAfee Labs, phishing remains a very effective attack vector, despite existing technologies intent on preventing it. While technical countermeasures can prevent obvious phishing e-mails, according to McAfee Labs Threats Report for August 2014, the primary burden remains on the user [9]. Unfortunately, in a survey of 16,000 business users, 80% fell for at least one type of phishing e-mails in the McAfee Phishing Quiz [10].

1.1 Need for Dynamic and Relevant Content

The continued success of phishing attacks indicates a potential weakness in existing prevention and detection measures. However, the weakness is not necessarily in the algorithms or methodologies being developed, but rather a fundamental flaw in how content is provided and utilized by these systems during the training and evaluation phase. The same flaw exists in how people are trained to detect phishing scams through user awareness training. Detection and prevention algorithms that identify phishing e-mails are trained on data from existing or already detected attacks, and while this is a good way of preventing known threat signatures, it is a reactionary approach that fails to prevent unknown payloads or variations, typically called “zero-day” e-mails. For phishing e-mails that rely on semantic content in order to coerce victims, rather than obfuscation techniques, this approach can’t succeed without flagging every e-mail as suspicious.

Rather than propose a new approach, or a new detection algorithm, we feel that the existing models can be improved by providing more effective content generation methods for training datasets and exercise content. By supplementing the existing datasets with more dynamic content, algorithms should be able to detect more holistic threat signatures. Similarly, with more dynamic content for live exercises, participants will not be able to key in on specific types of e-mails or specific content in e-mails that are reused, but would have to be able to detect the threat based on other characteristics.

In order to support this approach, though, there must be a methodology for generating test cases that emulate realistic attacks, and in such a way that a sufficiently large training dataset can be built. The generated training datasets must be large enough to compare with currently available datasets, or it would provide limited utility. With a large collection of test cases, live exercises could also be executed without repeating content over multiple exercises, or even duplicating the same e-mail within a single exercise cohort.

To address this need, we have developed an approach that uses generative grammars to dynamically create semantically valid phishing e-mails based on relatively small sets of grammar rules.

1.2 Problem Statement

Ultimately, phishing attacks are all about the probability of success. On the attack side, the individuals running these attacks try and find new techniques or payloads that may increase, even slightly, the number of successful responses. Traditionally, this has been accomplished by focusing the target list, researching obfuscation techniques to limit detection, or increasing the relevance of content for the targeted user. On the defensive side it is also an incremental approach, seeking to decrease the number of responses by making adjustments to filters, by training users, and by implementing new detection capabilities. With this combined approach, network defenders decrease the likelihood that an e-mail will make it to a user, and therefore decrease the likelihood that the user will be tricked into responding. If a proposed solution is able to increase the efficiency of any of these detection approaches, and ultimately reduce the throughput of successful phishing e-mails, then it could be considered a success.

In this thesis, we propose a method for supplementing the training datasets and test cases for existing models, so that they can be more effective in detecting and preventing phishing attacks. The same approach can be used in iterative testing to identify gaps in detection. We assert the following thesis:

A content generation methodology using a multi-pass approach and generative grammars can create test cases that determine equivalent click-through rates as human generated phishing e-mails.

.

We focus primarily on the methodology for maintaining semantic consistency, and demonstrate that our generated content is effective in simulating a realistic phishing attack [11]. We also show that the methodology creates diverse datasets, and that the datasets can be large enough for training purposes. We also demonstrate how this approach can be used in an iterative filter evaluation process that can be used to improve detection by showing how variations in content can bypass existing detection methods [12]. Once this underlying capability is demonstrated, applications to other live exercises frameworks and training algorithms is a fairly trivial exercise.

1.3 Fundamental Contributions

This dissertation contributes the following to the fields of computer security and computer linguistics:

- **Lessons learned and best practices for live phishing exercises**. (*Chapter 4*) To better enable other researchers to successfully execute live phishing exercises in a safe and effective way, we present some of the obstacles and solutions that we encountered during our research.

- **Impact of content complexity on click-through rates.** (*Chapter 5*) We provide empirical data from a motivational study executed from April 22, 2013 through April 26, 2013 involving 498 participants which tested the impact of different levels of content complexity on click-through rates.
- **Impact of content reuse on click-through rates.** (*Chapter 6*) We provide empirical data from a motivational study executed from April 30, 2014 through May 16, 2014 involving 333 participants which tested the impact of reusing content on click-through rates.
- **Impact of training on click-through rates.** (*Chapter 7*) We provide empirical data from a large study executed from June 23, 2014 through February 17, 2015 involving 19,180 participants which further demonstrated the impact of content on click-through values, and demonstrate the impact of training by analyzing recidivism and normalized gain.
- **Semantic validity engine methodology.** (*Chapter 8*) We present our system design and methodology for maintaining syntax and semantic validity in e-mail contents created by a generative grammar.
- **E-mail characteristics and metrics.** (*Chapter 9*) We discuss various features of test case datasets that we used to determine the effectiveness of our approach, and introduce a measurement for e-mail diversity. Over 10,000 e-mails were used in various simulations to compare existing creation methods with our approach.
- **Effective content from a generative grammar.** (*Chapter 10*) We conducted a 2,253 participant measurement study from June, 3 2013 through November 6, 2013 that

demonstrates the efficacy of our generated phishing content in producing click-through rates on par with content manually created by subject matter experts (SMEs).

- **Iterative testing methodology based on n-gram analysis and feedback.** (*Chapter 11*) We evaluate our approach against current phishing detection solutions deployed in a commercial environment using fuzzing techniques to illustrate the ability of our system to identify gaps in existing filter coverage.

1.3 Dissertation Outline

The remainder of this documented is organized as follows. Chapter 2 provides an overview of existing research and work related to our approach. Chapter 3 discusses some of the ethical consideration that were taking into account while performing our research. Chapter 4 discusses exercise design concepts and discusses some key lessons learned and best practices developed from setting up multiple phishing exercises. Chapter 5 presents the result of our initial motivational study that investigates the impact of content complexity on click-through rates, and demonstrate that semantic content is an important factor in the effectiveness of phishing e-mails. Chapter 6 presents the results of an additional motivational study that investigates the impact of reusing content across multiple exercises, showing that content may lose utility after multiple uses. Chapter 7 presents the results of a large study involving the entire population of a medium-sized commercial entity that demonstrates the impact of training and content across multiple exercises. Chapter 8 introduces PhishGen, our system for generating content, and elaborates on various design principles including probabilistic context-free grammars and semantic validity. Chapter 9 illustrates the various characteristics of our generated e-mails in comparison to existing approaches. Chapter 10 presents the results of a large study involving several thousand participants over the course of several months, which tested our generated e-mails against e-mails created by subject matter experts. Chapter 11 demonstrates the application of our approach to an iterative filter evaluation process that can be used to test and improve existing detection capabilities using a fuzzing methodology. Chapter 12 lists some of the additional applications and follow-on research that we plan on pursuing and summarizes the conclusions reached by our existing research.

CHAPTER 2:

BACKGROUND AND RELATED WORK

Phishing is a social engineering attack that attempts to trick or coerce victims into giving out sensitive information to an attacker. While not a new attack by any means, it is still a relevant attack in the current threat landscape for most organizations, as demonstrated by its use in multiple recent, high profile compromises. Phishing is often used as the initial attack vector to gain a foothold on a network, either by distributing malware or compromising credentials, and as such organizations need to be able to detect and prevent phishing attacks as part of a defense-in-depth strategy.

Our research is differentiated from existing research in multiple ways. First of all, we are not competing with existing detection or prevention approaches, nor do we feel that they are invalidated by our research. Instead, we are presenting a novel method for generating content that can supplement existing models. We focus primarily on the e-mail content, rather than the content of hosted websites, and while some research has been done involving generating text from context-free grammars, we expand the process to include stricter semantic constraints for both text and active HTML content.

2.1 Phishing Overview

Phishing was formally identified as an attack vector as early as 1987 in a presentation to the International HP Users Group, Interix , although the first actual use of the term is generally considered to be in 1995 when the hacking tool AOHell was released [13][14]. It is likely, however, that e-mail was utilized as an attack vector prior to the term being coined, as electronic mail had existed as far back as 1972 [15]. Even if we allow that phishing has only existed since 1987, it would still be an attack vector that has been around for nearly 3 decades. The term “phishing” has also been included to describe other attacks as well that use other media as the delivery method. For example, using SMS as the delivery method is generally called “SMS phishing”, but this phrase is then condensed to the more common term “smishing”. In the same vein, “voice phishing” or “vishing” utilizes phone calls, and “Twitter phishing” or “twishing” uses Twitter messages. The attack strategy is typically the same, regardless of the communication media: get the victim to provide information or perform some action that benefits the attacker.

The earliest phishing e-mails were typically broadcast out to a large number of target addresses, and often contained irregularities in grammar or spelling. Early phishing attacks were a numbers game: all the attacker needed was one successful response to make the attack worthwhile. After all, it cost the attacker nothing to send an e-mail to an extra hundred e-mail addresses, and even if the success rate of the attack is .001%, with enough target addresses it was highly probable that the attack would succeed. By today’s standards, though, these early phishing attacks are relatively easy to spot.

Over time, phishing has steadily evolved to include a broad range of attacks that vary targeting methods, payloads, and even methods of coercion. Rather than target a large number of e-mail addresses with a generic e-mail message, “spear-phishing” uses a smaller target set of e-mails while providing a more focused message [16]. This type of attack usually involves more research by the attacker, such as including valid account numbers or other personal information, so that the content of the e-mail is very convincing and establishes a high degree of trust with the victim. Spear-phishing results in a much higher probability of success per e-mail, and simultaneously reduces the number of e-mails that might be detected by network defenders. “Whaling” takes this approach even further by targeting an extremely small target list, often times just a single person [16]. This individual is usually a very high-profile target, such as a CEO or a high-ranking government official, and the message is extremely tailored. With whaling, the trade-off for a small target list is the extremely high payout if the attack succeeds. Other variations in phishing have included changing the way links are obfuscated, using graphics as a means to relay text, including malware payloads in attachments, mirroring target websites, or even compromising existing servers to support phishing attacks . The attack is constantly evolving to evade detection methods, and detection methods are constantly evolving to detect new attacks [17].

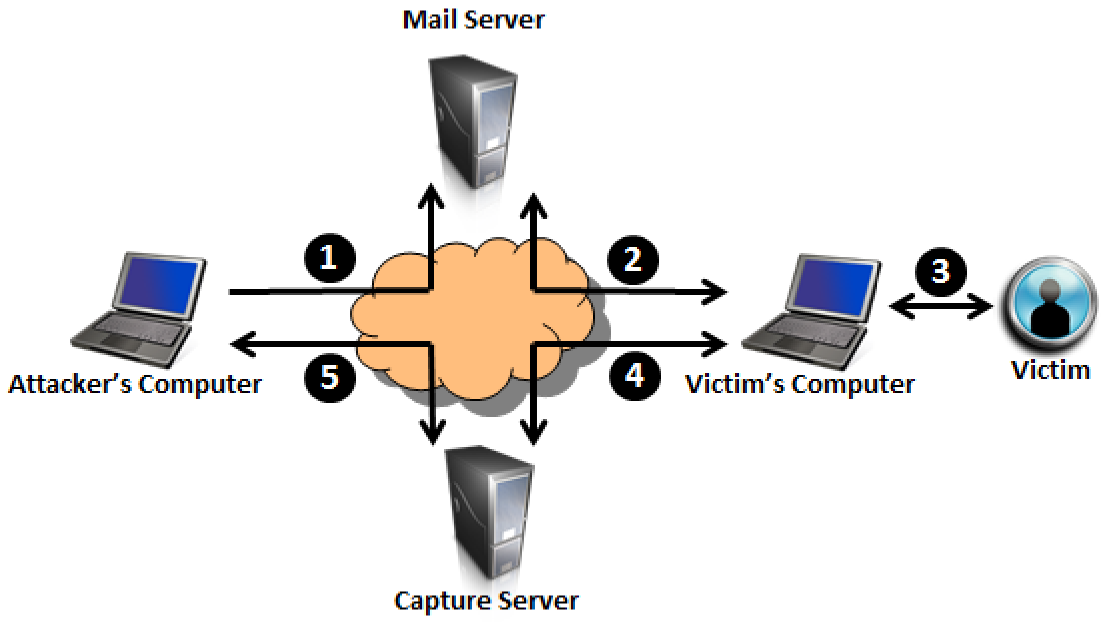


Figure 1 Focal points for existing research in the communication path of a phishing e-mail

2.2 Technical Mitigation Research

The overall attack process in phishing has been researched fairly exhaustively [18][19][20][21][22]. Figure 1 illustrates the typical message path in a phishing attack, and the stages in the attack that are typically focused on by current research. While fairly straightforward, it is important to illustrate the various locations where the attack may be detected, and where the attacker is able to gather information. During a typical phishing attack, the attacker generates an e-mail message and delivers it to a mail server. The victim's computer then uses a mail client to download the message from the mail server, and the victim opens the message. Usually the message will contain a link or other content that, once activated, initiates a request from the victim's computer to a capture server controlled by the attacker. Depending on the type of attack, the victim may be asked to enter sensitive information, or the browser may be compelled to download malicious code. Meanwhile, the attacker polls the capture server periodically to track responses and gather credentials. Research into detection or prevention of phishing typically focuses on one or more areas in this communication path.

Many research approaches to phishing attack detection focus on either detecting the incoming message at the victim’s mail server, or detecting the e-mail inside the victim’s mail client once it is downloaded locally. Research has focused on analyzing the incoming e-mail for suspicious elements, such as obfuscated links or suspicious phrases, or using feature selection algorithms to try and discriminate between phishing e-mails and normal e-mails [23][24][25][26][27][28][29][30]. The benefit to this approach is that, assuming the e-mail can be detected, it can be removed from the user’s workflow and the risk of them clicking on malicious links or attachments is completely negated. However, with active removal of e-mails, there is always the possibility that false positives will result in legitimate e-mails being removed.

Rather than attempt to detect the phishing e-mail on its way to the user, some research has focused on detecting the attack after a user has clicked a link and has attempted to communicate with the server that is being monitored by the attacker. This approach attempts to distinguish legitimate web sites from phishing websites using various novel approaches such as classification mining, white/black listing or reputation, or model-based features [31][32][33][34][35][36][37][34][35]. Similarly, researchers have developed security skins and add-ons that can be used in conjunction with browsers to prevent access to suspicious websites [38][39][40]. Similarly, many of these technical controls have been analyzed for effectiveness [41][42][27].

For the most part, technical controls have limited effect on preventing phishing in live networks for three reasons. First, theoretical models for detecting or preventing phishing have limitations that make them unsuitable for deployment in live networks. Most techniques are either too slow, or too ineffective to be considered a reliable stand-alone mitigation, and none can detect and/or prevent all phishing e-mails from reaching the end user [43]. Second, users that are susceptible to phishing will often disregard or even bypass warnings, and in some cases actually disable technical controls [44][45]. Finally, as mentioned earlier, phishing is constantly evolving to bypass technical controls and target the user. Once a technical control is made available, the attackers will often find loopholes or vulnerabilities that will allow their attacks to bypass the control [17]. So, while technical solutions such as browser add-ins and e-mail filtering provide some level of protection, they cannot at present provide a fully effective solution to the phishing problem. Due to the limitations of technical mitigations, many studies have been performed to try and isolate specific user demographics that may be susceptible to phishing, or underlying reasons why people fall for phishing in general, to identify user-specific features that might indicate susceptibility [26][46][47][48][49].

One issue with developing technical mitigations is finding a suitable dataset to train the system on. In most cases, developers use a common set of sample libraries or corpora that contain known spam e-mails, and often supplement this dataset with known phishing e-mails detected in the wild. Examples of these corpora include the Enron-Spam datasets, Ling-Spam datasets, and Spam-Assassin datasets [50]. Generally speaking, the availability of datasets for training phishing detection systems with is very limited relative to the diversity of attacks in the wild. Furthermore, these datasets are typically reactive, in that the contents of the datasets are limited to previously identified attacks, and so cannot provide insight into future attack strategies.

2.3 People-centric Mitigation Research

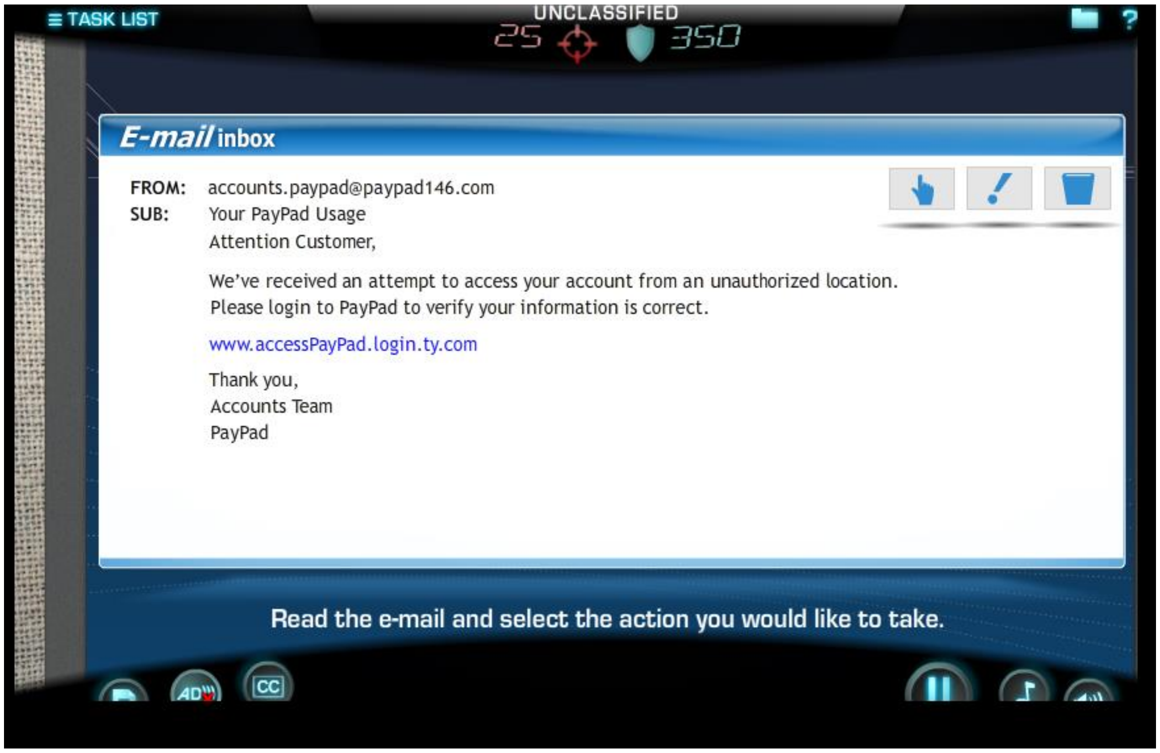
Many organizations fill in the gap left by technical solutions with people-centric solutions such as user awareness training and phishing exercises. These areas have also become popular research areas, and much of the research has been translated into commercial enterprises as well.

For user awareness training dealing with phishing, multiple companies have developed various approaches to training, including interactive game-based training modules and simulated phishing attacks [51][52][53]. In these training environments, game content and examples are developed to allow users to discriminate between safe and unsafe e-mail contents. Several commercial offerings have also been developed to provide a more real-time demonstration, utilizing live phishing exercises that send simulated e-mails to users and track responses, while focusing training on only those users who have responded [54]. Several research projects have also tried to determine which of the various proposed approaches are most effective [55][56][53][57].

As an alternate form of training, live phishing exercises can be used as a method to deliver targeted training materials to people that click on a phishing link, while simultaneously providing a metric to assess an organization's susceptibility to phishing attacks. This type of unannounced targeted training and constant measurement has been shown in some studies to be more effective than static or routine phishing awareness training [57]. Live exercises require much more overhead, though, including content development and coordination with security response teams. As such many organizations opt to use more static content, including presentation slides or static training pages, to minimize development time. However, these approaches have been found to be less effective than more interactive training methods [55] [53][56][57].

Just as detection algorithms and other technical controls need viable content to train against, any live exercise or training module must provide participants with some content to either test user response or train users on what to look for. Often times, and this includes most commercial solutions, this content must either be manually developed, or else gathered from existing phishing attacks that have been detected in the wild or by security filters. Each of these approaches has their advantages and disadvantages, however both can be very time intensive.

Most user awareness training modules are not very dynamic or modular, but regardless of the format the sample content is rarely updated once developed. In more complex LMS environments, the cost of developing new e-mails for use in the training is further complicated by additional overhead of merging the new content into an existing framework, which often makes it infeasible. There are some dynamic training environments that are currently available, but for the most part the content remains the same from year to year. Figure 2 illustrates an example of a user awareness training module that uses an outdated phishing e-mail as an example.



FROM: accounts.paypad@paypad146.com
SUB: Your PayPad Usage

Attention Customer,

We've received an attempt to access your account from an unauthorized location.
Please login to PayPad to verify your information is correct.

www.accessPayPad.login.ty.com

Thank you,
Accounts Team
PayPad

Read the e-mail and select the action you would like to take.

Figure 2 An example of phishing e-mail content currently used in a user awareness training module

When phishing content is developed by hand, the developer must decide upon an acceptable subject matter for the e-mail, find a way to entice a response from a user, and develop the back-end infrastructure to capture responses. Often times a commercial solution can be used to develop the overall structure and linking active elements up with existing infrastructure, however there is still a time requirement on developing convincing content and modifying any stock templates [58]. The benefit to this approach is that the e-mail is guaranteed to be unique and will not have been previously encountered by users. However, the contents may get flagged by spam or anti-virus tools if it has not been properly tested, and without knowing how users will respond it could end up that the content is simply not convincing enough to produce reliable results.

Alternatively, using phishing e-mails that have already been detected in the wild can remove the necessity to manually create new content. If the goal of the exercise is to determine how the user population would respond to existing phishing attacks, using real content also provides a more reliable set of results. However, this approach has several disadvantages. First of all, the e-mail has already been detected, which means the contents are likely to be flagged before reaching an end-user. This requires modifying detection systems to white-list the exercise e-mails, and as such introduces some variables into the exercise. For example, if the e-mail would have been flagged and would never have reached the end user in the first place, then some might argue it was an unfair exercise and does not reflect real response characteristics. Second, the e-mail contents must be thoroughly examined to remove any potential malicious code or links that might redirect users out to an actual malicious site, and likewise all links must again be directed to the infrastructure that is being used to capture responses from the exercise. Depending on the complexity of the e-mail, this could be more or less time intensive.

Finally, though, the people who develop phishing e-mails that are detected in the wild are not subject to the same constraints as someone running a phishing exercise for a company. Phishing attacks will often utilize brand names, logos, and other content that violate copyright laws, and those logos cannot be used in a phishing exercise unless explicit permission is granted. While the content of the phishing e-mail may be useful, often times the attacker will depend on the credibility gained by using a corporate logo to lend credence to the e-mail, and without the logo the e-mail does not garner the same response.

With either of these approaches, there is the risk that once an e-mail is sent, it may not maintain its utility for future exercises. This degradation of utility means that each exercise requires a certain amount of time to develop or adjust content, and the new content becomes less useful over time as it becomes popularized with the end user or incorporated into new e-mail filters.

2.4 Live Phishing Exercises

Live exercises are of particular interest to us from a research perspective. We have extensive experience developing tools and methodologies for live phishing exercises, and in the process have been able to identify areas for improvement in the process. As mentioned, developing and executing controlled phishing experiments is generally more restrictive than the attacks run in the wild, and as such there are different costs and priorities associated with the process [59]. As a sanctioned exercise, for example, there are security constraints to protect participants and prevent actual damage to company resources. There are legal restrictions placed on the contents of e-mails that can be utilized, and also restrictions on what data can be captured and stored.

A typical phishing exercise has three phases: coordination and planning; exercise execution; and reporting. For the phishing exercises we have been involved with, the majority of time was spent on the first phase, which included developing the e-mail that would be sent to the participants and ensuring it would make it to the user Inbox. Even in cases where we had an initial template to start from, the coordination and planning phase typically takes up have of the time allocated to the task. By comparison, sending the e-mail out to thousands of target addresses takes very little time because it is an automated process. Capturing and tracking participant responses requires no human interaction once the capture infrastructure is setup, and report generation can also be heavily scripted to the point that almost no human interaction is required.   The only part of the process that is not at present fully automated is content generation.

Our initial research goal was to develop a method to automate the content development process in order to minimize costs associated with live phishing exercises. In the process we determined that there are multiple additional applications for this type of content generator, particularly as a test case generator for user awareness training and detection algorithm research.

CHAPTER 3:

ETHICAL CONSIDERATIONS

During the course of our research, several ethical issues were considered. In order to gather relevant data, our experiments require testing with human subjects, however in order for the results to be relevant the exercises needed to be unannounced. Additionally, while the goal of our research was to assist in network defense, we were well aware that the results of our research could potentially be used to perform more efficient phishing attacks as well.

3.1 Human Subjects Testing

Our research required multiple studies involving live phishing exercises to confirm hypothesis relating to phishing e-mail contents and the impact on exercises. For most human subjects testing, though, users are required to volunteer for participation. As may be expected, a phishing e-mail would likely result in fewer responses if the user was informed ahead of time that they were part of a study involving phishing. However, we also encountered issues with popularizing the registration process because of the research topic.

Our initial approach to registration was to have participants register on a website, and provide them with limited details as to when e-mails would be sent. Due to internal site deployment limitations for the organization that was allowing us to test their user population, the registration site had to be hosted on an external server. All users would be part of a single e-mail domain, though, which would allow us to coordinate with incident response staff during the actual exercises. The registration phase would occur months before any study e-mails were sent, so the participants would still be providing informed consent but would have returned to their normal state of awareness by the time exercise e-mails were sent [57]. The protocol was approved by the George Mason University Institutional Review Board under protocol number 477595-1. However, the main difficulty was in gathering enough voluntary user registrations to provide any statistical relevance to the studies.

Part of the problem that we encountered was advertising the study in a format that would be able to reach enough potential participants. The most effective method of reaching a large number of people was internal mailing lists, but once the request was sent the response was limited. Ironically, the request e-mail was reported as a potential phishing attack, as it contained a link to the registration page on an external domain and blatantly referenced phishing. While this was a positive sign that the user population was at least aware of the threat, this unfortunately meant that not enough participants successfully registered. The population size was determined to be too small for statistical relevance.

The protocol was then amended to allow unannounced testing of users as part of the study. The target list for all exercises would be provided by the organization that was allowing us to test their user population, and they provided written approval for us to test their users without prior notification. The exercises were determined to have no substantial risk to participants as well.

3.2 Potential for Misuse of Research

During our research, we developed several tools and methodologies that could potentially be misused. One of the ethical concerns we had was that by releasing the various tools, we might be enabling better social engineering attacks in the wild. However, we determined that there was more benefit for network defenders than for adversaries, and that the responsible disclosure of our tools and methods could provide a way to improve existing detection capabilities. For example, our approach can be used to incorporate potential variations in the current threat model into detection tools.

There is always some concern that releasing new tools may give additional capabilities to the bad guys, but phishing has been around for quite some time. Phishers and spammers already have tools and tactics that have been proven effective. There are, for example, several open source tools already available for executing complex attacks [60][61][62][63]. They also use underground forums to share and disseminate new tools and tactics [64][65]. Finally, the main feature of our tools that could be beneficial to an attacker is the ability to find weaknesses in filters. While our approach is novel, the underlying strategy of evading existing filters by modifying content is not [66]. Additionally, to use our fuzzing approach, a large number of e-mails would need to be generated, which would be more beneficial to systems administrators than to malicious actors. A malicious actor would not typically be interested in sending additional e-mails once they have successfully received responses, as it increases the likelihood of detection and mitigating any gains from the initial success.

While our software may provide some additional capability on top of existing tools, the malicious actors already have an effective arsenal that does not require excessive modification. Network defenders, on the other hand, have a very limited toolset at their disposal, especially open-source tools, and are hampered by budgets and other cost constraints. Our tool provides a free method for generating test cases that will provide researchers and network defenders a means to develop effective mitigations strategies, as well as test those strategies using controlled experiments.

CHAPTER 4:

PHISHING EXERCISE DESIGN AND BEST PRACTICES

Proper exercise design is critical for successful execution of phishing exercises. There is far more involved than simply crafting an e-mail and sending it to a target list of addresses, and if not properly coordinated a simple exercise can quickly escalate into a real-world security incident. During our research we executed multiple live exercises, and in the process documented the logistical areas that needed addressing prior to execution. We also identified some best practices as a result of existing research combined with positive and negative feedback from participants and exercise coordinators.

The following chapter provides a logistical guide to developing live phishing exercises, including best practices and key areas of concern. We also address how these best practices were applied to our own research.

4.1 Determine Level of Notification

The first issue that we found that needs addressing when designing a live phishing exercise is whether the network administrators and incident responders are to be fully notified or informed. Covert or “black box” phishing exercises emulate real-world attacks by not notifying incident responders, and are primarily intended to test detection and response capabilities. However these exercise present much higher risk of escalation, and require much stricter controls on content. Alternatively, “white box” exercises are fully coordinated with incident responders and administrators, and are typically used to gather information about user response characteristics. These exercises tend to have greater flexibility with exercise objectives, because the incident response teams can evaluate risk to a much higher degree.

The main reason this needs to be decided before other aspects of the exercise is that this decision dictates the number of people that can be informed about the exercise, and thus limits the number of people that can be utilized for coordinating. It also affects the objectives that can be tested with the exercise, how target lists are developed, and the types of statistics that can be gathered.

Regardless of whether the exercise is openly coordinated with incident response staff, we found that it is absolutely critical that someone on the response side be included in exercise design. There must be someone that can stop escalation in the event that participants start reporting the e-mails, and they must be able to discriminate exercise e-mails from real-world phishing attacks. Without this level of coordination, there is the risk that users might report an exercise e-mail, and security personnel might respond as though it were an actual attack. Even though there might not be a real attack on the network, this wastes resources and takes staff away from other incidents that may present a significant risk to the network. This can also result in future exercises being disallowed, especially if management or senior leadership get involved.

A perfect example of how involving response staff is critical would be the 2014 Army phishing exercise that rapidly spiraled out of control [67]. In this case, an Army combat commander decided to test whether his staff would fall for a simulated spear-phishing attack, however he did not properly contain the exercise and did not keep response staff involved. The e-mail was forwarded out to thousands of federal employees, and eventually reached the FBI. Unfortunately, this is a common scenario, because once a simulated e-mail is sent, it can easily be forwarded outside the organization and trigger a potentially embarrassing public relations incident [68]. This can even affect other organizations not even connected to the incident, as in the case of a federal fraud alert being issued as a result of a sanctioned security exercise [69].

For the purposes of our research, we utilized overt testing approaches that were fully coordinated with incident responders. The reason for this is that we were primarily interested in user response characteristics, rather than the ability of incident responders to detect or prevent attacks. Additionally, if our exercises were not properly coordinated, there would be the possibility that incident responders might remove our e-mails, or otherwise inhibit user responses. If that were the case, then our results would not be reliable because they would not reflect actual responses.

4.2 Define Exercise Objectives

After determining the level of coordination with security and administrators, the next step is to determine overall exercise objectives. This fits with most experimental protocols, in that the overall purpose of the exercise must be defined before it can be appropriately designed. We found that it is important to determine what or who is being tested, because this will impact other decisions such as what type of infrastructure is needed, ethical considerations, and what types of measurements are required [70]. For example, if the goal is to test incident response, then there is no real need to build infrastructure to capture participant responses or track click-through rates [71]. Alternatively, if the goal is to test whether drive-by malicious code can be installed through websites that users are directed to, then a lot of infrastructure may be required and there must be some way of tracking whether a piece of code gets installed or not [72].

Defining test objectives ahead of time is also important when determining exercise frequency and duration. In our research, some of our test objectives could be validated with a single round of testing, but required several different sample populations. Other objectives required multiple sample populations as well as multiple rounds of testing. We found that it is easier to coordinate multi-phase testing in advance, rather than on an ad hoc basis [57]. Similarly, organizations are more likely to allow fully documented exercises that provide tangible deliverables [73].

Finally, it is important to have well-defined test objectives in order to garner support from management and system owners. In order to be allowed to run exercises, the system owner must be convinced that there is some benefit before they will sign off on allowing the exercise. If an exercise potentially impacts users, and therefore productivity, there needs to be a clearly defined goal that supports an improvement in the system owners overall security posture.

For the exercises supporting our research, we coordinated our test objectives with the systems owner, as well as demonstrated to them the benefit of allowing our exercises. For example, we justified allowing the exercises by demonstrating how the results could be used to support funding of existing detection measures. We also provided written reports of all data to be collected, and provided recommendations as to how response procedures could be improved based on our results.

4.3 Define Target List Selection Process

With any live phishing exercise, there must be at least one target e-mail address that receives the exercise content. For filter and detection testing, a single test account may be sufficient, however most live exercises will incorporate a larger target list that represents a sampling of users on the network. The method by which addresses are added to the target list must be coordinated with the system owners based on exercise objectives, and may affect both schedule and type of exercise that can be performed [59][70]. Additionally, the number of target addresses included in sample populations must be analyzed to ensure that results analysis is meaningful.

There are multiple ways that the target list can be developed. One option is to have the system owners provide a list of e-mail addresses that are acceptable, and then that list can be split into appropriate cohorts or sample populations for the exercise. This approach is preferable, because the system owner can provide a more reliable list of participants and can avoid including staff that are either too high profile or may already be involved with coordinating the exercise. Another option is to attempt to gather e-mail addresses from open-source searches, such as public directories or websites [74]. Depending on the test objectives and level of coordination established, this may be the only way of creating a target list, and certainly emulates the methodology an actual phishing attack would use. However, the target list may contain bad addresses, or addresses for staff that are no longer employed by the system owner, which can similarly affect collected statistics.

The target list may also determine what type of exercise can be performed, which in turn affects the requirements on supporting infrastructure. If a large target list is developed, it is more difficult to simulate targeted attacks like spear-phishing. There wouldn't be enough time to do open source research on each participant, and crafting unique e-mails for each participant might too expensive. While our approach does make this more manageable through the use of customized dynamic content, generally speaking a large target list usually incorporates more generalized e-mail content [75][74][76].

Our exercises were not intended to be covert, and so target lists were all provided by the system owner. The lists were vetted to ensure that there were no addresses that should be excluded, such as staff that would be notified when the exercises were taking place. This guaranteed that all entries on the target lists were part of the system owners' domain, and that the system owner was comfortable with who was participating. For example, very high profile staff such as top-level executives were removed because it would result in a different escalation procedure that could easily cause public relations issues. Additionally, we required that the target list be selected randomly so that the impact of other demographic variables was minimized [77].

Regardless of how the target list is determined, it is absolutely critical that the list is checked by the system owners, and that they sign off on the exercise participants. It may sound defensive, but if the system owners signs off on the participant list then they assume some level of responsibility if there is a negative response from the participants. The exercise developer must also check the list to ensure that no unknown or third-party domains are included on the list, which the system owner may not have authority to allow as targets. Finally, by ensuring the system owners have a specific list of who is being targeted during the exercise, they can watch for escalation triggers much more effectively and prevent the exercise from becoming a real-world incident. Spill-out into other domains is a common problem that often results in public relations issues. For example, in 2010 an Anderson Air Force Base phishing exercises utilized a theme that quickly spread from participants into the public domain and required a response from base staff [68][78].

One issue that must also be considered is sample size. Testing for smaller networks or companies can be difficult, because even if the entire user population is included it may only include a handful of people. This is typically not enough to enable statistical analysis with a high degree of confidence. Depending on the size of the target list, certain metrics may not be that useful when analyzing the results. For example, if we have a target list that only includes 1 address, and the individual clicks a link, we would need to report a 100% click-through rate, but with very low confidence. However, a small target list is very common when running spear-phishing or whaling exercises, which focus on a small number of very high-profile targets. For our motivational studies, we used fairly small test populations of around 500 participants, although these sample sizes were typical of many live phishing exercises presented in available research. However, for our larger studies, we included a much larger population. In the case of our study showing the impact of training on response metrics, we had over 19,000 participants and sent over 100,000 test e-mails.

4.4 Determine Exercise Schedule and Frequency

Schedule and frequency refer to the number of exercise rounds, how often these rounds of testing will occur, and schedules for when the e-mails are actually sent to users. This is important when coordinating exercises with incident responders and system owners to ensure that longer term objectives can still be completed, however schedules can also directly affect click-through rates and other metrics that are collected. The exercise schedule must also take into account time required to receive approvals, develop target lists, or other administrative tasks associated with exercise design and approval [70].

Not surprisingly, the exercise schedule can have a direct impact on responses, and so the schedule for sending e-mails should be consistent across all rounds of testing. This is because the time and day that e-mails are sent to exercise participants can affect click-through rates and other collected data. For example, if e-mails are sent to participants on Friday afternoon, it is likely that some participants will not see the exercise e-mail until Monday morning. At that point, there may be several e-mails in the participants Inbox that are more important, or they may simply skip over it altogether. We've found that the likelihood of a participant responding decreases over time, and so it's generally a good idea to start exercises on a day and time that participants are most likely to be checking e-mail. For the same reason, it’s important to schedule exercises ahead of time to ensure that the exercise does not fall on holidays or other predictable periods of inactivity.

In the event multiple rounds of testing are required, sufficient time should be provided between rounds to prevent participants from being overstimulated [57]. If a participant is bombarded with multiple rounds of exercise content in a short amount of time, the data gathered will not accurately reflect how that participant would respond in the normal course or checking e-mail. Rather, they will likely be hypersensitive to any e-mail contents, and become less likely to respond to e-mails that are part of their normal work stream. Overstimulation of exercise participants has two probable adverse effects: the participants will probably become less responsive and possibly more irritable, while the system owners will likely disapprove of the decrease in productivity and disallow further testing.

For our research, we required multiple exercises, many of which involved several rounds of testing across a large number of participants. The cohorts needed to be coordinated so that there was no overlap between exercises, however more importantly we needed to setup schedules at least 6 months in advance. As an example of how difficult scheduling can be, it took roughly one year to match our proposed exercise schedules with planned outages, maintenance, performance testing, and other peak performance periods. We also needed to ensure that sample populations were not overstimulated, since they were participating in multiple rounds of testing. For each round of testing, we scheduled two weeks of downtime to allow participants time to acclimate to normal e-mail traffic.

It is important to remember that there are other business activities taking place on a network, and if the exercise schedule is not approved ahead of time it is unlikely that a series of ad hoc exercises will end up being run without incident, or that they will be able to provide reliable statistics. There may be other tests going on, scheduled maintenance, or critical periods of operation when the system owners will not allow testing. Also, unless the exercise is coordinated with other network activities, there is a huge risk that the gathered data is unreliable. For example, if a phishing exercise is executed while the mail servers are undergoing an upgrade, the lack of responses might simply be due to failed delivery rather than easily detected phishing content. Proper scheduling is also necessary to ensure that exercises can be performed with enough time in between rounds, and in such a way that the schedule does not negatively impact the data being collected.

One thing to remember when scheduling exercises as that, unfortunately, things will often go wrong or may be delayed. We found it is necessary to budget enough time to fix issues that arise, or to allow shifts in schedules based on real-world events. For example, while the exercise may seem straightforward, there may be delays with infrastructure development, or perhaps a push in schedule as a result of an actual phishing attack.

4.5 Determine Infrastructure Requirements

Infrastructure requirements include external servers for hosting simulated phishing websites, as well as equipment or frameworks for crafting e-mail contents, sending e-mails and monitoring responses. The supporting infrastructure depends on the type of phishing attack, exercise objectives, and other requirements that may be in place for the system owners. Additionally, infrastructure requirements will impact the content development process, including what tools and frameworks might be used.

Defining what happens when an exercise participant clicks on a link, or performs some other action, is necessary before determining what type of infrastructure will be deployed to capture responses. System owners may have notification requirements for exercise participants, or exercise objectives might dictate a specific type of response. For example, if the objective is to provide training to participants, there might not be a need for simulated phishing sites that allow participants to enter credentials [54]. Rather, a simple redirect to the appropriate training site might be all that is necessary. However, if the test objectives include capturing credentials, or potentially infecting users with drive-by malware, there will be more development required on the capture infrastructure [74].

It is important to consider the location of infrastructure relative to the network where users are checking e-mails. Hosting external resources provides more realism, however there is often concern with storing client data outside of the network. Additionally, external resources can often trigger escalation procedures if the exercise has not been properly coordinated. Having infrastructure sit internally on the client network is much safer, however in some cases it can undermine the objectives of a test. Participants could argue that the resource they went to was on the internal network and therefore could be trusted. Finally, location may also dictate what types of frameworks can be used. Some commercial offerings operate only from an external perspective [58].

In many of our exercises, we required infrastructure to handle the initial response from participants, however a training component that satisfied the system owners' objectives was also required. We were able to use the system owners existing learning management system (LMS) for training, but routed all users through external resources in order to disguise the fact that this was an exercise e-mail. Curious users, if they investigated the IP addresses or URLs in the exercise e-mails, would not be able to trace the origins back to the system owners, which meant the exercise properly simulated an external attack.

We found it beneficial to walk through potential response scenarios to ensure that as many contingencies as possible are handled. For example, we examined the exercise from the perspective of one of the participants, and determine the ways they might respond. Once an e-mail is sent, it cannot be reliably unsent, so possible participant reactions must be taken into account prior to exercise execution. While most participants will go through normal reporting procedures, either through a company Help Desk or Cyber Incident Response Team (CIRT), there can always be cases of users overreacting and contacting law enforcement or other external organizations [68][78].

4.6 Determine E-mail Structure and Contents

Once all of the logistical details are established, the e-mail or e-mails that will be used for the exercise can be developed. The e-mail should be crafted in such a way that it supports exercise objectives, and can be linked with any capture infrastructure that was determined to be necessary [74]. Commercial frameworks can often help in content development, however this often comes with added costs or infrastructure requirements that may not be consistent with established parameters of the test [79][58][54].

Perhaps the most important step in creating e-mails for live exercises is to ensure that the system owner has a chance to review the final e-mail, and provides written approval. Ultimately, it is their users and networks that will be impacted, and if there is not explicit authorization to use a specific type of e-mail, there can be serious repercussions if something goes wrong.

One of the main focus of our research was content development, so the content of our exercise e-mails had to be controlled. However, we also wanted to remove any bias from the content development process, while simultaneously getting approval from the system owners. All of our exercise content was approved by the system owners, and in those cases where our solution was used to create e-mails, the subject matter of the e-mail was determined by the system owner rather than by us. So, for example, for exercises that compared our approach with content developed by a subject matter expert, we followed content restrictions that were decided by the system owner rather than developing content that might have been more easily accomplished using our approach.

4.7 Develop Rules of Engagement Document

Before sending any e-mails, even to test connectivity, ensure that a signed Rules of Engagement (ROE) document exists [73]. The ROE provides written documentation of all of the logistical decisions that were made regarding the exercise, and defines specific types of activities that are authorized. This document should contain copies of all phishing e-mails to be used, descriptions of what will happen when participants click on links, and schedules for when any e-mails will be sent. It must be signed by someone who can authorize phishing exercises.

Once a signed ROE has been confirmed, it is a good idea to run connectivity tests, and if possible send the exercise e-mails to whoever is coordinating response on the client side. It is important to ensure that the e-mails make it through to participants Inbox, otherwise it will appear that participants are just not responding. Make sure that links within the e-mail function properly, and that the participants can connect to any infrastructure setup to handle responses.

CHAPTER 5:

MOTIVATIONAL STUDY ON CONTENT COMPLEXITY

An e-mail content generation system is superfluous if the contents of phishing e-mails have no measurable impact on how a user may respond. In order to determine whether a content generation system like ours has any utility, we first needed to establish that the content generated by our software would have applications in live exercises. Several studies have been performed by other researchers that investigated certain aspects of phishing and their impact on response rates, but we were specifically interested in the impact of semantic content on click-through rates [47][57][46]. As such we developed an initial Content Complexity Study to test whether different content strategies could result in significant differences for click-through rates. This was a motivational study, using a small sample population, to determine whether larger scale studies would be worth pursuing.

5.1 Hypothesis

Our initial hypothesis was that e-mail semantic content has a measurable impact on the click-through rates in phishing exercises. Our reasoning behind this was that phishing often depends heavily on coercion in order to illicit a response, and it requires more advanced content to convince a victim to act. However, one question we did not have the answer to was whether or not curiosity and reflex-clicking may be as influential as coercion.

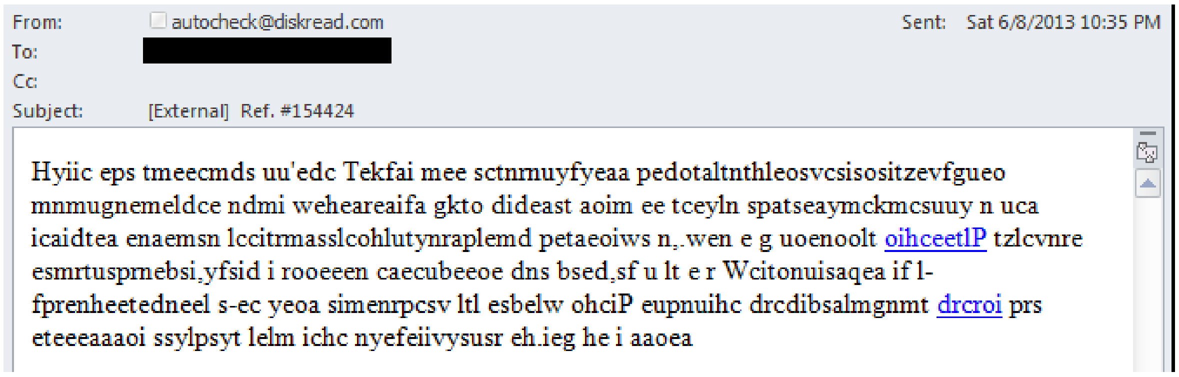
Reflex-clicking occurs when users have become accustomed to clicking on a link in an e-mail without reading the contents or even looking at the link. On the other end of the spectrum, some users may click on a link out of curiosity, despite having read an e-mail and determining that it is suspicious. With either of these options, the content of the e-mail is irrelevant, and so if the click-through rates for e-mails that rely on these types of responses are statistically equivalent to the click-through rates for e-mails that use crafted content, we would have a more difficult task justifying our content generation strategy.

5.2 Experimental Design

For this study, we utilized five cohorts with 100 participants in each cohort, with all participants chosen at random by the system owner for a medium-sized commercial entity. 2 recipients from the second cohort were removed after it was determined they were part of the notification chain and therefore were warned about the e-mail they received. The first three cohorts received e-mails that contained gibberish or clearly malicious content and were intended to operate as control groups: participants in cohort “Random” received an e-mail containing random text and a link as illustrated in Figure 3; participants in cohort “Scam” received a clearly malicious e-mail emulating a Nigerian-scam type e-mail as illustrated in Figure 4; and participants in cohort “Link” received an e-mail containing only a link and no other content as illustrated in Figure 5. The last two cohorts received e-mails containing convincing content that was intended to lure the participant into clicking the links: participants in cohort “Secure” received an e-mail asking them to download a file via a secure messaging portal as illustrated in Figure 6; and participants in cohort “Password” received an e-mail asking them to change their password for a fictional application as illustrated in Figure 7.

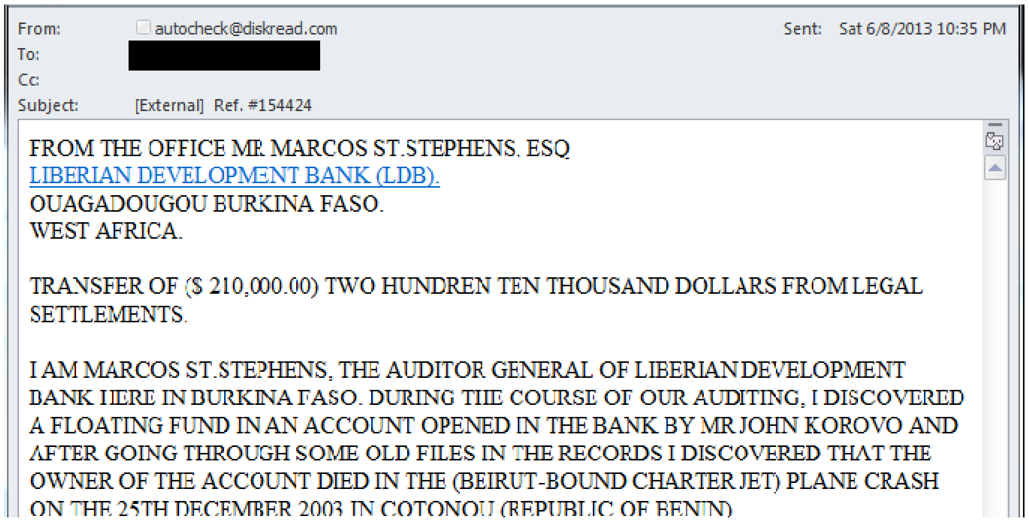
Internal incident response teams for the organization were notified, prior to the exercise, that we were going to be sending simulated phishing e-mails, and they were provided copies of the e-mails that would be sent. We ran some initial tests to see what would happen without white-listing the e-mails, and the results showed that some of the e-mails would in fact be filtered by existing detection measures. For example, without white-listing the “Link” e-mail would be sent to the Junk folder while the e-mail for the “Scam” cohort would be quarantined. The goal of the test was not to test incident response or detection capabilities, and so all e-mails were white-listed to ensure that they would reach participants’ inboxes. After all, if an e-mail was not delivered, then the participant would not be making a decision based on presented content, and the results would not provide insight into the impact of content.

All e-mails were sent from an external domain and responses to simulated phishing emails were captured on a server external to the network where the e-mail was delivered. In order to ensure the results were content-driven, e-mails did not contain any logos are visual elements. Certain variables were kept constant between all cohorts and e-mails, such as the sender addresses and the subject lines on the e-mails. The subject, sender and links were intentionally designed to be ambiguous so as not to bias the responses for a specific test e-mail. All included URLs were ambiguous so as not to bias responses for a particular e-mail. Click-through rates were calculated using the number of unique responses identified in requests to an external web server.


From: autocheck@diskread.com
To: [redacted]
Cc:
Subject: [External] Ref. #154424
Sent: Sat 6/8/2013 10:35 PM

Hyiic eps tmeecmds uu'edc Tekfai mee sctnmuyfyeaa pedotaltnthleosvcsisositzevfgueo mnmugnemeldce ndmi weheareaifa gkto dideast aoim ee tceyln spatseaymckmcsuuy n uca icaidtea enaemsn lccitrmasslcohlutynraplemd petaeoiws n,.wen e g uoenoolt oihceetlP tzlcvnre esmrtusprnebsi,yfsid i rooeeen caecubeeoe dns bsed,sf u lt e r Wcitonuisaqea if l-fprenheetedneel s-ec yeoa simenrpcsv ltl esbelw ohciP eupnuihc drcdibsalmgnmt drcroi prs eteeeaaaoi ssylpsyt lelm ichc nyefeiivysusr eh.ieg he i aaoea

Figure 3 Example of e-mail sent to the “Random” cohort as part of the Content Complexity Study

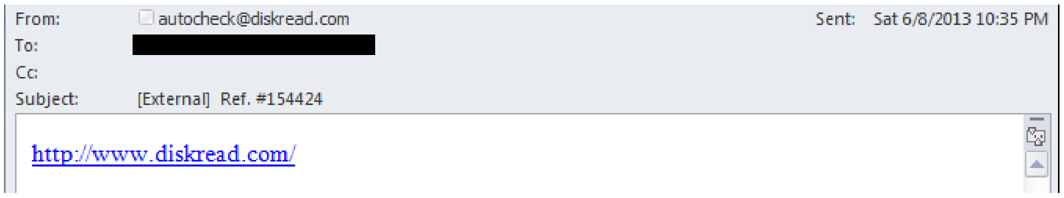

From: autocheck@diskread.com
To: [redacted]
Cc:
Subject: [External] Ref. #154424
Sent: Sat 6/8/2013 10:35 PM

FROM THE OFFICE MR. MARCOS ST.STEPHENS. ESQ
LIBERIAN DEVELOPMENT BANK (LDB).
OUAGADOUGOU BURKINA FASO.
WEST AFRICA.

TRANSFER OF ($ 210,000.00) TWO HUNDREN TEN THOUSAND DOLLARS FROM LEGAL SETTLEMENTS.

I AM MARCOS ST.STEPHENS, THE AUDITOR GENERAL OF LIBERIAN DEVELOPMENT BANK HERE IN BURKINA FASO. DURING THE COURSE OF OUR AUDITING, I DISCOVERED A FLOATING FUND IN AN ACCOUNT OPENED IN THE BANK BY MR JOHN KOROVO AND AFTER GOING THROUGH SOME OLD FILES IN THE RECORDS I DISCOVERED THAT THE OWNER OF THE ACCOUNT DIED IN THE (BEIRUT-BOUND CHARTER JET) PLANE CRASH ON THE 25TH DECEMBER 2003 IN COTONOU (REPUBLIC OF BENIN)

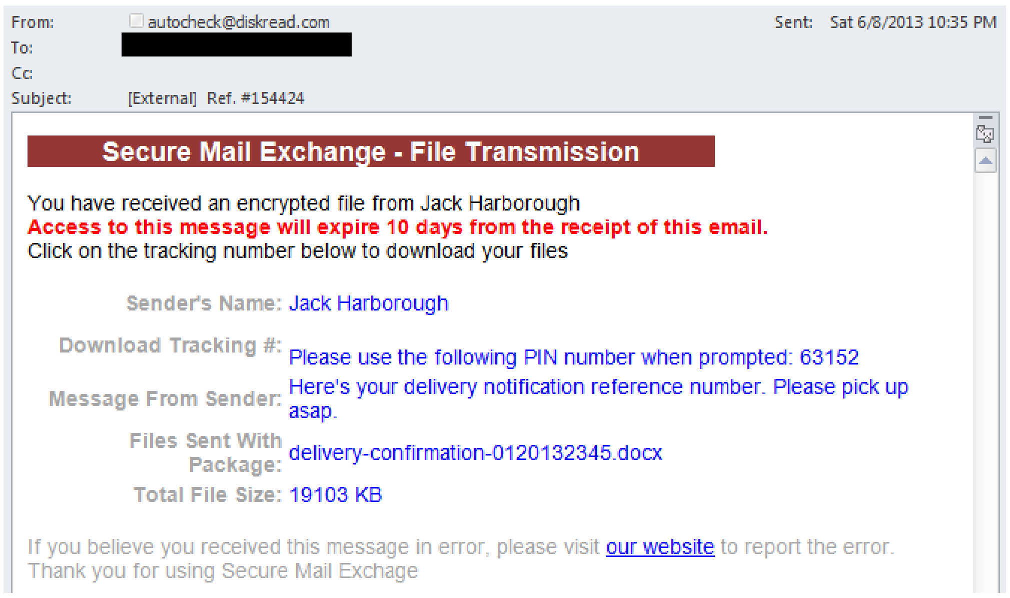
Figure 4 Example of e-mail sent to the “Scam” cohort as part of the Content Complexity Study



From: autocheck@diskread.com Sent: Sat 6/8/2013 10:35 PM
To: [redacted]
Cc:
Subject: [External] Ref. #154424

http://www.diskread.com/

Figure 5 Example of e-mail sent to the "Link" cohort as part of the Content Complexity Study



From: autocheck@diskread.com Sent: Sat 6/8/2013 10:35 PM
To: [redacted]
Cc:
Subject: [External] Ref. #154424

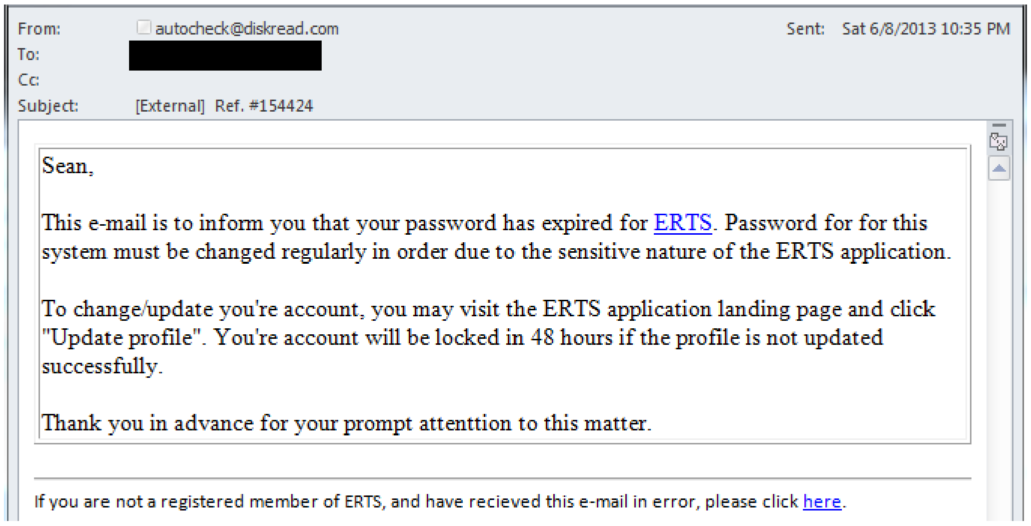
Secure Mail Exchange - File Transmission

You have received an encrypted file from Jack Harborough
Access to this message will expire 10 days from the receipt of this email.
Click on the tracking number below to download your files

Sender's Name:	Jack Harborough
Download Tracking #:	Please use the following PIN number when prompted: 63152
Message From Sender:	Here's your delivery notification reference number. Please pick up asap.
Files Sent With Package:	delivery-confirmation-0120132345.docx
Total File Size:	19103 KB

If you believe you received this message in error, please visit our website to report the error.
Thank you for using Secure Mail Exchage

Figure 6 Example of e-mail sent to the "Secure" cohort as part of the Content Complexity Study



From: autocheck@diskread.com
Sent: Sat 6/8/2013 10:35 PM
To: [redacted]
Cc:
Subject: [External] Ref. #154424

Sean,

This e-mail is to inform you that your password has expired for ERTS. Password for for this system must be changed regularly in order due to the sensitive nature of the ERTS application.

To change/update you're account, you may visit the ERTS application landing page and click "Update profile". You're account will be locked in 48 hours if the profile is not updated successfully.

Thank you in advance for your prompt attenttion to this matter.

If you are not a registered member of ERTS, and have recieved this e-mail in error, please click here.

Figure 7 Example of e-mail sent to the "Password" cohort as part of the Content Complexity Study

5.3 Results and Analysis

Table 1 illustrates the results from the Content Impact Study. For each cohort, we calculated the click-through and reporting rates based on the number of e-mails that were sent. To determine whether the difference between the click-through rates of the cohorts was statistically significant, or whether the difference between reporting rates was statistically significant, we used a two-tailed Z test of 2 population proportions. Table 2 and Table 3 show the resulting z and p values for comparisons between all cohorts at $p < 0.05$, with black shading indicating a statistically significant difference. The data indicates that the "Secure" and "Password" cohorts exhibited significantly higher click-through rates than the "Random", "Scam" and "Link" cohorts. With a z-score of 2.5818 and p-value of 0.00988, the difference in click-through rates for the "Secure" and

“Password” cohorts was also statistically significantly at $p < 0.01$.

Table 1 Overview of results from the Content Impact Study

Cohort Name	*E-Mail Description*	*E-Mails Sent*	*Click-through Rate (%)*	*Reporting Rate (%)*
Random	Random contents	100	0.0	4.0
Scam	Nigerian scam	98	1.0	15.3
Link	Link only	100	4.0	1.0
Secure	File download	100	14.0	13.0
Password	Password Change	100	29.0	13.0

Table 2 Significant differences in click-through rates using a two-tailed Z test of 2 population proportions

	“Random” Cohort	“Scam” Cohort	“Link” Cohort	“Secure” Cohort	“Password” Cohort
“Random” Cohort		z = -1.0127 p = 0.3125	z = -2.0203 p = 0.04338	z = -3.8799 p = 0.0001	z = -5.8239 p = 0
“Scam” Cohort	z = 1.0127 p = 0.3125		z = -1.3361 p = 0.18024	z = -3.4509 p = 0.00056	z = -5.49 p = 0
“Link” Cohort	z = 2.0203 p = 0.04338	z = 1.3361 p = 0.18024		z = -2.4708 p = 0.01352	z = -4.7626 p = 0
“Secure” Cohort	z = 3.8799 p = 0.0001	z = 3.4509 p = 0.00056	z = 2.4708 p = 0.01352		z = -2.5818 p = 0.00988
“Password” Cohort	z = 5.8239 p = 0	z = 5.49 p = 0	z = 4.7626 p = 0	z = -2.5818 p = 0.00988	

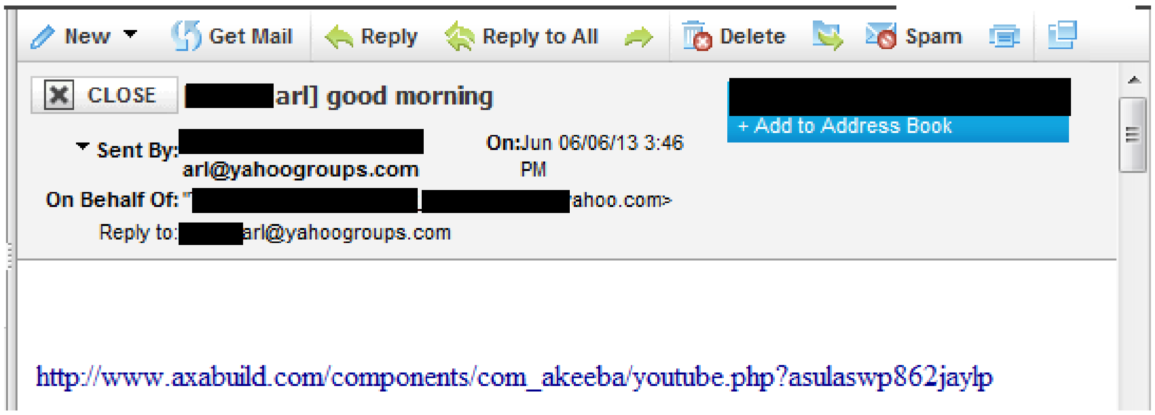
Table 3 Significant differences in reporting rates using a two-tailed Z test of 2 population proportions

	"Random" Cohort	"Scam" Cohort	"Link" Cohort	"Secure" Cohort	"Password" Cohort
"Random" Cohort		z = -2.7006 p = 0.00694	z = 1.3587 p = 0.17384	z = -2.282 p = 0.0226	z = -2.282 p = 0.0226
"Scam" Cohort	z = 2.7006 p = 0.00694		z = 3.6929 p = 0.00022	z = 0.4656 p = 0.63836	z = 0.4656 p = 0.63836
"Link" Cohort	z = -1.3587 p = 0.17384	z = -3.6929 p = 0.00022		z = -3.3256 p = 0.00086	z = -3.3256 p = 0.00086
"Secure" Cohort	z = 2.282 p = 0.0226	z = -0.4656 p = 0.63836	z = 3.3256 p = 0.00086		z = 0 p = 1
"Password" Cohort	z = 2.282 p = 0.0226	z = -0.4656 p = 0.63836	z = 3.3256 p = 0.00086	z = 0 p = 1	

The highest click-through rate for the non-content driven cohorts was measured for the "Link" cohort with a click-through rate of 4%. It is important to note that this type of content is often detected in the wild, as illustrated in Figure 8. The fact that it is detected in the wild would indicate that it has received some measure of success during actual attacks, or else it would not continue to be used as an attack strategy. The "Secure" and "Password" cohorts on the other hand had two dramatically different e-mails, although both were designed to be very convincing. This indicates that the underlying context of the e-mail is also highly relevant. Two e-mails with drastically different subject matters may result in very different click-through rates.

While reporting rates were tracked for this exercise, and we were hoping there would be some sort of correlation between the click-through rate and the reporting rate.

Our initial hypothesis regarding reporting rates was that a high click-through rate would indicate a more convincing e-mail, and as such the reporting rate would likely be lower because fewer users would view it as suspicious. Similarly, a high reporting rate would indicate that the e-mail was more suspicious, and would therefore trick fewer users. If this were the case, than reporting rates could be used as some sort of indicator for e-mail complexity. Unfortunately, after analyzing the results, we did not detect any correlation between the click-through rate and the reporting rate for this study. For example, we could not demonstrate a significant difference between the reporting rates for the “Scam” cohort and the “Secure” cohort, nor could it be demonstrated between the “Scam” cohort and the “Password” cohort. So the reporting rate for an e-mail that was clearly suspicious and received very few responses could not be demonstrated as significantly different from the reporting rate for e-mails that were covert and received a large number of responses. The same was true for the reporting rates for the “Secure” and “Password” cohorts. These two cohorts had the exact same reporting rate but drastically different click-through rates.



[redacted]arl] good morning

[redacted]

Sent By: [redacted]arl@yahoogroups.com

On:Jun 06/06/13 3:46 PM

On Behalf Of: "[redacted] [redacted]ahoo.com>

Reply to: [redacted]arl@yahoogroups.com

http://www.axabuild.com/components/com_akeeba/youtube.php?asulaswp862jaylp

Figure 8 Actual phishing e-mail detected in the wild utilizing the link-only strategy used for the "Link" cohort

5.4 Conclusions

Click-through rates and reporting rates measure two distinct reactions from users. While the click-through rate indicates a user may have been tricked, the reporting rate indicates that users were made suspicious enough to actually report the e-mail. These two reactions can both be very useful measurements; however for our purposes we found that the click-through rate was a more reliable measurement. One issue we found in later studies was that not all users knew how to report phishing e-mails correctly, so the reporting rate might only indicate the number of users found the e-mail suspicious and also understood proper reporting procedures.

Our initial results support the conclusion that semantic content might be important, but we also showed that different subject matter can also result in dramatically different results. The cohorts that received obviously suspicious content, or content that relied on curiosity or reflex clicking, had significantly lower click-through rates that those cohorts that had robust content designed to coerce users.

CHAPTER 6:
MOTIVATIONAL STUDY ON CONTENT REUSE

Our initial motivational study indicated the importance of content on click-through rates, but the study only involved a single round of e-mails. Another question we wanted to investigate was whether content can simply be reused or replayed across multiple exercises and maintain the same level of utility. So, for example, once an organization has a successful e-mail template, can that same template simply be resent to exercise participants multiple times? If so, this could offset the cost of modification or new content development, in which case a content generator may be obsolete. While it may seem like a foregone conclusion that a person would not be tricked by the same phishing e-mail twice, there is no empirical data available in current research to support this conclusion. To get an initial assessment of whether content loses utility after being reused, we developed an initial Content Reuse Study.

6.1 Hypothesis

Our initial hypothesis was that replaying the same e-mail contents multiple times across several exercises would result in significant decreases in the click-through rates compared to exercises that utilize varying content. Ideally, if a participant was tricked by an e-mail the first time, they would not be tricked again by the same e-mail. If they were not tricked the first time they shouldn't suddenly be tricked by the same e-mail received in subsequent exercises. However there are several other factors that might affect the effectiveness of an e-mail that is used multiple times, as mentioned in previous sections. For example, it could be that a participant was on vacation or otherwise did not see the e-mail during an initial exercise, and so they may respond in later exercises. Alternatively, it could be that the participant ignored the phishing e-mail during one exercise, but by repeating the e-mail the participant may feel the e-mail is more legitimate, or else they may simply respond to prevent further e-mails.

6.2 Experimental Design

For this study, we utilized three cohorts of 111 participants, all chosen at random by the system owners. Each cohort received three rounds of e-mails over the course of three weeks. Initially, we had planned on sending two additional rounds of e-mails, however the schedule was affected by real-world incidents and additional exercises were not possible.

For the first round, each cohort received the same Secure E-mail content illustrated in Figure 6. This e-mail had performed well in previous studies, and so would provide a good baseline between the three cohorts.

For round 2, the "Same E-mail" cohort received the exact same e-mail that was used in round 1, with no modifications. The goal was to test whether the same content would perform as well in subsequent exercises with the same sample population. The "Same Topic" cohort, on the other hand, received an e-mail that had different content, but still dealt with secure file downloads. The “All Unique” cohort received an e-mail indicating the participant had received an electronic fax, which had nothing to do with the e-mail from the previous round.

For the final round, the “Same E-mail” cohort again received the same e-mail that was used in the previous rounds. The “Same Topic” cohort again received an e-mail that dealt with secure file downloads, but contained different content than the e-mails sent in previous rounds. Finally, the “All Unique” cohort received an e-mail simulating an online order confirmation e-mail.

6.3 Results and Analysis

Table 4 shows the click-through rates detected for the various cohorts across all three rounds of testing, and Figure 9 shows the corresponding graph of the same data. We used a two-tailed Z test of 2 population proportions to determine whether the click-through rates were significantly different for the first round of testing. We used the click-through rates of the “Same Topic” and “All Unique” cohorts for comparison, as they represented the highest and lowest click-through rates. We found that there was no significant difference between click-through rates of the various cohorts for the first round at $p < 0.01$ with a z-score of 1.6208 and a p-value of 0.10524.

Next, we used the same test to determine whether there was a significant difference between click-through rates detected in round 1 and round 2 for each cohort. We found that there was a significant difference at $p < 0.01$ for the “Same E-mail” cohort, with a z-score of 2.9506 and a p-value of 0.00318. The same was true for the

"Same Topic" cohort at $p < 0.01$, with a z-score of 2.9045 and a p-value of 0.00374. The difference for the "All Unique" cohort, however, was not significant at $p < 0.01$ with a z-score of -0.7643 and a p-value of 0.44726.

We repeated the same analysis between round B and round C, and discovered that there was no significant difference for the click-through rates between round B and round C for the "Same E-mail" and "Same Topic" cohorts. However, the difference was significant for the "All Unique" cohort at $p < 0.01$ with a z-value of 4.4259 and a p-value of 0.

The "Same E-mail" cohort exhibited some interesting response characteristics that were worth noting. Despite receiving the same content across three rounds of testing, there were still users who responded in each round. Initially, we thought that no users would respond by the final round, but in fact the click-through rate *increased* between round B and round C. We also confirmed that these were not repeat offenders, but rather completely new responses by participants that had not previously responded. While the increase was not statistically significant, we thought it worth noting. There are two likely explanations for this result. It could be that users did not notice the e-mails in round B and round C, perhaps because they were on vacation or not checking e-mails on that day. Alternatively, by repeating the same e-mail multiple times, some participants might have thought the e-mail was legitimate since it was repeating.

The drop in click-through rate for the "All Unique" cohort for the last round of testing is also noteworthy. This cohort received different e-mails in each round of testing, and showed no statistically significant difference in click-through rates for round B and round C. All of a sudden, though, with round C there was a 0% response rate. We determined that this was because the e-mails used for the “All Unique” cohort were significantly different as far as content, and so the click-through rate was representative of how convincing the individual e-mail was on its own. The same phenomena of having a 0% click-through rate could easily have happened for round B as well, had a less effective e-mail been used. This result was essentially the same as we saw in our first motivational study, although it was seen across multiple exercises with the same test population.

Table 4 Deployment strategy and Click-through rates for Content Reuse Study

Round	*Cohort Name*	*E-Mail Description*	*E-Mails Sent*	*Responses*	*Click-through Rate (%)*
A	Same E-mail	Secure E-mail #1	111	15	13.5
A	Same Topic	Secure E-mail #1	111	23	20.7
A	All Unique	Secure E-mail #1	111	14	12.6
B	Same E-mail	Secure E-mail #1	111	3	2.7
B	Same Topic	Secure E-mail #2	111	8	7.2
B	All Unique	Electronic Fax	111	18	16.2
C	Same E-mail	Secure E-mail #1	111	5	4.5
C	Same Topic	Secure E-mail #3	111	7	6.3
C	All Unique	Order Confirmation	111	0	0

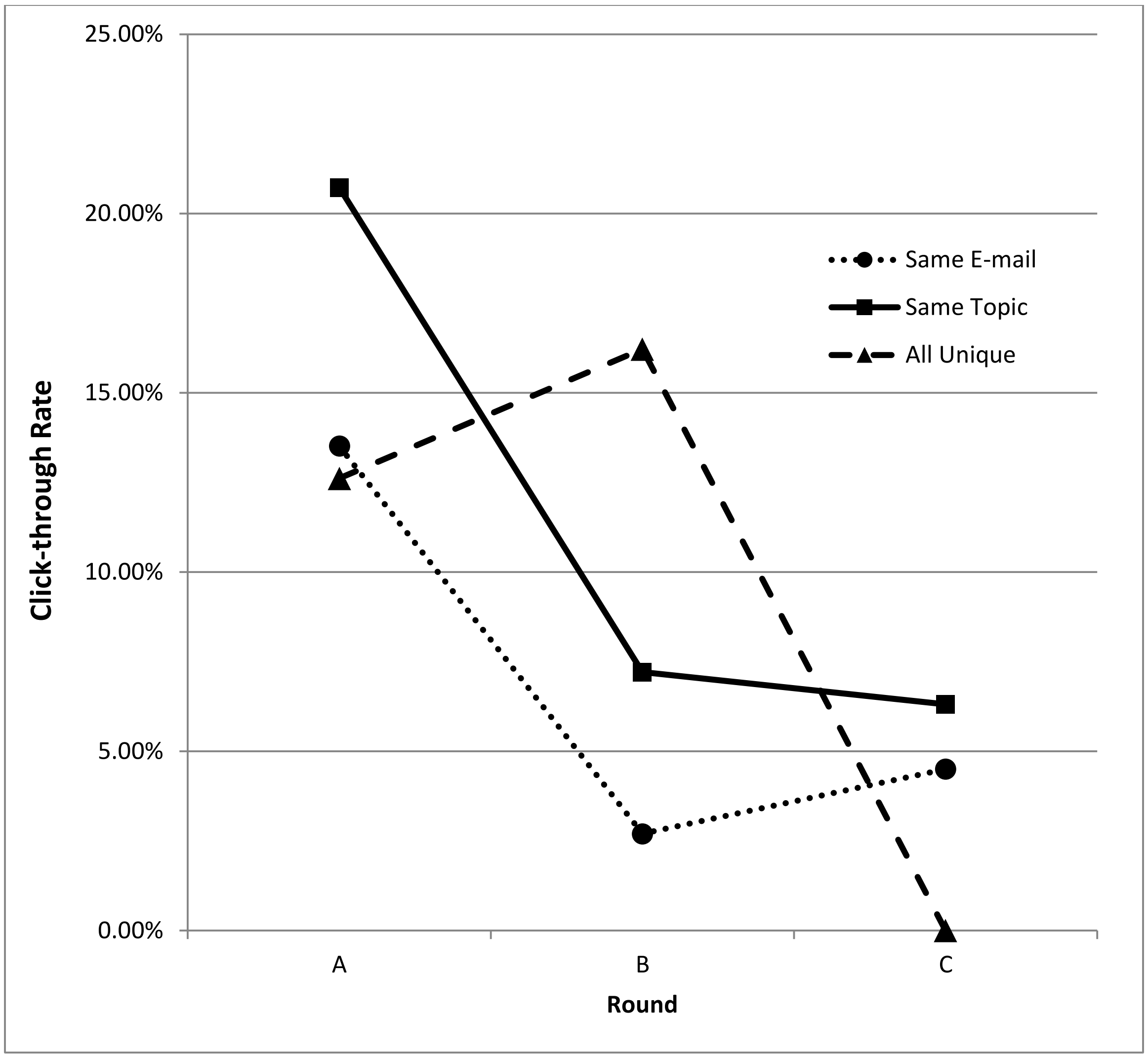


Figure 9 Click-through rates for three rounds of the Content Reuse Study showing potentially decreased utility

6.4 Conclusions

The results of this study were quite interesting, although not as conclusive as we would have hoped. Our initial hypothesis was partially validated, in that there was a significant decrease in click-through rates for the "Same E-mail" cohort after round A. We also detected a significant decrease for the "Same Topic" cohort, which received different e-mails, indicating that repeating the same content or general subject matter may result in decreased utility over multiple exercises when applied to the same test population. So while this could validate that reusing content is not as effective as utilizing new content, we also concluded that reusing the same topic for the e-mail may have the same effect.

While our hypothesis was initially confirmed with the “Same E-mail” and “Same Topic” cohorts, we must concede that there was not enough data to perform a conclusive trend analysis. That being said, the results from the “All Unique” cohort provides additional confirmation that content has an impact on click-through rates, as in the previous motivational study.

CHAPTER 7:
IMPACT OF CONTENT COMPLEXITY AND TRAINING ON RESPONSES

Our initial motivational studies demonstrated that content was a relevant factor in the success of phishing exercises, and that e-mails may experience decreased utility as they are used multiple times. However, given the small sample sizes in both studies, we determined that a much larger study would provide more reliable data. Additionally, we would be able to investigate whether training had any impact on responses rates, which could influence analysis and experimental design for future exercises. To differentiate this study from our motivational studies, we used a broader set of e-mails that all contained relevant content, and included the entire population of a medium-sized organization as our test population. We also provided a training component for participants if they clicked on a link.

7.1 Hypothesis

Our hypothesis was that differences in semantic contents of e-mails will result in significant differences when measuring click-through rates, as previously indicated by our earlier studies. Additionally, we hypothesized that a decrease in click-through rates for the population would be observed over the course of multiple exercises. This would be due to more susceptible users receiving training in earlier rounds of testing, and the test population in general becoming more aware of phishing attacks after additional rounds of testing.

7.2 Experimental Design

In order to run unannounced phishing exercises, we first requested permission from the system owners of a medium-sized organization interested in training their user population. Participants were then selected from an initial list of 23,062 e-mail addresses provided by the organization. The original list was pruned down to 19,180 individual addresses after removing addresses that were either invalid, belonged to accounts that were no longer active, or were specifically requested for exclusion by the organization. For example, some addresses belonged to distribution lists, which would be problematic for tracking individual responses. Similarly, some addresses belonged to incident response staff that were involved in coordinating the exercises, and those individuals would already be aware of the e-mail contents and deployment schedule.

The 19,180 participant addresses were then divided into 32 different test groups. The general construct of these groups was based on feedback from the organization so that deployments could be scheduled in order to minimize business impact, and also so that results could be tracked and reported based on a specific test group if necessary. Some groups were organized by business unit, while others were organized by job title or other internal categories. Each group of exercise participants received 6 e-mails over the course of several months, delivered to their normal work e-mail address. Table 5 lists the final size of each group.

The e-mails were developed manually by subject matter experts to mimic common types of attacks that had been detected in the wild. These attacks ranged from simple reward promises such as a free cruise, to more complex content involving bogus receipts for online transactions. Due to the fact that some of the simulated e-mails contained contents that would be flagged by spam filters and other detection measures, all e-mails were white-listed to ensure delivery. The justification for this approach was that the study was not intended to be a detection or incident response exercise, rather the goal was to measure user responses to various types of e-mails and measure training effectiveness. By ensuring that e-mails were delivered, our measurements reflected participant decisions rather than detection capability effectiveness.

In the event a participant clicked on a link, they were directed to a login page for a training element that was deployed by the organization. Training was provided using STAR*Phish, a commercial tool developed by Booz Allen Hamilton. The tool utilized an interactive training interface that walked participants through a process of identifying suspicious elements in a phishing e-mail. The training was only provided to users that clicked on links in the simulated phishing attacks, and the e-mail that they were tricked by was used as an example inside the training. Exercise participants that did not fall for the simulated e-mail were not sent through the interactive training. The system utilized a tagging approach within the deployed e-mails, so that each e-mail contained unique links. This allowed us to accurately measure the number of unique participants that responded, even if the recipients responded multiple times to a single e-mail.

.The focus of our analysis was on the click-through values for each round. Exercises that track click-through percentages can be distinguished from exercises that only track submitted credentials or number of accessed attachments, because the click-through percentage also indicates users that could have been targeted with redirection attacks or drive-by malicious code attacks. Essentially, once a user clicks on a link in a phishing e-mail, they send a request to the attacker's web server, at which point they might be redirected by the attacker to other malicious websites. For our exercises, the action of clicking a link resulted in a redirection to the STAR*Phish login page. In order to analyze recidivism rates, we also tracked the number of participants that received training for each round after clicking a link. Finally, we coordinated with the organizations incident response team to track the number of participants that correctly reported the test e-mails.

Table 5 Cohort sizes for the Complexity and Training Study

Cohort	*Population Size*
1	937
2	495
3	431
4	391
5	870
6	915
7	680
8	618
9	888
10	929
11	472
12	423
13	368
14	942
17	893
18	927
19	233
20	206
21	157
22	68
23	269
24	435
25	901
26	267
27	916
28	934
29	949
30	286

15	694	31	937
16	594	32	155

7.3 Results

Table 6 lists the detected percentage of individuals who clicked on links for each test population over the course of the 6 exercises, and Table 7 lists the percentage of participants who received training in each round. For example, in round 1, 71 out of 937 participants clicked the link in the test phishing e-mail, as indicated by a 7.6% click-through value. However, just because a participant was redirected to the training did not mean they logged in or completed the training. For example, often times an individual will immediately close the browser once they realize they clicked on a potentially harmful link. Of the 71 participants that were redirected to the training for group 1 in exercise A because they clicked a link, only 18 logged in or completed the training, as indicated by the 25.4% training percentage in Table 7.

Table 8 shows the combined click-through values for each type of e-mail used in each exercise. For example, if an e-mail was sent to 3 test populations in round A, we combined the click-through values for those three groups to get the combined click-through value for that e-mail in that round. For example, in round A 2,112 participants received the Bazaar e-mail and 146 participants clicked the link, resulting in a combined click-through value of 6.9% for that e-mail in round A.

Table 6 Percentage of participants from 32 test populations in the Complexity and Training Study who clicked on links in phishing e-mails across 6 rounds of testing

	Exercise Round					
Group	**A**	**B**	**C**	**D**	**E**	**F**
1	7.6	6.3	4	3	1.8	2.7
2	3.2	2.4	12.3	0.2	1.2	9.1
3	8.1	4.2	1.9	0.5	0	0
4	5.4	3.1	0.5	0.8	15.9	0.8
5	0.5	3.5	5.9	0.9	3.3	5.2
6	0.8	26.3	3	3.8	0.1	0.1
7	8.7	2.5	21.9	0.6	1.9	0
8	1	5.5	7	0.2	10.7	14.9
9	11.3	0.1	4.7	0.7	0.5	5.2
10	2.7	9.2	0.4	1	0	19.2
11	5.3	1.9	0	0.6	9.3	0.8
12	1.9	4.3	26.7	12.3	0.7	1.9
13	7.3	1.4	0.5	3	2.2	10.6
14	4.5	7.6	9.6	4.1	15.2	11.5
15	4	7.2	0.1	25.8	0	2
16	3.5	5.9	5.2	1	0.2	4.2
17	27.4	3.1	5.9	4.8	3.3	3.5
18	0.3	8.7	0.1	2.5	2.7	4.4
19	9.9	21.9	2.6	4.3	1.3	0
20	35.9	1.9	2.4	19.4	2.9	11.2
21	6.4	20.4	6.4	28.7	5.7	0
22	0	11.8	11.8	1.5	16.2	17.6
23	0.7	8.6	6.3	0	10.4	6
24	34.5	7.1	20.9	2.5	2.8	3
25	29.6	0.3	2.4	19.5	14	0.1
26	2.6	36.3	18.4	0.8	8.6	6.7
27	10.6	29.6	7.8	16.4	2.3	1
28	3.3	2.5	0	17.8	2.1	12.4
29	8.8	6.8	3.7	1.2	0.1	0.7
30	7	0.3	1.4	0.7	0	2.1
31	0	6.7	0.1	4.3	0.4	4.6
32	0	2.6	0.7	4.5	0	9
Overall	**7.9**	**7.7**	**5.4**	**5.8**	**3.7**	**5.13**

Table 7 Percentage of participants from 32 test populations in the Complexity and Training Study who received training after responding to a phishing e-mail

	Exercise Round					
Group	**A**	**B**	**C**	**D**	**E**	**F**
1	25.4	35.6	70.3	78.6	70.6	80
2	50	33.3	80.3	0	83.3	75.6
3	42.9	22.2	100	100	n/a	n/a
4	33.3	25	50	100	82.3	66.7
5	50	40	78.4	62.5	65.5	75.6
6	0	36.1	77.8	97.1	100	100
7	33.9	17.6	84.6	50	92.3	n/a
8	50	35.3	90.7	100	72.7	77.2
9	40	0	85.7	83.3	50	76.1
10	28	31.8	100	55.6	n/a	82
11	48	44.4	n/a	66.7	75	100
12	50	50	81.4	75	66.7	75
13	40.7	40	100	81.8	100	89.7
14	40.5	34.7	84.4	76.9	79	78.7
15	35.7	30	0	83.2	n/a	92.9
16	23.8	28.6	77.4	66.7	100	80
17	33.1	21.4	79.2	88.4	86.2	58.1
18	0	37	100	95.7	88	75.6
19	43.5	27.5	50	90	100	n/a
20	37.8	25	80	90	83.3	69.6
21	30	31.3	70	80	88.9	n/a
22	n/a	62.5	50	100	45.5	75
23	50	17.4	82.4	n/a	92.9	81.3
24	32.7	35.5	67	81.8	75	84.6
25	31.1	66.7	90.9	83	77.8	100
26	42.9	34	83.7	100	87	88.9
27	37.1	28.4	70.4	75.3	66.7	77.8
28	58.1	43.5	n/a	84.9	75	78.4
29	36.1	38.5	82.9	54.5	100	71.4
30	20	0	100	100	n/a	50
31	n/a	22.2	100	85	75	83.7
32	n/a	50	100	57.1	n/a	92.9
Overall	**34.8**	**32.5**	**79.8**	**81.8**	**78.5**	**78.9**

Table 8 Combined click-through values for each test e-mail across all test populations for 6 rounds of exercises

	Exercise Round					
E-Mail	**A**	**B**	**C**	**D**	**E**	**F**
AGCE	n/a	7.5	n/a	n/a	2.8	5.5
Bank 1	n/a	n/a	n/a	0.8	2.4	n/a
Bank 2	0.8	0.2	0.5	n/a	n/a	n/a
Bazaar	6.9	n/a	3.8	n/a	2.2	n/a
Big Box	n/a	n/a	n/a	0.8	n/a	n/a
Celebrity	0	n/a	0.1	n/a	0.1	n/a
Cellular	n/a	9	n/a	n/a	2.7	1.1
Certify	3	1.6	n/a	n/a	0.7	0.8
Charity	n/a	n/a	n/a	2.8	1.6	0.7
Complaints	26.3	n/a	17.9	17.8	12.8	3.5
Domain	5.3	n/a	7.8	n/a	n/a	n/a
Dragon	0.7	0.3	0.4	n/a	0.1	n/a
Fax 1	n/a	27.1	n/a	17.2	n/a	1.6
Fax 2	n/a	26.3	22.5	19.4	n/a	15.6
Federal	n/a	5.7	n/a	4	n/a	n/a
Funds	9.9	4.9	5.1	n/a	n/a	n/a
IQ Test	n/a	6.7	6.4	n/a	10.4	n/a
Malware	3.5	2.9	n/a	n/a	n/a	n/a
NACA	3.6	3.1	2.5	n/a	1.8	n/a
Newsletter	n/a	n/a	0.2	n/a	0.4	0.1
Order	28.5	36.3	n/a	n/a	n/a	10.7
Outfitters	n/a	n/a	n/a	4	3.2	3.2
Password	10	7.3	n/a	n/a	2.8	n/a
Secure	n/a	6.3	n/a	2.5	n/a	4.9
Shipping	n/a	n/a	n/a	21.2	13.2	9
Sports	0.3	n/a	0	0.1	0	n/a
Tax	9	n/a	n/a	n/a	n/a	3.2
Warehouse	n/a	2.5	2.6	n/a	n/a	0.1

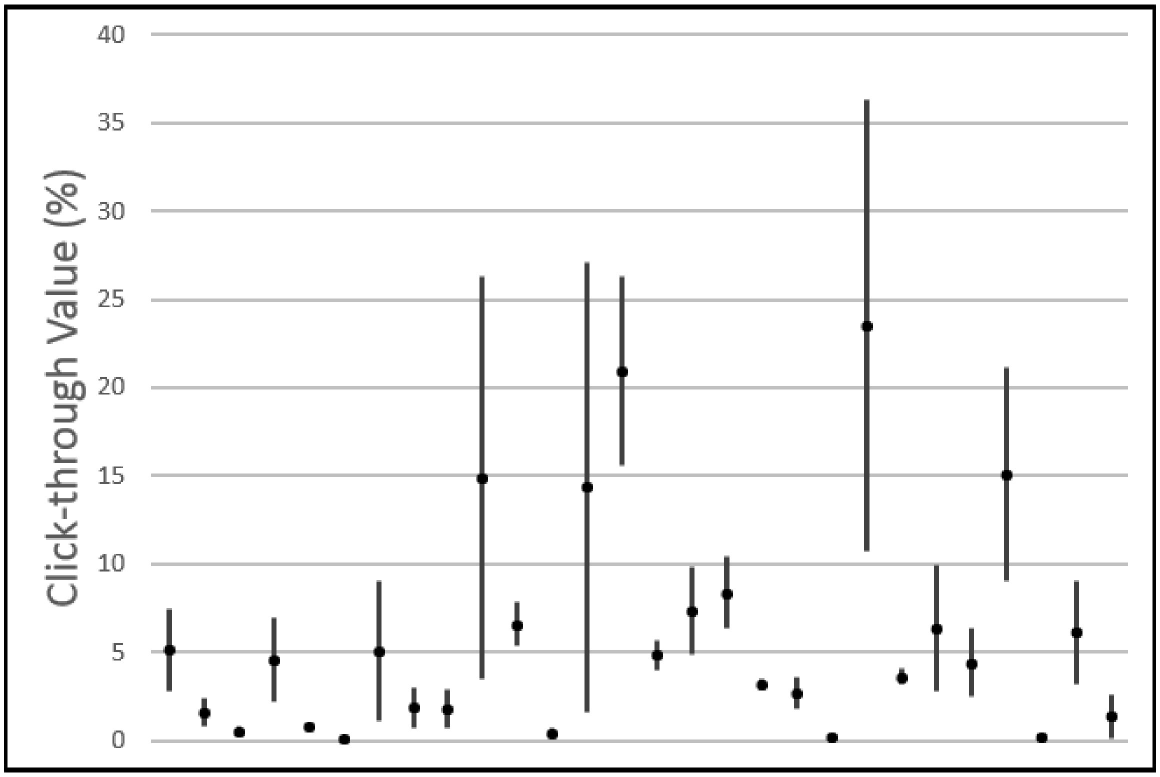


Figure 10 Click-through value ranges for the 28 e-mails used in the Complexity and Training Study

7.4 Analysis

For our analysis of the data, we focused on several areas to provide reliable results and determine whether our initial hypothesis was correct. First, we analyzed the raw click-through value results to illustrate how analyzing non-normalized data may result in misleading conclusions, and illustrate how a selection bias may exist during exercise design due to differences in e-mail complexity. We then used an approach to normalize the data to remove e-mail complexity as a variable, in order to measure e-mail effectiveness across groups that received different e-mails. Finally, we analyzed recidivism rates to determine whether training had a measurable impact on participant responses.

7.4.1 E-mail Complexity

The total click-through percentage available in Table 6 is often what organizations are most interested in, as it presumably indicates the overall susceptibility to phishing attacks. Typically, an organization will view decreasing click-through values over multiple exercises as an indicator of training effectiveness. However, this approach fails to take into consideration differences in e-mail complexity, and depending on the number of exercise iterations may not provide sufficient data for trend analysis.

As an example of how using the raw click-through value can be misleading, our data shows multiple instances of increasing click-through values over the course of multiple exercises when looking at individual cohorts. For example, Group 20 started out with 35.9% of participants responding in exercise A, dropped to 1.9% and 2.4% respectively in exercise B and C, but then jumped back up to 19.4% in exercise D. In fact, of the 32 test populations, only Group 3 experienced a consistent decrease in click-through values across all exercises. While there may have been a downward trend in some other groups, this did not appear to be consistent across all test populations. Even when the combined click-through value across all test groups is calculated, there is an increased click-through value between exercise round E and F. This indicates that the click-through rate is tied to the complexity of the e-mail, and does not necessarily reflect a performance improvement on its own.

Similarly, small populations or limited iterations of testing can result in data that can be used to support an interesting narrative, but ultimately the data is subject to the same issues of content complexity. With only two rounds of exercises are considered, the response metrics from group 20 could easily be used to support an argument in favor of the impact of training or some other improvement metric. However, with the inclusion of the data from our third and fourth rounds of testing, it becomes evident that this is purely coincidental. We can also show this is coincidental by including other participant groups in our results analysis. This is because the raw data is not scaled to e-mail complexity or exercise difficulty, and different groups are receiving e-mails with varying levels of content complexity in different rounds of testing.

When the click-through percentages of each population are compared with the e-mail deployment schedule, a correlation can be made between instances of higher click-through rates and certain e-mails, even across multiple test populations and exercise rounds. This is the same sort of response metrics we saw in our initial motivational studies as well, although here we see that this phenomenon holds across different sample populations as well. For example, relatively high click-through values in exercise round A for groups 17 and 25 were both associated with the same e-mail. That e-mail also resulted in high click through rates for groups 2, 13, 14, and 20 in exercise round F. This illustrates how an e-mail is capable of generating significant responses even after 5 previous rounds of exercises. Very low click-through values could also be attributed to certain e-mails across all test populations as well, regardless of exercise round.

To illustrate how selection bias might influence analysis, we calculated the combined click-through value for each e-mail, for each exercise round that it was used in. The results for each e-mail are illustrated in Table 8. For example, if an e-mail was sent to 3 test populations in round A, we combined the responses for those three groups to get the combined click-through value for that e-mail. In some cases, an e-mail was not used for a particular round of testing, so there was no combined click-through value for that round. Figure 10 illustrates the ranges of combined click-through values for all e-mails that were used in the study, using the data in Table 8. We can see that certain e-mails clearly outperform others with regard to click-through values, and that certain e-mails can be more effective regardless of exercise round and previous training efforts. If we compare the Sports e-mail and the Complaints e-mail, we see that the former never received more than a 0.3% click-through value, while the latter has click-through values between 3.5% and 26% depending on the round it was used in. The data also shows that some e-mails have a large variance in response metrics, such as the Complaints e-mail, which could mean that these e-mail are less effective against certain populations, or that some other variable affected the click-through values in different rounds of testing.

The differences in how well certain e-mails perform means that there can be a selection bias during exercise design, specifically regarding the order of e-mails. If an organization is using overall click-through values as an effectiveness measure, but the e-mails used for exercises decrease in complexity, then the observed click-through values might be misinterpreted as an improvement in participant awareness, when in fact it is likely due to the fact that the e-mails being sent were not as "difficult" in subsequent rounds of testing.

7.4.2 Normalized Gain

A decreased click-through rate may be misinterpreted as an improvement due to training, when in fact it could simply be that a very weak e-mail was used for that round of testing. Exercise difficulty, which we determined by the strength or quality of the e-mails used, should be taken into consideration when analyzing results. In this section we propose an approach to normalizing the response metrics according to exercise difficulty so that a fair determination can be made regarding improvement based on e-mail effectiveness. For example, we don't want to over inflate the impact of “hard” or complex e-mails, nor trivialize the impact of “easy” or less complex e-mails. In the end, the data must be normalized in order to provide reliable results.

Measuring how well a participant responds to a particular phishing e-mail is similar to measuring how a student might respond during an exam. The response is influence by the preparation of the student, but also by the difficulty of the exam itself. Normalized gains have been used to analyze this type of response for decades [80][81]. For example, Hake et al. proposed that normalized gain could be used as a meaningful measure of how well a course teaches ideas to students. In one of his studies, he defined the normalized gain as the average increase in students' scores on a particular test divided by the average increase that would have resulted had all students had perfect scores. In the case of Hake's study, the normalized gain was used to measure the effectiveness of an academic course in promoting conceptual understanding of physics concepts.

We defined normalized gain in the context of our study as the ratio of observed click-through values to the maximum expected click-through value for a given round of testing. For each e-mail, we took the largest observed click-through value from any round of testing according to Table 6 and defined that as the optimal expected click-through value for a specific e-mail. We then calculated the number of expected clicks for every round by multiplying each sample size by the expected click-through value for the e-mail received in that round. This value was our equivalent of the perfect score metric employed by Hake, and represented the number of clicks that would be detected if the population responded under ideal conditions. Finally, we calculated the normalized gain (G) for each round of testing with the equation in Equation 1

Equation 1 Formula for measuring normalized gain

$$G = \frac{(\text{Clicks}_{\text{observed}})}{(\text{Clicks}_{\text{optimal}})}$$

In this equation $\text{Clicks}_{\text{observed}}$ is the total number of unique links clicked from all groups, and $\text{Clicks}_{\text{optimal}}$ is the total number of clicks that would be expected had the e-mail performed under optimal conditions. In the context of evaluating a phishing exercise, a lower normalized gain value indicates that the sample population was less vulnerable than expected, because the e-mail was not as effective as it could have been.

As an example, in round A we used the “Order” e-mail for group 25. That e-mail had an optimal expected click-through value of 36.3% according to Table 6, so given the group population size of 901 participants, we would estimate a best-case performance of around 327 clicks for round A from group 25 ($327 ≈ 901 * 0.363$). However, we only observed 267 responses from participants, resulting in an effectiveness rating of 81.7%. We used the “Tax” e-mail for group 27 in round A, so by the same logic we would estimate a best-case performance of 97 clicks ($97 ≈ 916 * 0.106$) of which we observed

97. This meant that the e-mail in round A for group 27 performed at its optimal level or 100%. After doing this for each e-mail in each round of testing, we could then calculate the average efficiency rating across the entire population for each round.

Table 9 lists the normalized gain for each e-mail used in our study, and Figure 11 shows the average normalized gain plotted over the course of 6 exercises compared to the average click-through value.

Table 9 Efficiency rating for 6 rounds of test e-mails in the Complexity and Training Study using normalized gain analysis.

	Exercise Round					
Group	**A**	**B**	**C**	**D**	**E**	**F**
1	87.7	100	33.6	43.8	45.9	59.5
2	38.1	85.7	34.5	14.3	28.6	25.1
3	77.8	66.7	66.7	33.3	0	0
4	45.7	26.1	100	30	55.4	8.6
5	50	78.9	50	66.7	36.7	68.2
6	31.8	98.8	100	61.4	12.5	25
7	100	85	61.1	40	44.8	0
8	120	81	67.2	50	29.7	55.8
9	100	4.8	54.5	26.1	100	14.5
10	78.1	100	44.4	69.2	0	71.8
11	45.5	16.4	0	25	32.6	9.3
12	44.4	38.3	100	41.6	21.4	18.2
13	71.1	38.5	25	68.8	25.8	29.3
14	100	100	81.8	92.9	42.3	31.6
15	100	61.7	100	90.4	0	19.2
16	100	94.6	44.9	75	100	96.2
17	75.6	77.8	57.6	76.8	72.5	33
18	100	77.9	100	59	100	41.8
19	85.2	75	66.7	100	50	0
20	100	57.1	100	72.7	26.1	31.1
21	62.5	69.6	55.6	100	75	0
22	0	100	100	100	45.8	60
23	33.3	95.8	54.8	0	100	88.9

24	96.2	93.9	78.4	40.7	25	68.4
25	81.7	37.5	61.1	66.2	48.8	3.8
26	63.6	100	69	11.1	24.2	90
27	100	100	66.4	57.3	30	28.1
28	83.8	56.1	0	49.6	23.5	46.6
29	78.3	100	42.7	44	100	17.5
30	19.6	16.7	16.7	28.6	0	23.1
31	0	64.9	4.5	95.2	100	67.2
32	0	66.7	33.3	100	0	31.8
Overall	**67.8**	**70.8**	**58.5**	**57.2**	**43.6**	**36.4**

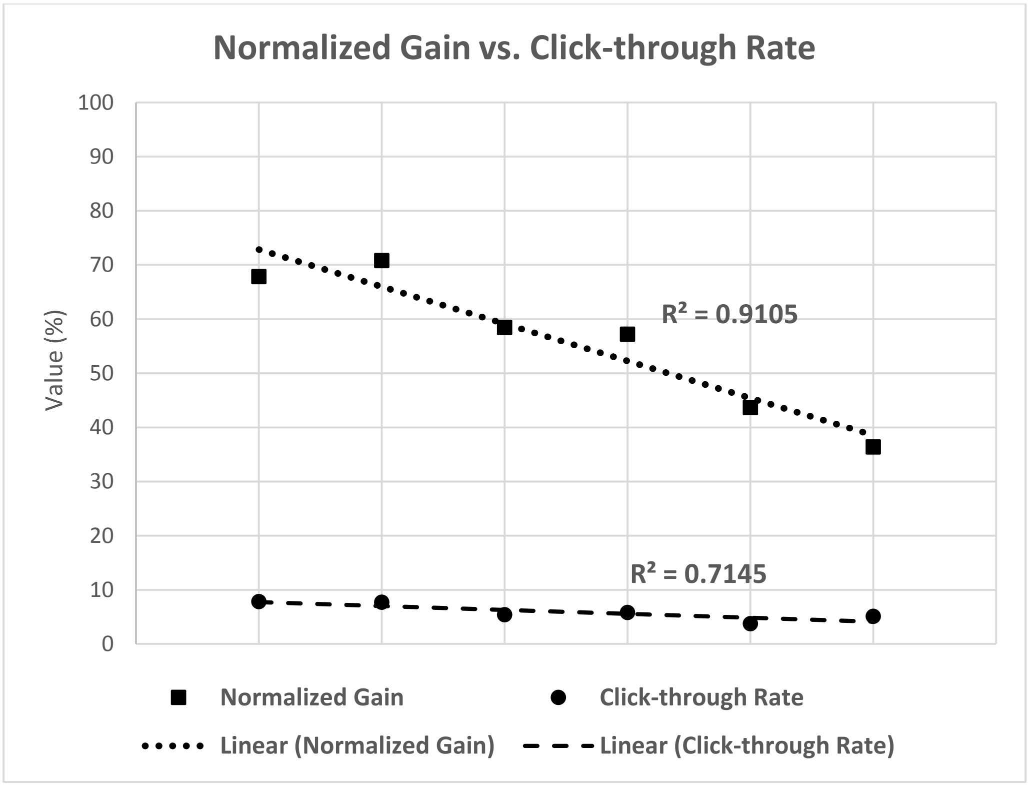


Figure 11 Normalize gain over the course of 6 exercises compared to raw click-through rate, indicating a decrease in the effectiveness of phishing attacks

7.4.2 Recidivism

Aside from determining whether a given population is susceptible to phishing, some organizations are interested in determining whether their training approaches are effective. Presumably, if a particular training methodology is effective, participants will be less likely to fall for a phishing e-mail after receiving the training. In order to measure this, we can look at recidivism rates, or in other words the number of users that responded to additional e-mails after having received training. For this metric, we are primarily interested in repeated responses across multiple exercises. A repeated response in our data is where a participant responds to an e-mail after already having responded to a previous e-mail in a previous round. Each additional time the participant responds was considered one repeated response instance.

While our normalization approach is applied to the overall click-through values for the various rounds of testing, recidivism deals with individual participants and their repeated responses between different rounds of testing. As demonstrated earlier, though, e-mail complexity can have a significant impact on click-through rates, and so we must take this into consideration before relying on repeated response data as any sort of indicator of training effectiveness. Some users, will inevitably receive "easier" e-mails after having fallen victim to a "harder" e-mail, and as such might be less likely to have responded regardless of training. To determine whether recidivism in our study was due to increased e-mail complexity, we did an analysis of the order in which users fell for e-mails, and tracked whether they had received training at any point in the process as well. In order to measure increased or decreased complexity for the e-mails between rounds, we used the highest click-through value received for a particular e-mail in any round and any population, as well as the click-through value received for the previous e-mail in that particular test population.

As an example, in one case a participant responded to the “Malware” e-mail in round B, received training, and then also fell for the “Complaints” e-mail in round D. For that particular test population, the click-through value for round B was 2.5%, while the click-through value for round D was 17.8%. Additionally, the maximum click-through value received by the “Malware” e-mail in any population was 4.5%, while the maximum click-through value for the “Complaints” e-mail was 35.9%. This would indicate that the “Malware” e-mail was less effective than the “Complaints” e-mail for that test population, and likewise less successful when looking at all test populations, and so we considered this an increase in difficulty for this particular recidivism instance. However, the same user then fell for the “Fax 2” e-mail in round F. The click-through value for round F was 26.7%, and the max click-through value for the Fax 2 e-mail was 26.7%. In this case, then, we considered this a decrease in e-mail complexity for this recidivism instance, as the “Fax 2” e-mail was considered less effective than the “Complaints” e-mail. For this participant, then, we registered two instances of repeated responses.

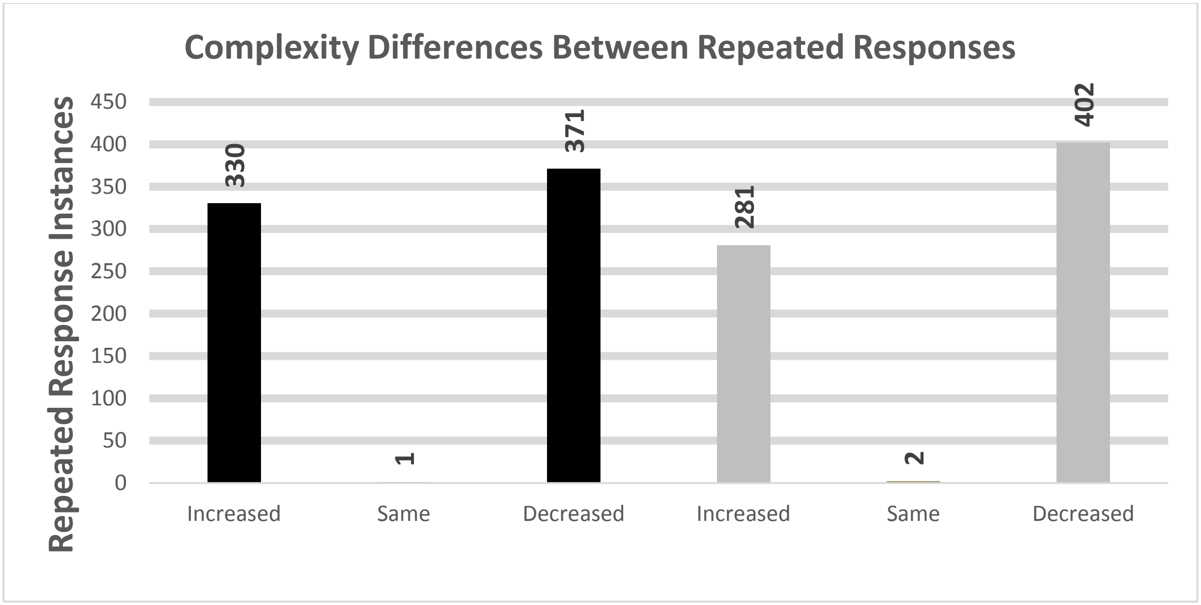


Figure 12 Instances of repeated responses in the Complexity and Training Study where e-mail difficulty increased, stayed the same, or decreased

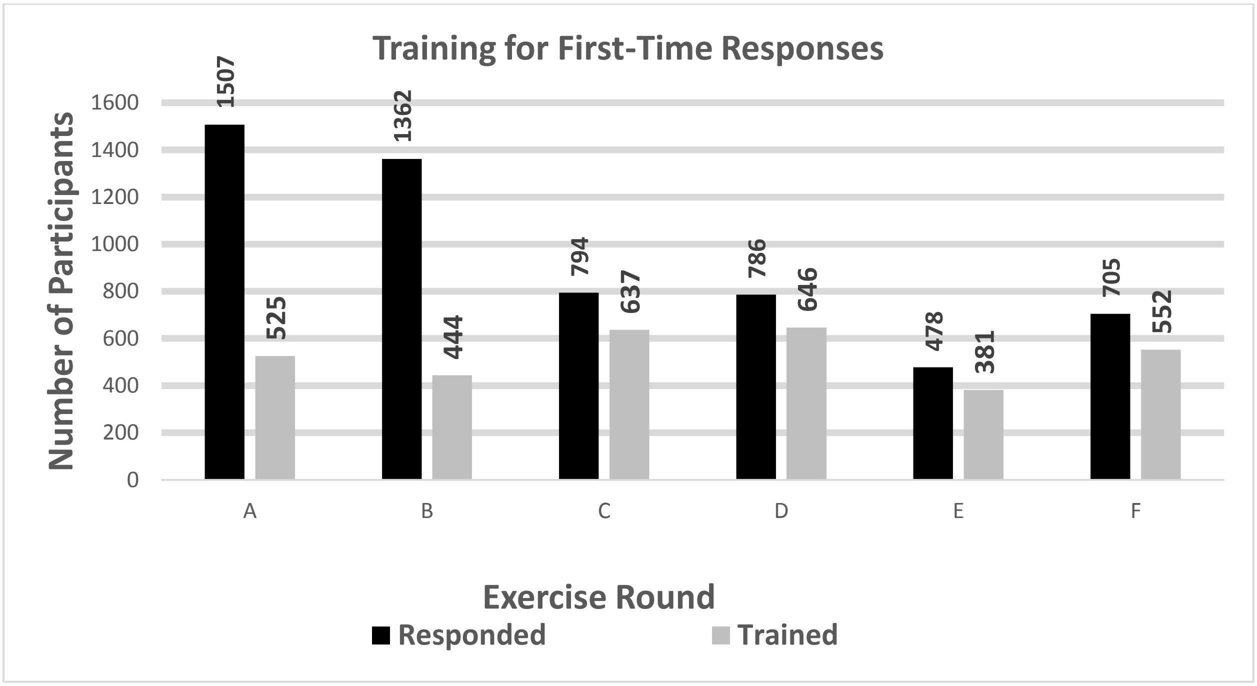


Figure 13 Number of trained participants in the Complexity and Training Study after first-time responses across 6 rounds

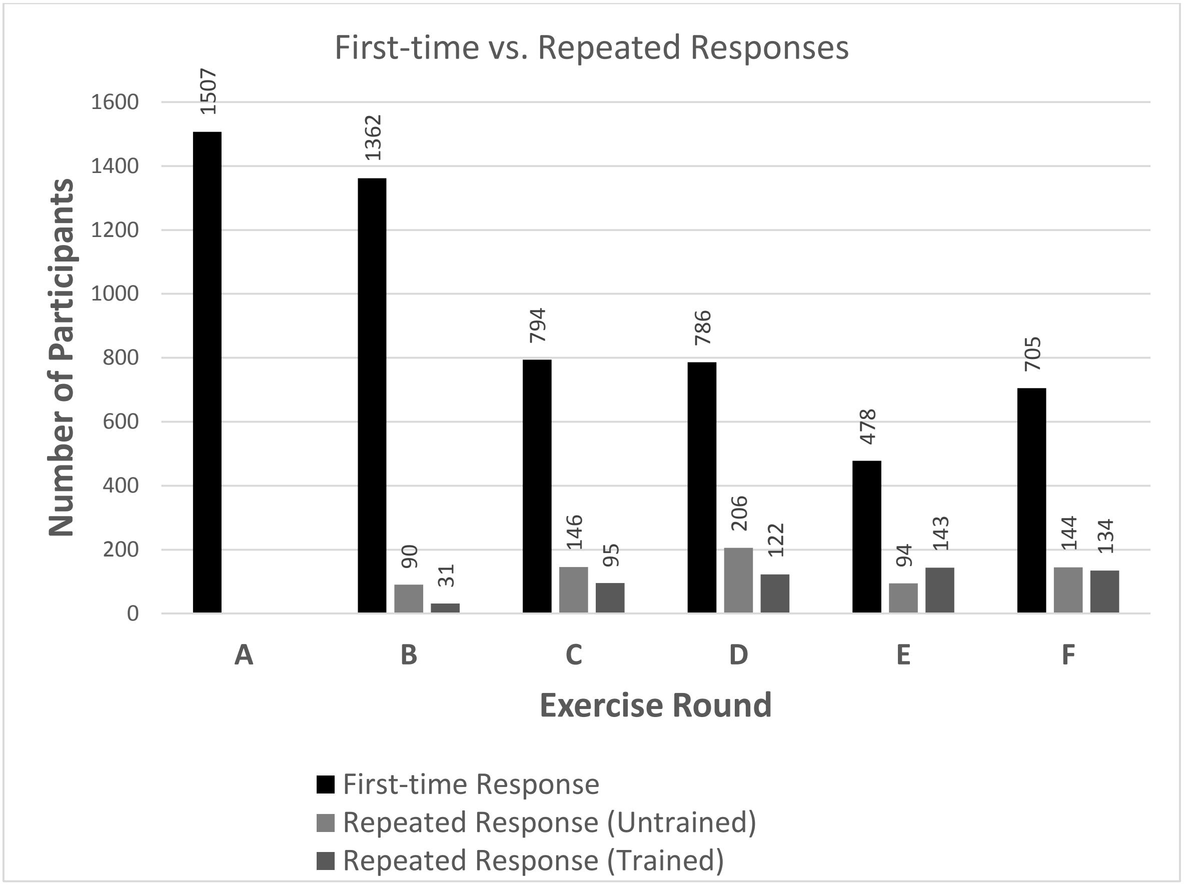


Figure 14 First-time responses and repeated responses for trained and untrained participants in the Complexity and Training Study across 6 rounds

Figure 12 illustrates a breakdown of the number of repeated response instances in which e-mail complexity increased, stayed the same, or decreased using the maximum click-through value for comparison. It should be noted that using our other approach, which compared the click-through values only within that test population, yielded similar results, but are omitted to conserve space. Additionally, these repeated response instances are broken out according to whether the participant had received training prior to the failure. We see that the number of instances of decreased e-mail complexity are actually higher for both trained and untrained participants. As such, the recidivism data is not simply the result of increasing e-mail complexity.

Figure 13 illustrates the number of participants in our study that received training after their first response to a test e-mail, while Figure 14 illustrates the number of first-time responses vs. repeated responses across the six exercises. The repeated failures in Figure 14 are broken out into participants who received training and those that did not. In exercise A, there were no repeated failures because this was the first exercise, but only 525 out of 1,507 participants received training. For exercise B, however, there were 121 users who had already fallen for the e-mail in exercise A and also responded to the e-mail in exercise B. Of these, 90 had ignored the training, while 31 had accessed the training. Similarly, there were 241 users who had responded either to the e-mail in exercise A or the e-mail in exercise B, as well as responding to the e-mail in exercise C. Of those, 145 had ignored training, while 95 had previously received training in exercise A or in exercise B.

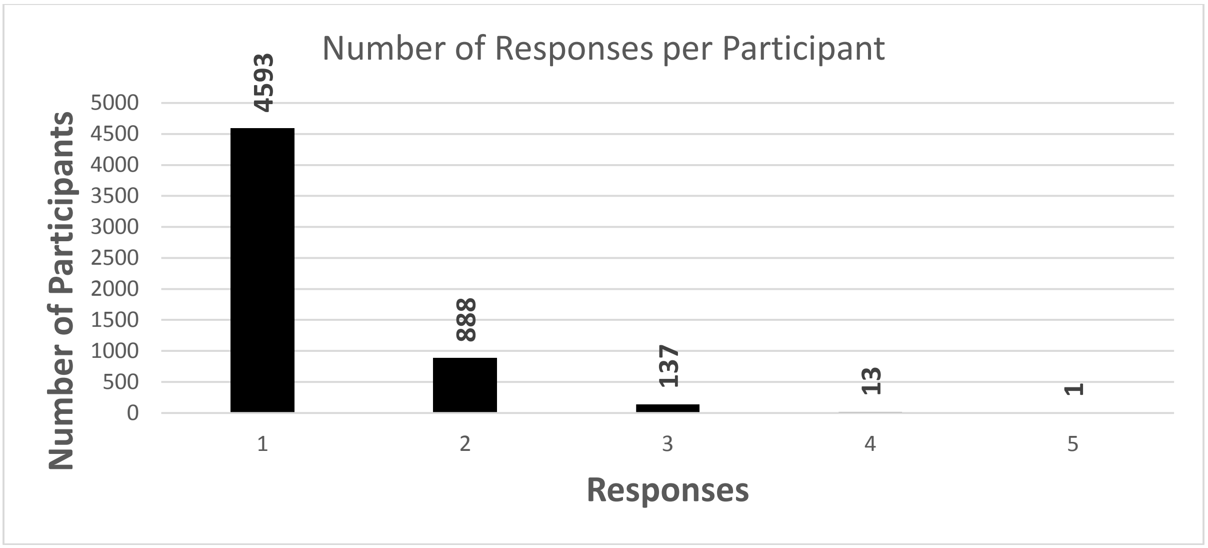


Figure 15 Total number of responses per participant in the Complexity and Training Study

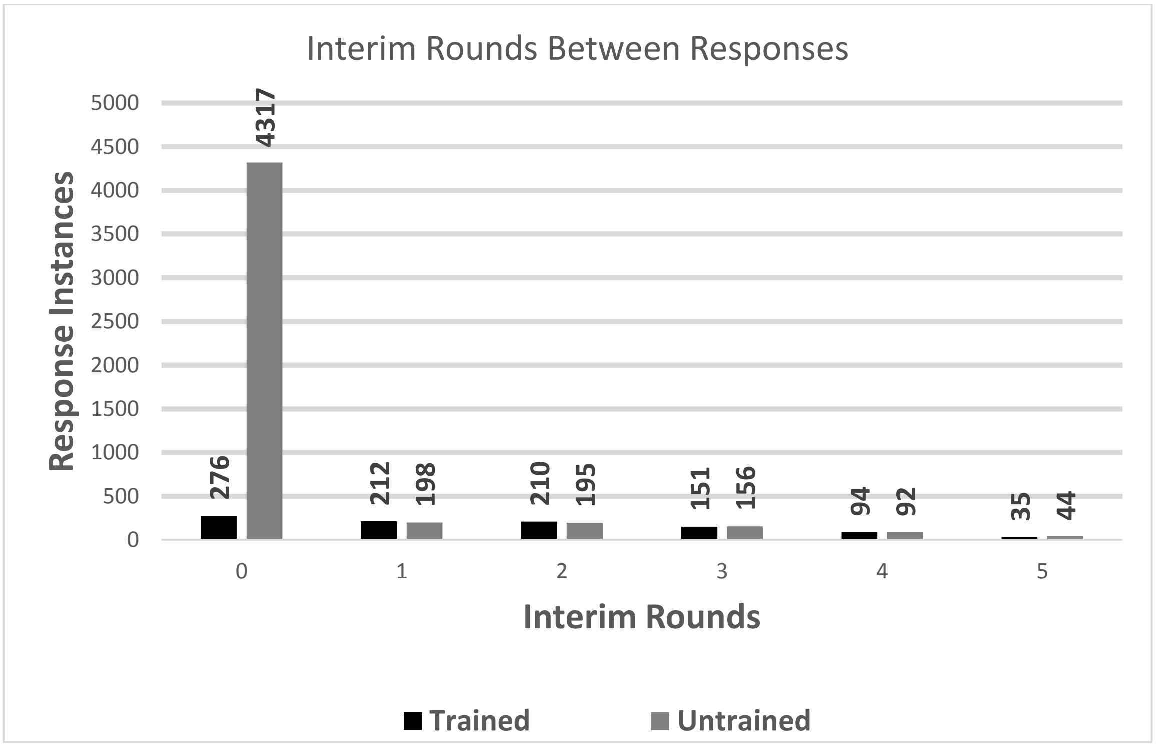


Figure 16 Interim rounds of testing between responses in the Complexity and Training Study

After looking at individual responses from each participant for each exercise, we observed that some users had multiple repeated responses, which means the trained and untrained populations in Figure 14 sometimes included the same participant across multiple exercises. To determine whether this was a significant factor in analysis, we looked at the overall number of responses per participant, as illustrated in Figure 15. We see that there were 1,039 participants that responded in multiple rounds of testing, compared to 4,593 that only responded in a single round and had no instances of repeated responses. In the group of participants with multiple responses, only 151 responded to 3 or more exercise e-mails, with one individual responding to 5 exercises. In total then, out of the 19,180 participants, 29.4% responded to at least one test phishing e-mail. However, only 5.4% of the population responded to multiple phishing e-mails, compared to 23.9% who only had a single response. Only 0.8% of the population responded to 3 or more exercises.

We looked at the number of interim rounds of testing between repeated responses to determine whether the amount of time between failed exercises had an impact. Again, we took into account whether or not the participant had received training prior to each instance of a repeated response as well. Figure 16 illustrates the number of interim rounds of testing between repeat responses. No interim rounds indicates a first-time responses, while a single interim round means the response occurred immediately after responding to the previous round of testing. Apart from the first-time responses, the number of repeated response instances with various numbers of interim rounds was virtually identical between trained and untrained participants. For example, of the number of repeated response instances that occurred after having not responded to a single exercise, 212 were attributed to participants that had received training, while 198 were attributed to participants that had not received training.

Finally, to determine whether training contents had a significant impact, we performed a log-rank test using data from the various rounds of testing. The log rank test is used to test the null hypothesis that there is no difference between the populations in the probability of an event, which for our purposes was a repeated response after either receiving or ignoring training, at any time point. The analysis is based on the times of events, which we mapped to each round of testing. For each such time we calculate the observed number of repeat responses in each group and the number expected if there were in reality no difference between the groups. Since we did not know which users would respond prior to testing, and thus could not assign participants to groups prior to testing, we assigned participants to groups based on the round where we first detected responses, and then used all successive rounds of testing for analysis. We then tested the null hypothesis that there was no difference between the trained and untrained groups generated from that round of testing.

For example, we first looked at the responses in Round A, and found all the first time responses from that round. Of those first-time responses, Group A contained all the participants that responded to round A and received training, while group B contained all participants that ignored training. We then performed a log-rank analysis for exercise rounds B through F for those participants, and removed a user from the analysis once they fell for a second e-mail. We did the same thing for all first responses in round B and performed a log-rank analysis across rounds C through F.

H_0 for our analysis was that there is no difference between trained and untrained users with regards to recidivism. The χ^2 critical value for analysis from groups determined by first-time responses in Round A is 0.19 + 0.10 = 0.29 with 1 degree of freedom resulting in a cumulative probability $P(\chi^2 <= CV)$ of 0.41. The χ^2 critical value for analysis from groups determined in Round B is 0.35 + 0.17 = 0.52 with 1 degree of freedom resulting in a cumulative probability $P(\chi^2 <= CV)$ of 0.53. The χ^2 critical value for analysis from groups determined in Round C is 0.03 + 0.12 = 0.15 with 1 degree of freedom resulting in a cumulative probability $P(\chi^2 <= CV)$ of 0.3. The χ^2 critical value for analysis from groups determined in Round D is 0.01 + 0.05 = 0.06 with 1 degree of freedom resulting in a cumulative probability $P(\chi^2 <= CV)$ of 0.19. Finally, the χ^2 critical value for analysis from groups determined in Round E is 0.37 + 1.36 = 1.73 with 1 degree of freedom resulting in a cumulative probability $P(\chi^2 <= CV)$ of 0.81. First responses from Round F were not used, because there were no additional exercises to compare the groups against. With none of these can we reject H_0 at more than 90% confidence. The closest we get is in the last segment, which only tests one additional round of testing (groups derived from Round E first-responders, analysis performed in Round F).

7.5 Conclusions

As demonstrated by the results of this study, our original hypothesis was confirmed: e-mail content has a significant impact on the results of phishing exercises. While this may seem intuitive, due to the fact that not all phishing attacks have the same success rate, we determined that it is an important variable to consider when planning exercises and interpreting results. In all three studies different types of e-mails were shown to cause significant differences in click-through values, even across multiple exercises.

By calculating normalized gains instead of relying on raw click-through data, we can also provide a more fair assessment of how the test population improves over time with regards to overall performance, independent of e-mail complexity. In our study, the normalized gain analysis indicates that the effectiveness of phishing e-mails decreased significantly over the course of 6 exercises, from 67% down to 36%, meaning users became less susceptible to phishing attacks over time.

From our analysis of recidivism, we can make several important observations. First, the percentage of users that received training after responding to an e-mail for the first time increased from 34.8% in exercise A to 78.3% in exercise F, as illustrated in Figure 13, which could indicate that the participant population became less suspicious of the training resource over time. While this does not reflect training effectiveness, it may indicate that multiple exercises are a method of popularizing a specific training solution with exercise participants over time. However, while the training percentages for first-time responses increased over 6 rounds of testing, the number of repeat responses per exercise did not increase at a similar rate. In other words, the majority of responses were attributable to participants who only failed a single exercise, which would indicate that some variable influenced whether a participant might respond to multiple e-mails.

When looking at all of the first-time responses, only 56.6% of these participants received training after clicking the link, yet repeat responses were significantly lower across both trained and untrained users. We did not see a significant difference between trained and untrained participants when looking at multiple failures in Figure 14, or when looking at the number of interim rounds between responses in Figure 16. This was also supported by the results of our log-rank test analysis. From this we conclude that the content of the training did not have a significant impact on recidivism, but that informing participants that they fell for a phishing e-mail does have significant impact. This is similar to the conclusions reached by Bliton et al. regarding failure-triggered training approaches, in that the idea of immediately notifying a participant that they performed a potentially dangerous action will increase their level of awareness and decrease the probability of future responses [57].

Our original hypothesis, that e-mail contents and training have a significant impact on participant responses, was confirmed by this study. We conclude then that a content generation methodology would have utility, if it can achieve equivalent responses as manually generated e-mails. However, in testing such an approach, we must also take into account the impact of training on participants in between rounds.

CHAPTER 8:

GENERATING CONSISTENT SEMANTIC CONTENT WITH GRAMMARS

Our initial studies demonstrated that content was a relevant factor in the success of phishing exercises, and that e-mails may experience decreased utility as they are used multiple times. Based on these results, we determined that an automated content generator could provide some benefit if it is able to generate new content that perform as well as e-mails created using current methods. To create such a generator, we utilized PCFGs with additional post-processing for context enforcement and developed a tool called PhishGen.

8.1 Generative Grammars

Generative grammars have been an area of active research for many years, in fact the Standard Theory corresponds to the original model of generative grammars proposed by Noam Chomsky in 1965. Recent research has focused on the use of generative grammars for use in translation systems and linguistic analysis [82][83][84]. However, our goal was not to analyze existing e-mail contents for syntactic structure, but rather to generate new content based on a set of grammatical or structural rules. In translation systems the generative grammar is not used to create new content, there is an existing sentence that is converted to the destination language by applying rules and maintaining syntactic structure. So generative grammars to date have typically been analytical rather than purely generative.

The closest research to what we were interested in developing was SCIGen, an automatic Computer Science paper generator [85]. SCIgen uses a hand-written CFG to create papers for submission to conferences, ideally to illustrate weaknesses in submission standards [86]. An interesting aspect of this tool is that it also creates relevant charts and graphs, something which would also be useful for phishing e-mails. However, the approach taken by SCIgen is to trick submission panels into allowing bogus papers into conferences through use of ambiguous technical language rather than convincing and coherent content. As we saw in our initial studies, gibberish or random content does not generate significant responses in a phishing exercises, but the underlying approach of using context-free grammars had merit.

Early in the evolution of generative grammars, Noam Chomsky described a hierarchy of three types of grammars: regular grammars; CFGs; and transformational grammars [87]. Regular grammars, according to Chomsky, are inadequate to describe human language due to the inability to center-embed strings in the language. Context-free grammars fix this problem, but Chomsky argues that regardless of the ability to center-embed strings, CFGs are still inadequate when it comes to describing human language. However, as an underlying structure for creating an e-mail template, we found CFGs to be effective enough, and also required less experience on the part of the developer in order to generate effective rules.

One of the problems with CFGs is that they can create an infinite set of strings if the rules are not properly designed. By definition, a CFG cannot determine a specific number of times a rule has been applied, and so there must be some external control in place to either validate rules in the grammar, or else place a limitation on the number of times a rule can be applied. While this may seem like a limitation, the tremendous variation and growth afforded by CFGs was in fact a feature we wanted to harness. Rather than creating a single e-mail, which is not very cost-effective and provides limited utility over time, we wanted to be able to create a set of rules that could generate millions of potential e-mails [88][62]. CFGs provide that potential growth.

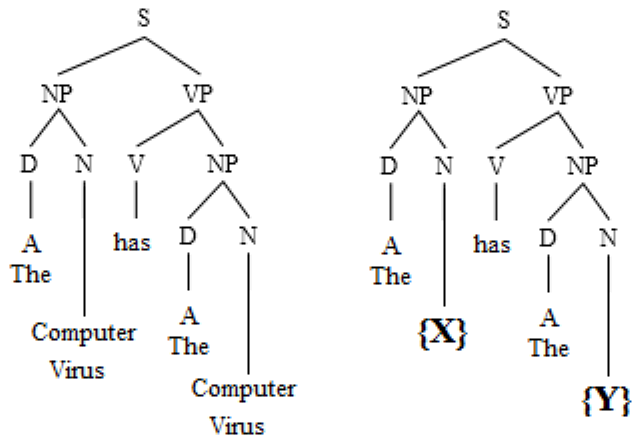


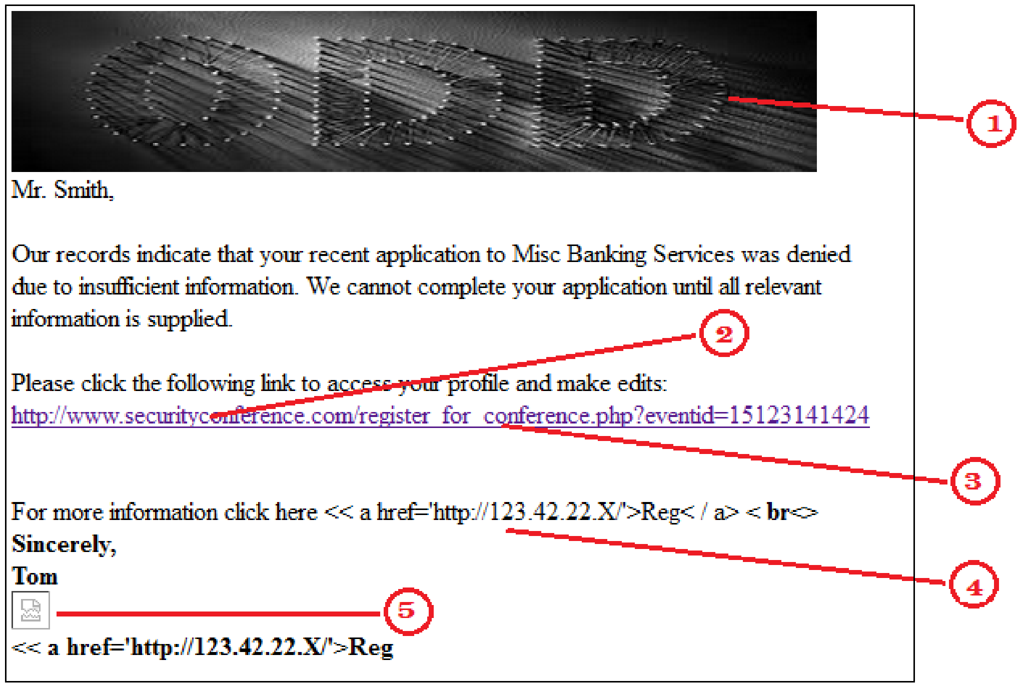
Figure 17 Sample grammar trees showing an example implementation of variable interpolation

By using a CFG, we inherit a serious limitation: maintaining context. For example, when determining which rule to use next, the grammar does not keep track of previously applied rules. Even worse, a grammar is essentially limited to generating syntactically valid strings, and cannot determine rules based on semantics. So, while the phrase ``The virus has a virus'' might be perfectly valid using the simple grammar on the left in Figure 17, it makes very little sense when read by a person. While the application of weights to rule decision may prevent some obvious repetition, in this case preventing the choice of the word ``virus'' twice in the same proximity, this still does not guarantee that there is consistency throughout the entire e-mail. The grammar is not able to determine that computers have viruses, and not the other way around.

Semantic validity is a difficult problem from a generative grammar perspective. Generally speaking it is easy to evaluate a given sentence to determine if it is syntactically valid, and likewise given a grammar it is easy to generate strings that are in the language described by that grammar. The generated string may even make sense on its own, though often times that may simply be coincidental. However as you create longer blocks of text, it becomes far more difficult to create content that is semantically consistent throughout, because the grammar only bases decisions on available rules, and not on overall meaning or context. As an example, when generating a paragraph of text the grammar could start out by creating syntactically valid sentences about apples, but end up creating sentences about elephants towards the end of the paragraph. When read by a user, this paragraph might not make sense, although syntactically it would still be valid according to the grammar. Within phishing e-mails, the goal is to coerce users into clicking a link, and so maintaining context is critical in order to ensure the e-mail makes sense. For example, if the e-mail is attempting to get them to change a password for an account, we don't want to throw in content that describes a prize the user may have won. Generally speaking, we wanted to avoid creating elements that would cause the reader to pause, question the intent or validity of the e-mail, or otherwise reconsider clicking links.

Besides basic text, phishing e-mails also have other elements that need to be consistent within the e-mail. For example, links and images have to be relevant to the content of the message, and links should also be relevant to the e-mail as well. For example, if we have an e-mail that is trying to get a user to click a link in order to change their password, we don’t want to have pictures of elephants (unless elephants have some relevance to the application). Similarly, a link that references registering for a conference, such as “http://www.test.com/registerforconference.cgi”, would also be out of place. Visual presentation is also important, so the underlying structure of the e-mail also has to be consistent.

Figure 18 shows an example of an e-mail with multiple context errors. The image that is included at 1) does not have any relevance to the text. The domain reference at 2) does not reference the company used in the text, and the target page at 3) is clearly not related to the type of activity the e-mail is requesting. Additionally, HTML errors at 4) have caused display issues, and a typographic error at 5) has resulted in additional visual errors including a missing image reference. While unlikely that all of these types of errors would all occur in the same e-mail, any one of them would be enough to raise doubts in the mind of the recipient. Figure 19 shows the same e-mail with context-relevant content.



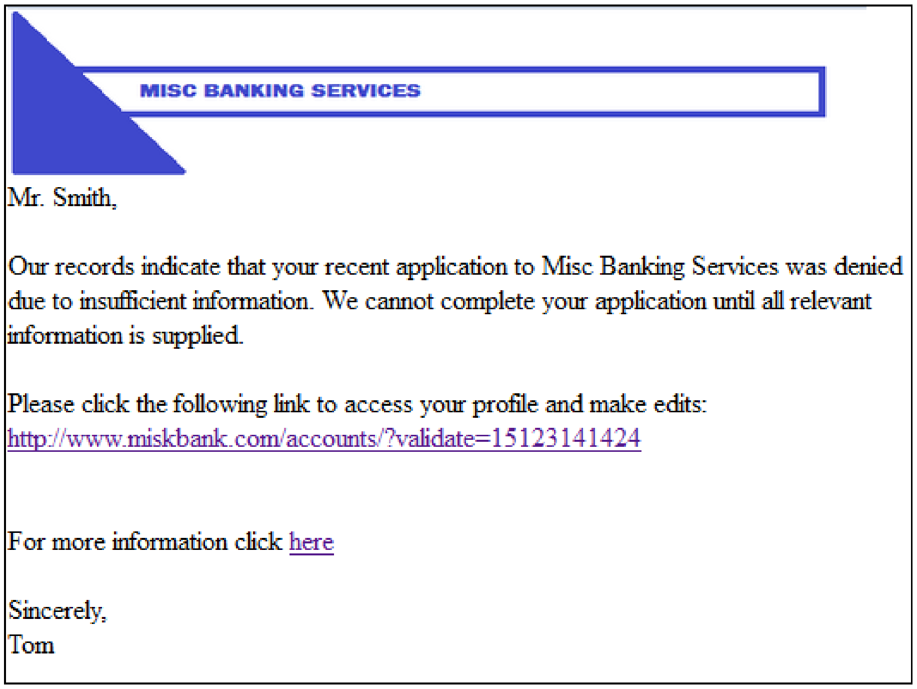
Mr. Smith,

Our records indicate that your recent application to Misc Banking Services was denied due to insufficient information. We cannot complete your application until all relevant information is supplied.

Please click the following link to access your profile and make edits:
http://www.securityconference.com/register_for_conference.php?eventid=15123141424

For more information click here << a href='http://123.42.22.X/'>Reg< / a> < **br<>**
Sincerely,
Tom
<< a href='http://123.42.22.X/'>Reg

Figure 18 Example of a simulated phishing e-mail with obvious context errors



Mr. Smith,

Our records indicate that your recent application to Misc Banking Services was denied due to insufficient information. We cannot complete your application until all relevant information is supplied.

Please click the following link to access your profile and make edits:
http://www.miskbank.com/accounts/?validate=15123141424

For more information click here

Sincerely,
Tom

Figure 19 Example of a simulated phishing e-mail with context-appropriate visuals

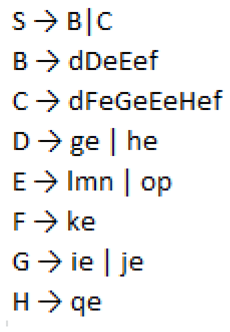
In order to enforce context on generated content, we decided to utilize a form of variable interpolation. Rules within the grammar generate the overall structure of the e-mail with some grammatical elements, however key contextual references are substituted for placeholders. After the template is finalized, the software goes through and finds appropriate values to apply to each variable. In this way, our system can go through and establish the same values in different locations, ensuring that content is consistent, or ensure that values are not repeated. A general example is illustrated in the right in Figure 17, which incorporates variable interpolation into the grammar tree on the left. By differentiating the variables, we can ensure that there is no repetition unless it is intended. We can also define a subset of acceptable values for {X} and for {Y}, and as such ensure that the generated sentence is both syntactically and semantically valid.

8.2 Context-free Grammars in PhishGen

A context-free grammar *G* is defined by the 4-tuple $G = (V, \Sigma, R, S)$ where:

1. V is a finite set; each element $v \in V$ is called *a non-terminal character* or a *variable*. Each variable represents a different type of phrase or clause in the sentence. Variables are also sometimes called syntactic categories. Each variable defines a sub-language of the language defined by G.
2. Σ is a finite set of *terminal*s, disjoint from V, which make up the actual content of the sentence. The set of terminals is the alphabet of the language defined by the grammar G.
3. R is a finite relation from V to $(V \cup \Sigma)^*$, where the asterisk represents the Kleene star operation. The members of R are called the *(rewrite) rule*s or *production*s of the grammar. (also commonly symbolized by a P)
4. S is the start variable (or start symbol), used to represent the whole sentence (or program). It must be an element of V. [90][91]

Figure 20 illustrates the production rules P for an example grammar G = {{S,B,C,D,E,F,G,H}, {d,e,f,g,h,i,j,k,l,m,n,o,p,q}, P,S}, represented in Chomsky normal form. This is the type of structure we decided to emulate in our system, however this CFG illustrates the difficulty in developing a structure for representing production rules for a grammar with such a large alphabet. Instead of using alphanumeric variable naming, we needed to find a different way of encoding the various terminals and variables in our grammar. The main feature we are interested in is the ability to validate termination, such that every variable has a production rule that results in only terminals, so that we can prevent the grammar from generating infinitely long e-mails. However we also need to allow for post-production variable interpolation as we mentioned earlier.



```
S → B|C
B → dDeEef
C → dFeGeEeHef
D → ge | he
E → lmn | op
F → ke
G → ie | je
H → qe
```

Figure 20 Example of a context-free grammar (CFG) in Chomsky normal form

Table 10 is a more robust example of the production rules required for our system, in this case implementing essentially the same grammar as in Figure 20 but with more elaborate variable naming conventions. The terminal character “e” in the sample grammar is instead shown as “{<br>}”. In this way the structure becomes more legible to a developer, and allows us to include more information in the structure. For the purposes of explanation, we call this set of rules a “template”, as it provides the basic production rules for creating a certain type of e-mail. As developers create templates, the system is able to pull from the various production rules in order to create content.

Table 10 Example list of production rules for generating content with PhishGen

ID	*Left Rule*	*Right Rule*	*Formality*	*Template Restrictions*
1	{START}	{FORMAT1}	B	1
2	{START}	{FORMAT2}	F	1

3	{FORMAT1}	{<html><body>} {INTRO}{
		}{BODY}{
		} {</body></html>}
			B	*

4	{FORMAT2}	{<html><body>} {HEADER}{
		}{INTROF}{
		} {BODY}{
		}{SIGNATURE}{
		} {</body></html>}
			B	1

5	{INTRO}	{Hi there,}{
		}
			B	*

6	{INTRO}	{Hi [FIRSTNAME],}{
		}
			B	*

7	{INTROF}	{[SGENDER] [LASTNAME],}{
		}
			F	*

8	{INTROF}	{Dear Sir/Madam,}{
		}
			F	1

9	{HEADER}	{[/tmp/img/banner3.jpg,45,100]}{
		}
			B	1

10	{BODY}	{There is a problem with your has([X]).} {Please go to the following as soon as possible, or your has([X]) may be affected: [HREF]}	B	1
11	{BODY}	{There is a problem with your has([X]).} {Please click [HREF:on this link] by [MONTH] [RAND20], 2013 to fix.}	B	1

12	{SIGNATURE}	{Sincerely,}{
		}
			B	1

In our template, we tokenize chunks of text into terminals and variables, and make it so that they can be easily distinguished by our software. We use braces as our token delimiter, and define a non-terminal character as a token composed of a pair of braces separated by all capital letters. For example, the token “{TEST}” would be a non-terminal character, while the token “{THIS IS A TEST}” would not be. This would allow for variables to be defined in such a way that they can be more easily followed by a developer.

For each production rule, as illustrated in Table 10, we define a left rule and a right rule, but we can also add additional features such as formality and template restrictions. This provides an additional level of control over when and where a certain production rule can be applied.

When generating e-mail contents, we found that there were additional features beyond basic syntax or structure that made the e-mail more or less plausible. The tone of the e-mail is highly relevant, for example, as is the formality. As such, we added additional fields to track these classifications, as well as classifying the overall e-mail in order to provide more control over the choice of rules.

An attacker will use different tones when coercing users for different reasons. On one hand, they may be overly nice or consoling in order to gain sympathy. Alternatively, they could be harsh or demanding in order to come across as authoritarian, and try and bully a response out of a user. In the same sense, these different approaches may use different formalities when addressing the victim. Very informal language is used to relay the idea that a person is drafting a personal message, rather than a general e-mail being sent to a broader audience, and typically uses more colloquial expressions. On the other hand, more formal language can be used to provide a sense of professionalism or legitimacy.

During our initial testing, we found that mixing the two types of language often resulted in confused language that made the e-mail suspicious. While some sentences are fairly ambiguous and don't require classification, others can drastically change the tone of an e-mail. If, for example, a rule is chosen that results in highly informal language, we needed to set that as a restriction for the rest of the e-mail.

As we generated more templates, we found that some production rules could be used across multiple templates, while others were too specific and needed to be limited to a particular type of e-mail. Similar to the formality classification, the template restrictions provide a way to limit certain production rules to certain types of e-mails, or to allow them to be used generally across any template, without requiring an overly complex modification of the variable names. Instead of having a non-terminal token "{BODYFORTEMPLATE1}", we can simply assign the template restriction for whatever production rules are only for this template and not intended for global use.

8.3 Content Generation Process Overview

The system we created used a multi-pass expansion of our grammar to first generate the overall structure, and then provide fine-grained expansion of substrings inside the terminal tokens. Before beginning the process, we first test the grammar to ensure that every non-terminal token can be expanded into a series of terminal tokens, in order to prevent loops that might generate infinite text. While this may seem overly restrictive, it greatly reduces the complexity of the grammars and makes developing sets of rules much easier.

Figure 21 shows one possible expansion of our sample grammar from Table 10. At each step, a non-terminal token is expanded until there are not non-terminals left in the final text. In our example, we start with the "{START}" token, which has two possible production rules that can be implemented: 1 or 2. We might choose the production rule 2, which then has additional non-terminals that can be expanded. We then apply production rule 6, followed by production rule 11, to get our final structure.

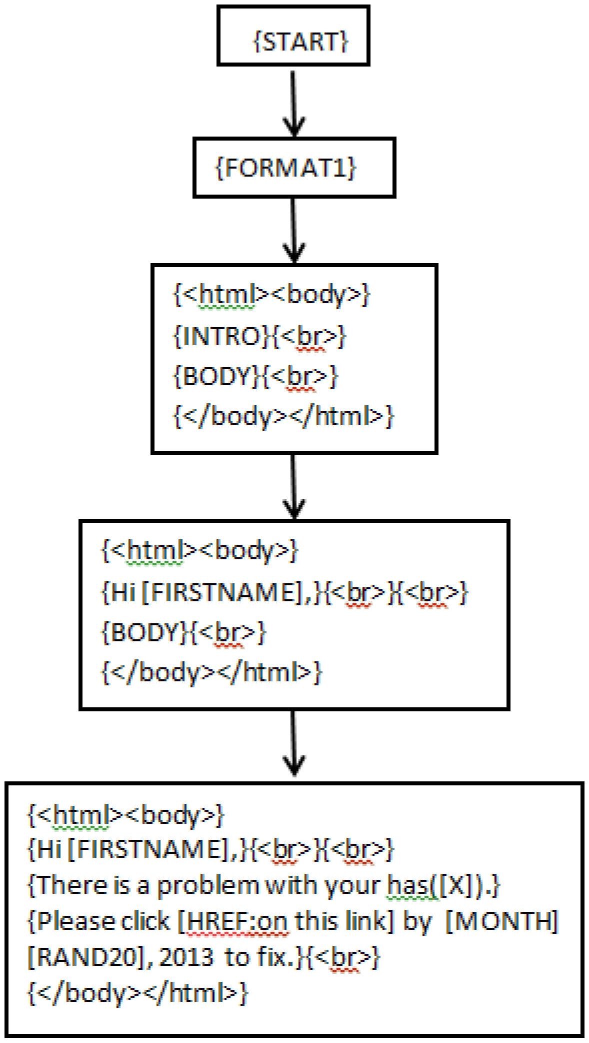

{START}
{FORMAT1}
{<html><body>}
{INTRO}{
}
{BODY}{
}
{</body></html>}
{<html><body>}
{Hi [FIRSTNAME],}{
}{
}
{BODY}{
}
{</body></html>}
{<html><body>}
{Hi [FIRSTNAME],}{
}{
}
{There is a problem with your has([X]).}
{Please click [HREF:on this link] by [MONTH] [RAND20], 2013 to fix.}{
}
{</body></html>}

Figure 21 An example of one possible expansion of the start state using production rules in our sample grammar

At this stage we have a generic structure for an e-mail, but as can be seen in our example there are some elements that still require some resolution. For example, the string “[FIRSTNAME]” should not be used in the final e-mail, rather we need to replace that string with an appropriate value. We also need to make sure that the final e-mail contains some way for our system to map responses back to specific sets of production rules. The current structure, though, contains the general format for an e-mail, so it can be stored off in a library for later use if necessary. As PhishGen creates new e-mails, this library grows, and since each production rule used to generate the structure can be noted along with the actual content, PhishGen can avoid storing duplicates.

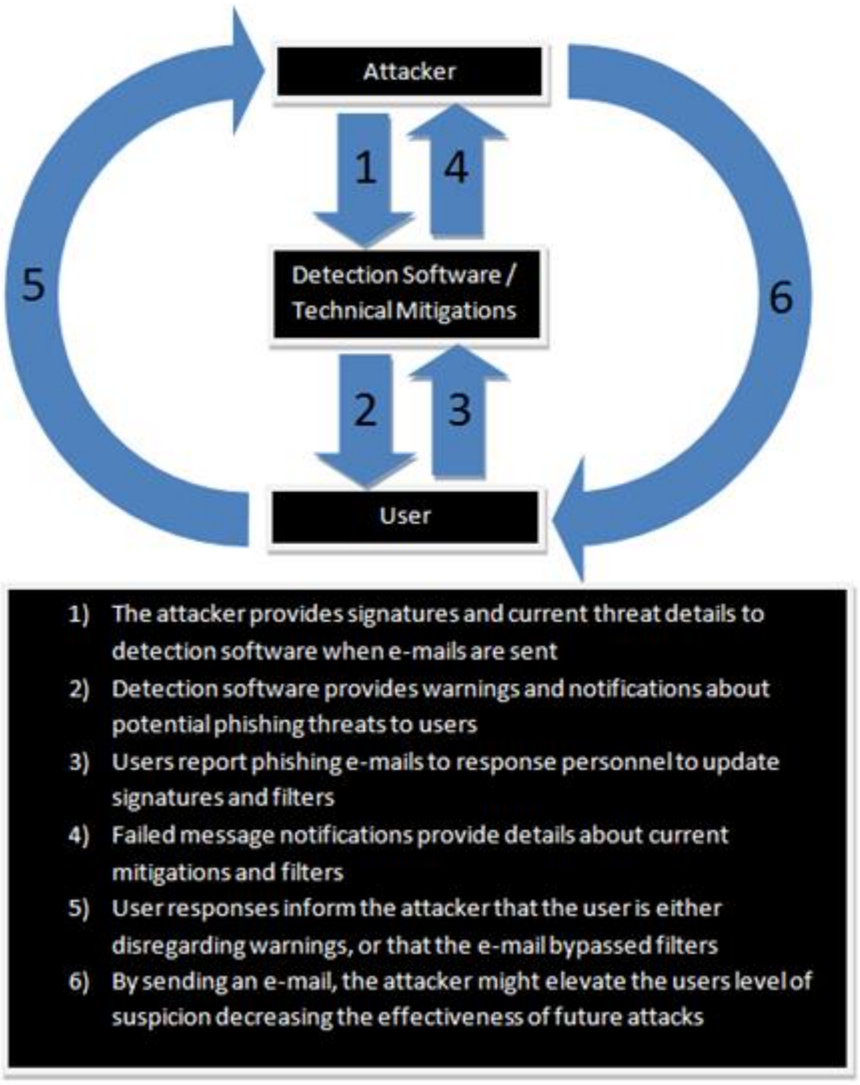
To get to the final e-mail structure, we use variable interpolation and replace certain strings with relevant and context-sensitive values. For this, we define two additional types of elements within the text that will be evaluated by our system: *global* and *relational variables*. Global variables are defined by our system as a string of capital letters within brackets, while relations are defined as a lowercase string followed immediately by a parenthetical. In our example, “[FIRSTNAME]” is a global variable while has([X]) is a relational variable. During the interpolation phase, our system parses the text to find global and relational variables, replacing them with appropriate values that are defined either by the developer or programmatically within the system.

Global variables are for values that will be consistent throughout the entire e-mail, and can actually be decided prior to creating the generic structure. The recipient’s first name is a constant value, as are things like the current month, company names, or an application that is being referenced. The system does not know appropriate values for some of these global variables until the e-mail is ready to be sent, but for the most part these values are predetermined once a target e-mail address is selected.

An added benefit to the use of global variables is that we can swap in additional functionality down the line without requiring edits to the generic templates. For example, we have the global “[HREF]”, which is used to indicate that a URL should be placed in the e-mail. When the system encounters this global placeholder for links, it picks from a list of various approaches for obfuscating links that have been included in the system. Additional obfuscation methods can be added to PhishGen later, though, without the need to modify every template or generic e-mail structure that had been previously been utilized.

The link generation process is critical to the effectiveness of PhishGen. Link obfuscation techniques are one of the areas where phishing has evolved over the years, and is one of the areas where researchers have focused their detection algorithms [92][28][93][94]. As the attack evolves in the future, PhishGen must be able to implement the new attack vectors for testing, as well as implement proactive approaches to link obfuscation that haven’t been detected in the wild. More importantly, though, the link used in each e-mail needs to be unique for each target address, and mapped to a specific set of production rules. If our system detects a response, that response needs to contain enough information for PhishGen to determine which set of production rules were used to generate that e-mail. To accomplish this we merged our existing methods for tracking statistics and other details from phishing responses into the link generation process, and as a result were able to apply novel obfuscation techniques on a per e-mail basis, while maintaining a unique tagging system to map responses to generator rules [57] [95][54].

The substring “has([X])” is an example of a relational variable in our system, and this is another way to enforce semantic context throughout the e-mail. When the system sees “[X]”, it knows to reference previous values that have already been used in the e-mail. For example, the first time the system evaluates “has([X])”, it does not see any previous instances of the relational variable “has()” and so it might resolve it to “password” or some other appropriate value as defined for that relation. The next time the system sees “has([X])”, it will chose a value that has already been used. This allows two sentences that may have been generated by very different production rules in the grammar to have the same overall context. On the other hand, when the system sees “[Y]”, it knows to create a new reference. So if one of our production rules had “has([Y])” in it, the system would find an appropriate value for the “has()” relation that has not already been utilized.



1) The attacker provides signatures and current threat details to detection software when e-mails are sent
2) Detection software provides warnings and notifications about potential phishing threats to users
3) Users report phishing e-mails to response personnel to update signatures and filters
4) Failed message notifications provide details about current mitigations and filters
5) User responses inform the attacker that the user is either disregarding warnings, or that the e-mail bypassed filters
6) By sending an e-mail, the attacker might elevate the users level of suspicion decreasing the effectiveness of future attacks

Figure 22 Possible changes in the environment and potential areas for feedback during a phishing attack

8.4 Information Flow and Feedback

During the course of a phishing attack, there are several ways in which the attack can be detected, and perhaps prevented from succeeding in future attacks. At any point, if the attack is detected as a phishing attack, the attacker will likely not get a response. Additionally, it could also mean that future attacks would not succeed, as detection algorithms might be updated to include new signatures that will detect the same e-mail if it is resent. Figure 22 illustrates the various ways that the environment can change once the attacker sends an e-mail, based on information being reported to and from incident response and other detection measures. From the attacker's perspective, though, once a response is received it means that the e-mail successfully made it through all existing filters, that a valid e-mail address was targeted, and also that the e-mail contents coerced a human into clicking the link.

The same countermeasures that prevent an attacker from getting responses in the wild can also affect responses during a live exercise. For example, e-mails might get sent to Junk mailboxes or quarantined if not properly designed, and exercise participants might not find the e-mail convincing enough. The same feedback cycle can also provide an effective way for our system to learn over time and adapt to changing conditions. Because every e-mail generated by our system can be broken down into a series of production rules, we can determine which rules are effective and which rules are ineffective by tracking participant responses. Our CFG can then be implemented as a PCFG by weighting production rules according to their utility.

If we use our previous e-mail from Figure 21 as an example, we can see how connecting the information flow to the PCFG production rule weights can be implemented. As the e-mail is being generated, we pick randomly from whichever production rules have the largest weight values. We also decrease the weight of every production rule and variable interpolation technique as it is applied. So, for example, in the first replacement we would compare the weight of production rule 1 to production rule 2, and if chosen decrease the weight for production rule 2. The same process is repeated for each step until the generic format is created. When generating the links for specific e-mail as indicated by “[HREF]”, we would also chose techniques based on their weights, and similarly decrease the weights for the various obfuscation techniques that were used.

Once the attacker sends the e-mail to a mail server, it may be scanned or analyzed by phishing detection systems, at which point it might be flagged as suspicious and quarantined. In this case, there is some production rule or set of production rules that created content that ended up being suspicious, but those rules now have a decreased weight after having been chosen. The same is true if the e-mail makes it to the local e-mail client, and local spam filters mark the e-mail as spam and move it to the Junk mailbox. In any situation where the e-mail does not generate a response, our system assumes that any production rule used to generate that e-mail could be the reason for the failure. If, however, the e-mail makes it to the Inbox, and the user does not identify the e-mail as suspicious, they might click a link, at which point some feedback can be detected by the attacker. This feedback contains unique information to that e-mail, so PhishGen can then determine which specific set of production rules were used, and now it knows that those production rules did not create content that would be caught by detection methods. It also knows that those production rules created content that were convincing enough to trick a user, and so the weights for all associated production rules for that e-mail are increased. The next time e-mails are generated, these new weights are considered when making decisions, and so previously successful production rules should have a better chance at being chosen.

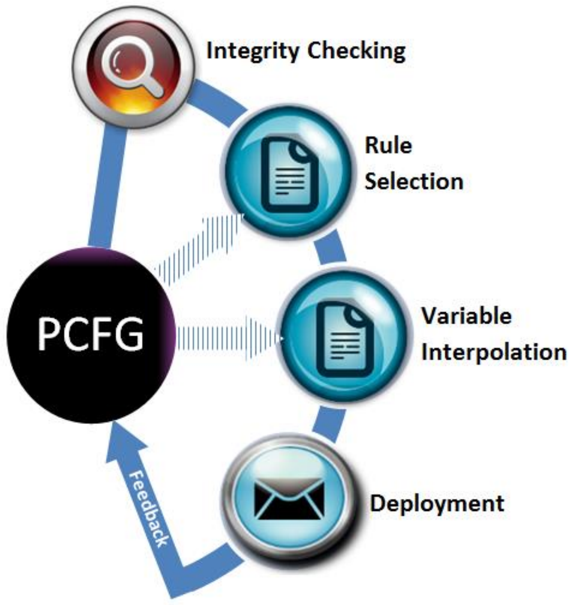


Figure 23 Process overview for generating content in PhishGen

The final process overview for PhishGen is illustrated in Figure 23. First the developer provides a set of PCFG rules, and selects a set of target e-mails and some other details such as the scheme to use. The system performs some integrity checks on the rules to make sure they are consistent, and ensures that values are defined for any available placeholder variables. The system then goes through and creates the generic structure of the e-mail by selecting production rules according to their weights. Values are then substituted in for placeholder variables and other context-sensitive elements in the generic structure, and global values are also substituted in where appropriate. The e-mail is then delivered, and responses are tracked at a capture site. The responses trigger weight adjustments for production rules that were used in successful e-mails, and the new weights are used for the next e-mails that is generated.

8.5 User Interface Design

As a proof of concept to show that our approach would actually generate useful content, we developed a tool called HyperTwish to craft content that would fit into SMS or text messages [96]. Rather than use standard texting services, which can be expensive, we decided to use Twitter as our testing medium. Twitter provides a web-interface for sending short messages that are essentially SMS messages, and developing software that can interact with this web service is very simple. Twitter caps the size of messages at 140 characters as well, so maintaining context across a large amount of text wasn’t an issue. A text message or tweet generator also provides a method for testing our ability to create online content such as links or images.

HyperTwish was not tested on actual users, but rather as a proof-of-concept for our grammar approach to content generation. The content was generated as expected, and the overall approach was validated to the point that we could expand the approach to longer messages. However one major take away was that a more efficient way of developing templates and production rules was needed. After all, one of our main design goals was to provide an efficient and cost-effective method for creating test cases. We initially started out with a basic spreadsheet format, but as the concept of global and relational variables developed, the template structure quickly became unwieldy. From a distribution perspective, it would also have been difficult to relay instructions for building a template as well.

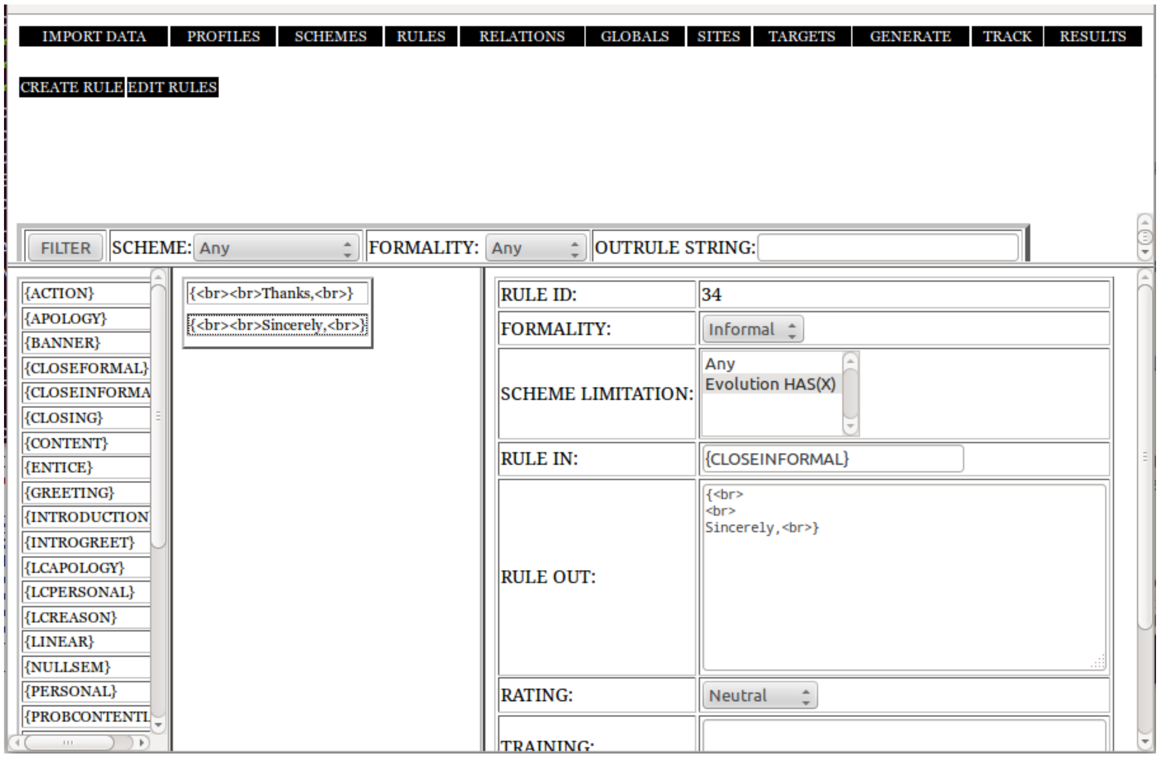
To solve the issue of creating and managing production rules, global variables, relational variables, and other placeholders, we created PhishGen as a web-based application. The interface provides developers with a central location to store templates, quickly add or adjust production rules, edit weights, and generally customize the e-mail generation process to their specific needs.

Figure 24 illustrates the user interface for the PhishGen web application, specifically the production rule modification interface. The developer starts by importing data into the system in the form of some basic spreadsheets to can be used as a starting point. These sample files are packaged with the system, and can be easily modified. Once imported, the developer can go through the various tabs and adjust specific variables for the system.

The “Profiles” tab handles target domain details for sending e-mails. For each profile, the developer defines a domain, and the associated mail servers and websites. This allows the PhishGen to run exercises that include target e-mails across multiple domains, because each e-mail is sent through a relevant mail server for that e-mail address. Additionally, obfuscation techniques can fill in relevant website addresses for links.

The “Schemes” tab is where developers can modify overall descriptions for templates that have been developed. A scheme includes the production rules for a specific type of e-mail, and PhishGen provides the ability to classify these templates according to various classification definitions.

Details about specific production rules can be modified under the “Rules” tab. The developer can select a non-terminal value from a list of available variables in one menu, and see all available expansions in another menu. By selecting one of these expansions, the developer can edit values and change weights, and they are given the option of creating a new production rule with the new values. In this way, the number of available production rules can quickly be expanded. The same basic methodology for editing rules can be used to edit relations and global values under the “Relations” and “Globals” tabs.



RULE ID:	34
FORMALITY:	Informal
SCHEME LIMITATION:	Any Evolution HAS(X)
RULE IN:	{CLOSEINFORMAL}

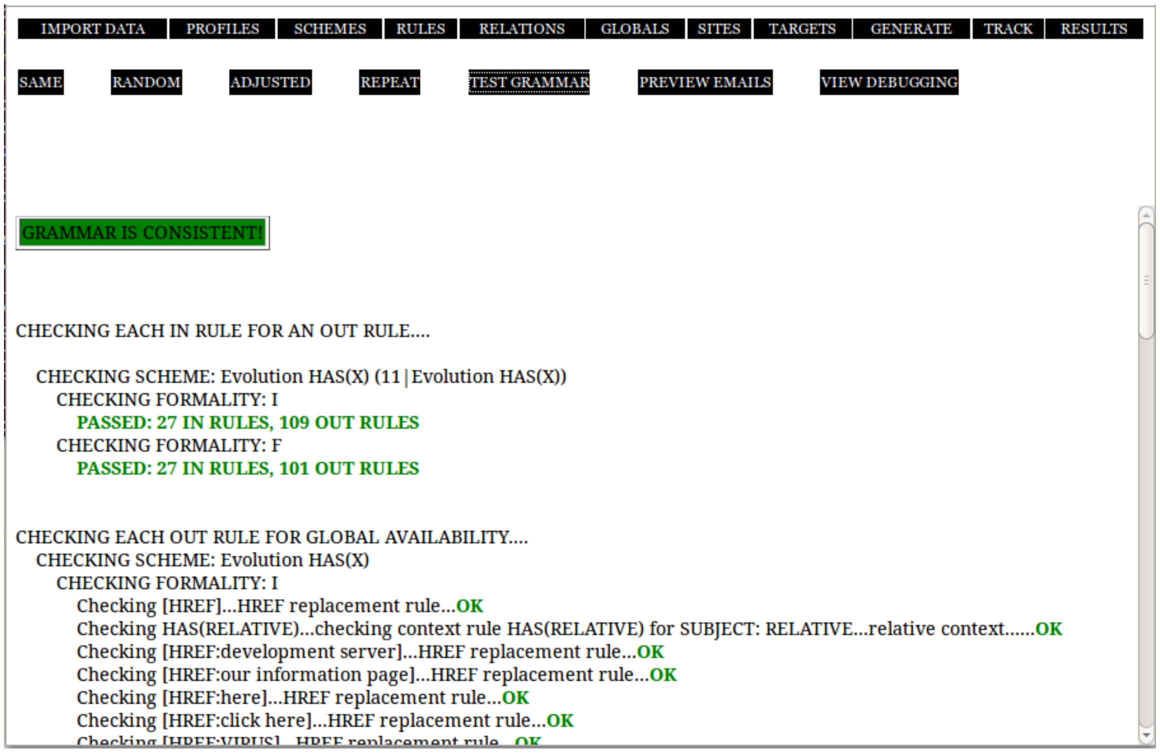
RULE OUT:	{
	
	Sincerely,
	}

RATING:	Neutral
TRAINING:	

Figure 24 Illustration of the web-based user interface for PhishGen

One important component in generating links is to tie them into a domain that is hosting a capture server. If the user clicks a link, for example, the request needs to be sent to a server that can register the response with PhishGen. Under the “Sites” tab, the developed can add domains that can be used when generating links.

The “Targets” tab provides an interface for importing lists of e-mail addresses that can be used during exercises. Each lists is treated as a separate sample populations or cohort, so developers can generate sets of e-mails for specific target lists to support specific test objectives and schedules.



```
CHECKING EACH IN RULE FOR AN OUT RULE....

   CHECKING SCHEME: Evolution HAS(X) (11|Evolution HAS(X))
      CHECKING FORMALITY: I
         PASSED: 27 IN RULES, 109 OUT RULES
      CHECKING FORMALITY: F
         PASSED: 27 IN RULES, 101 OUT RULES


CHECKING EACH OUT RULE FOR GLOBAL AVAILABILITY....
   CHECKING SCHEME: Evolution HAS(X)
      CHECKING FORMALITY: I
         Checking [HREF]...HREF replacement rule...OK
         Checking HAS(RELATIVE)...checking context rule HAS(RELATIVE) for SUBJECT: RELATIVE...relative context......OK
         Checking [HREF:development server]...HREF replacement rule...OK
         Checking [HREF:our information page]...HREF replacement rule...OK
         Checking [HREF:here]...HREF replacement rule...OK
         Checking [HREF:click here]...HREF replacement rule...OK
```

Figure 25 E-mail generation procedures in PhishGen

Once the underlying production rules and variables are setup, the developer can proceed to generating sets of e-mails under the “Generate” tab. This interface provides a convenient way of testing the production rules to make sure there are no infinite loops, and for validating that all possible global and relation references have available values.

Figure 25 shows the process of testing the production rules for consistency, and also illustrates some of the available e-mail generation strategies included in the system. This tab also provides an interface to preview e-mails that have been generated, and to view debugging output describing information about choices made in the e-mail generation process.

Finally, the "Track" tab shows a list of responses that have been detected and the associated e-mail that generated the response, while the "Results" tab provides generated charts and graphs that track various features of the generated sets of e-mails. This data was primarily used for the purposes of validating our approach, however we left this capability in the tool in case it might be useful for other researchers.

8.6 Content Generation Strategies

The strategies for generating a sets of test e-mails in PhishGen, as shown under the "Generate" tab in Figure 25, includes strategies that emulate existing methods, as well as various approaches to take advantage of the PCFG weights and other automated features. For example, the "Same" strategy creates a single e-mail structure based on the current production rules and applies it to each target address on the chosen target list. This approach is basically the way current exercises are typically run, where every participant receives the same e-mail. We included this feature in order to test various features of the generated e-mails when compared to our other strategies.

The "Random" approach creates a random e-mail for each address on the chosen list, but does not take into account production rule weights when making decisions. We used this approach as a sort of benchmark for later tests to see whether our other approaches were more or less effective, but it is also a useful approach when testing incident response procedures. If each user receives a potentially unique e-mail, then when they report that e-mail it does not reflect the e-mail that other users have received. In this way, it can test the network defenders ability to track down multiple threats during an exercise.

The “Adjusted” approach takes full advantage of the adjusted production rule weights in our PCFG. As e-mails are generated for the chosen target list, the production rules are chosen based on their current weight, and adjusted once the rule is used. This is done for each e-mail individually, so the first target e-mail on the list receives crafted content based on different weights than the second target e-mail. The overall effect is that the participants all receive different e-mails during the first exercise, similar to the “Random” approach, but during additional rounds the e-mails reflect adapted production rule weights based on user responses.

With the “Repeat” strategy, PhishGen determines a set of production rules that created a successful e-mail based on participant responses, and reuses that exact set of production rules for every e-mail address on the target list. Essentially, it finds a success, and replays that same e-mail to everyone for the current exercise. This is similar to how an attacker in the wild might operate, where they try a couple different approaches at first and then reuse the most successful. We used this approach when testing other features of our generated e-mail to demonstrate.

CHAPTER 9:

CONTENT CHARACTERISTIC SIMULATIONS

By implementing a PCFG, PhishGen has the ability to adapt the content generation process for future rounds based on feedback from previous exercises. Over time we can isolate specific production rules that are not being caught, and favor those rules over production rules that may result in filtering. This feature is ultimately dependent on the ability to properly balance weights for the production rules.

To test how different production rule weights and other variables affected the diversity of our generated datasets, as well as to test the ability to avoid detection, we performed multiple simulations that tested various settings within PhishGen. For example, we needed to ensure that the amount that a weight was decreased was not excessive to the point the rule would never be used again, and we also needed to ensure that when a weight was increased due to a detected response that it didn’t become so dominant that other rules would never be chosen.

9.1 Simulation Environment and System Settings

To evaluate the various approaches of setting bounds on the weights, we first needed some way of running simulations, without involving actual users. Testing these bounds would require thousands of e-mails, and this was deemed too intrusive for live human testing. Instead, we sent test e-mails to a single account on a live corporate network that had e-mail threat detection features in place including Ironport spam filtering. According to the vendor website, the product also performs some level of context-sensitive analysis of incoming e-mail messages [97]. The local e-mail client used for testing was Microsoft Outlook with McAfee E-mail Scan implemented.

Simulations were run with multiple approaches to e-mail generation. For the simulations using our PCFG approach, different upper-bounds were placed on the weights for production rules in the PCFG. For each simulation, we sent 4 rounds of 100 e-mails, adjusting the weights after each round according to preset values for that simulation. Multiple simulations were run with each upper-bound value to include variations in the Outlook Junk e-mail filter settings, as well as the underlying production rules [98]. We used an optimistic approach, and assumed that every e-mail that made it through to the Inbox of our account would result in a response and considered a success. Anything that was quarantined or sent to the Junk mailbox was determined a failure.

During initial tests, we quickly realized that there was a discrepancy in the amount and frequency that weights were decreased versus increased. Since the weights are adjusted after each e-mail is generated, the weights for rules are frequently reduced, whereas even in an optimistic exercise it is less frequent that a weight will be increased. Even if a certain rule is effective, it might have been used for a large test population, and as a result it would have been decreased so often that the effectiveness is nullified. As a result, we placed a lower-bound of 0 for the weights of production rules. For new rules entering the system, we set an initial weight of 100 to ensure that unused rules will always be chosen prior to rules that have been used extensively. Similarly, when a rule is used the weight value is decreased by 1, while a success results in the weight being increased by 5.

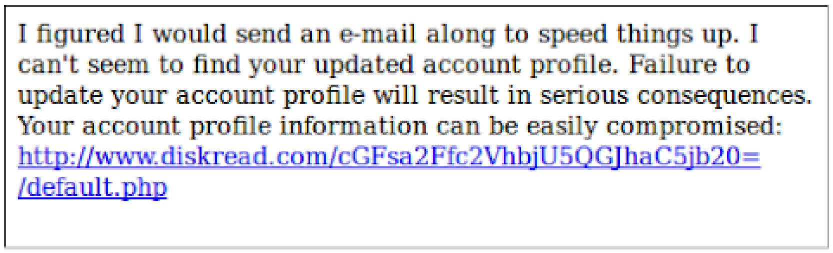
While the initial weights and increase/decrease parameters are essentially arbitrary values, as long as they are consistent between the various simulations the effects of the upper-bound testing will be effective. For example, a high upper-bound with a high weight increase value would have the same result as a lower upper-bound with a lower weight increase amount. With a static increase value, the only variable being evaluated is the upper-bound limitation.

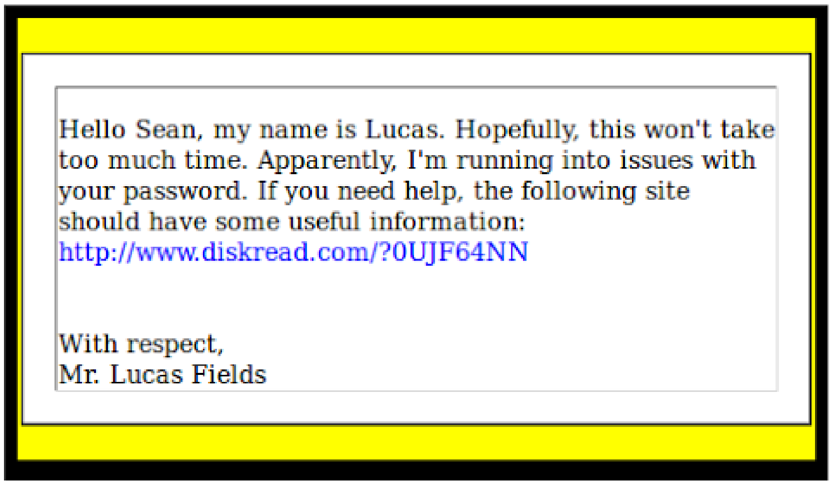
9.2 Learning, Creation, and Diversity Metrics

For each of our simulations, we collected data on learning, creation and diversity. Conclusions about learning were based on the number of e-mails that successfully reached the Inbox. By tracking the throughput across multiple rounds of testing, we can measure the ability of a given approach to adjust rule selection in order to increase the number of successful e-mails.

The creation metric measured the number of e-mails created using unique sets of production rules. Each time the system generates an e-mail by selecting production rules, it keeps track of the rules that were selected. This is critical in order to deduce which rules may have resulted in filtering, but it also allows us to compare e-mails to evaluate whether they used different production rules for generation.

The diversity metric measured the average differences between e-mails in a single set, which we used to show that the e-mails being generated were different as a result of rules being adjusted. For example, we wanted to ensure that the same e-mail was not being generated over and over again, as would happen with the “Repeat” strategy described earlier. Instead we wanted measurable differences in the e-mails, such as in Figure 26 which illustrates two significantly different e-mails that were generated from the same set of production rules.


I figured I would send an e-mail along to speed things up. I can't seem to find your updated account profile. Failure to update your account profile will result in serious consequences. Your account profile information can be easily compromised: http://www.diskread.com/cGFsa2Ffc2VhbjU5QGJhaC5jb20=/default.php


Hello Sean, my name is Lucas. Hopefully, this won't take too much time. Apparently, I'm running into issues with your password. If you need help, the following site should have some useful information: http://www.diskread.com/?0UJF64NN

With respect,
Mr. Lucas Fields

Figure 26 Example of two different e-mails generated from the same available production rules in PhishGen

To measure the diversity in a set of e-mails, we used a diversity measurement shown in **Error! Reference source not found.**. The average distance between e-mails as computed using the normalized Levenshtein distance, which is a measurement of how many character changes would be required to make two strings the same. This was multiplied by the number of unique e-mails, which was determined by the number of unique sets of production rules used to create an e-mail, and then divided by the total number of e-mails created multiplied by the average character length of all e-mails. We then multiplied the resulting value by 100 for display purposes.

In order to ensure that the diversity calculation is not being biased by content such as user names, e-mail addresses, or unique identifiers in links, these elements were all removed to provide a normalized value for comparison. As a result, the Levenshtein distance was calculated using only significant contextual elements as opposed to values used for global replacements. The diversity equation also takes into account the number of e-mails generated and the number of unique e-mails in order to prevent elevated diversity ratings for large sets of e-mails that simply repeat the same content.

Equation 2 Formula for measuring diversity in a set of e-mails

$$Diversity = \frac{(Emails_{unique})\ (Distance_{avg})}{(Emails_{total})\ (Length_{avg})} * 100$$

9.3 Hypothesis

Our initial hypothesis was that having no upper bounds on the weights would result in certain rules becoming so dominant that there would be no variation in e-mails, but that once an appropriate upper-bound was determined our approach would outperform existing methods when measured against learning, diversity, and creation metrics.

9.4 Results

Figure 27 shows the learning measurements for several different upper-bound values as well as a comparison against existing methods for e-mail generation. An upper-bound value of 400 performed the best out of the values we tested. Setting the upper bound at the same level of the initial weight for production rules resulted in the same type of behavior as randomly generating content. This is because successful rules had as much chance of getting chosen as untested rules, and so the selection of rules that resulted in detection was higher. However, when the upper bound on production weights was higher than the initial value, it allowed more effective rules to be preferred over new rules. As the upper bound was increased, it provided more room for successful rules to outpace less successful rules. This is because the weight value for successful rules, after being decreased upon selection, was still higher than the weight for less successful rules.

When compared to existing methods, our approach out performed most methods. When randomly generating e-mails, the throughput to the inbox never reached 100%, there were always some e-mails that got filtered. As a result, this approach was not able to learn and create more effective content. However with this approach the results were fairly consistent across multiple rounds of e-mails, which could be seen as an advantage. Not using an upper-bound resulted in a lower rate of learning for the second round compared to our upper-bound testing, however as more successful rules became dominant the throughput reached 100% in round 3, indicating the system had learned to generate only effective content. The current methodology of generating a single e-mail for all participants had the worst throughput measurements overall. Essentially, it was hit or miss with each round. If the e-mail was filtered, no e-mails got through, otherwise all of them got through. Over multiple simulations this averaged out to a much lower throughput, but also indicated an inability to effectively learn from previous exercises.

Figure 28 illustrates the diversity measurements for the various upper-bound simulations and comparisons with existing methods. Again, setting the upper-bound at the same value as the initial weight for new rules resulted in the same kind of measurement as randomly picking rules. For other weights, there appeared to be more diversity in the second round for an upper bound of 200, however all of the diversity measurements appeared to stabilize near round 3.

Diversity measurements for other methods were dramatically lower than our approach, with random generation of e-mails being the only exception. When the e-mails are all randomly generated, there is very little similarity across the set of e-mails, resulting in a consistently high diversity measurement across all exercises. Our method had a decrease in diversity over the course of several rounds of e-mails, primarily because more successful rules were chosen more often. However, when there was no upper-bound placed on the weights, the diversity quickly dropped to zero because the same set of rules were consistently chosen over several exercises. This same thing happens with when replaying a successful e-mail from the first round. The current method, where all participants receive the same e-mail in each round, had a diversity measurement of zero for all rounds.

Figure 29 illustrates the number of new e-mails created for our various upper-bounds simulations as well as for other methods. Once again, setting the max weight to the same value as the initial weight for new rules resulted in creation metrics similar to random generation. Overall, though, most of our methods consistently created a large number of unique e-mails for each round. An upper-bound of 400 had the best performance across all four rounds of testing though.

When compared with other methods, our approach again performed better when measuring the number of unique e-mails created per round. Without putting an upper-bound on weights, the same rules were chosen more often resulting in no new e-mails being created after the second round of testing. This same behavior was observed when replaying a successful e-mail from previous rounds. While the current approach of sending the same e-mail to all participants did create new content for each round, it was only a single e-mail.

Table 11 Creation, learning, and diversity results for all content characteristic simulations

Learning Results							
Filter Settings	*Generation Method*	*Email Scheme*	*Emails Sent*	*Round 1*	*Round 2*	*Round 3*	*Round 4*
HIGH	CAP 200	11	100	7	50	100	100
HIGH	CAP 300	11	100	7	48	99	100
HIGH	RANDOM	11	100	11	9	14	12

HIGH	RANDOM	2	100	88	83	90	87
HIGH	REPEAT	11	100	8	100	100	100
HIGH	REPEAT	2	100	87	100	100	97
LOW	CAP 100	11	100	45	52	50	63
LOW	CAP 100	2	100	100	100	100	100
LOW	CAP 200	11	100	45	88	100	100
LOW	CAP 300	11	100	45	85	100	100
LOW	CAP 400	11	100	45	85	100	100
LOW	CAP 500	11	100	45	81	100	100
LOW	NOCAP	11	100	41	73	100	100
LOW	RANDOM	11	100	43	56	47	56
LOW	RANDOM	2	100	100	100	99	100
LOW	SAME	1	100	0	100	100	0
LOW	SAME	11	100	0	0	0	0
LOW	SAME	2	100	100	100	100	100
WEB	RANDOM	11	100	100	100	100	100
WEB	RANDOM	2	100	100	100	100	100
LOW	RANDOM	11	100	52	43	49	42
LOW	RANDOM	2	100	97	100	99	98

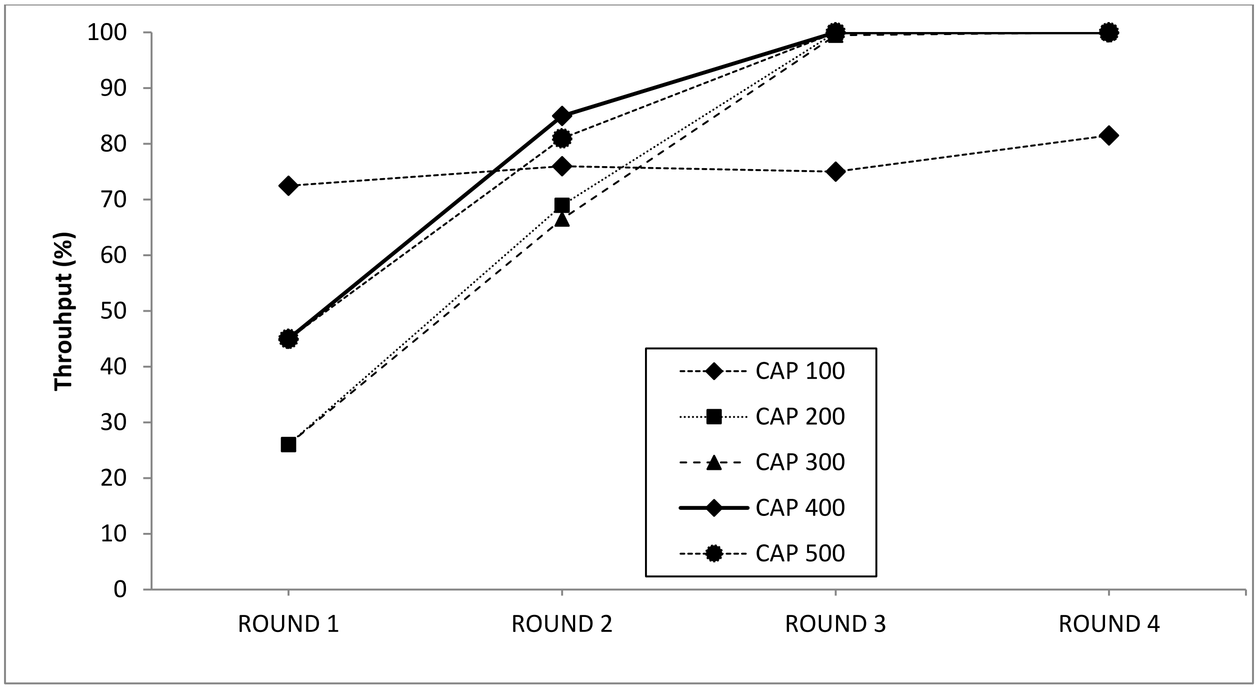
Diversity Results

Filter Settings	*Generation Method*	*Email Scheme*	*Emails Sent*	*Round 1*	*Round 2*	*Round 3*	*Round 4*
HIGH	CAP 200	11	100	82.14	58.23	17.98	18.08
HIGH	CAP 300	11	100	82.14	58.25	18.18	13.63
HIGH	RANDOM	11	100	82.83	75.59	82.04	76.04
HIGH	RANDOM	2	100	43.06	42.34	43.56	43.72
HIGH	REPEAT	11	100	82.13	61.37	58.68	48.43
HIGH	REPEAT	2	100	41.45	41.29	41.19	40.29
LOW	CAP 100	11	100	82.12	81.2	81.05	80.4
LOW	CAP 100	2	100	41.45	40.37	40.39	40.38
LOW	CAP 200	11	100	82.12	22.26	18.54	18.93
LOW	CAP 300	11	100	82.12	22.27	16.34	15.53
LOW	CAP 400	11	100	82.12	22.27	15.4	11.43
LOW	CAP 500	11	100	82.12	22.28	15.12	7.8
LOW	NOCAP	11	100	82.39	5.81	1.31	1.31
LOW	RANDOM	11	100	82.83	75.59	82.04	76.04
LOW	RANDOM	2	100	43.06	42.34	43.56	43.72
LOW	SAME	1	100	0	0	0	0
LOW	SAME	11	100	0	0	0	0
LOW	SAME	2	100	0	0	0	0
WEB	RANDOM	11	100	82.83	75.59	82.04	76.04
WEB	RANDOM	2	100	43.06	42.34	43.56	43.72
LOW	RANDOM	11	100	75.02	82.77	84.78	78.31
LOW	RANDOM	2	100	43.41	43.25	41.89	43.46

Creation Results

Filter Settings	*Generation Method*	*Email Scheme*	*Emails Sent*	*Round 1*	*Round 2*	*Round 3*	*Round 4*
HIGH	CAP 200	11	100	100	97	65	62

HIGH	CAP 300	11	100	100	97	62	38
HIGH	RANDOM	11	100	100	100	100	100
HIGH	RANDOM	2	100	100	100	100	100
HIGH	REPEAT	11	100	100	0	0	0
HIGH	REPEAT	2	100	100	0	0	0
LOW	CAP 100	11	100	100	100	96	96
LOW	CAP 100	2	100	100	100	100	100
LOW	CAP 200	11	100	100	100	100	100
LOW	CAP 300	11	100	100	100	100	99
LOW	CAP 400	11	100	100	100	100	99
LOW	CAP 500	11	100	100	100	100	99
LOW	NO CAP	11	100	96	11	0	2
LOW	RANDOM	11	100	100	100	100	100
LOW	RANDOM	2	100	100	100	100	100
LOW	SAME	1	100	1	1	1	1
LOW	SAME	11	100	1	1	1	1
LOW	SAME	2	100	1	1	1	1
WEB	RANDOM	11	100	100	100	100	100
WEB	RANDOM	2	100	100	100	100	100
LOW	RANDOM	11	100	100	100	100	100
LOW	RANDOM	2	100	100	100	100	100



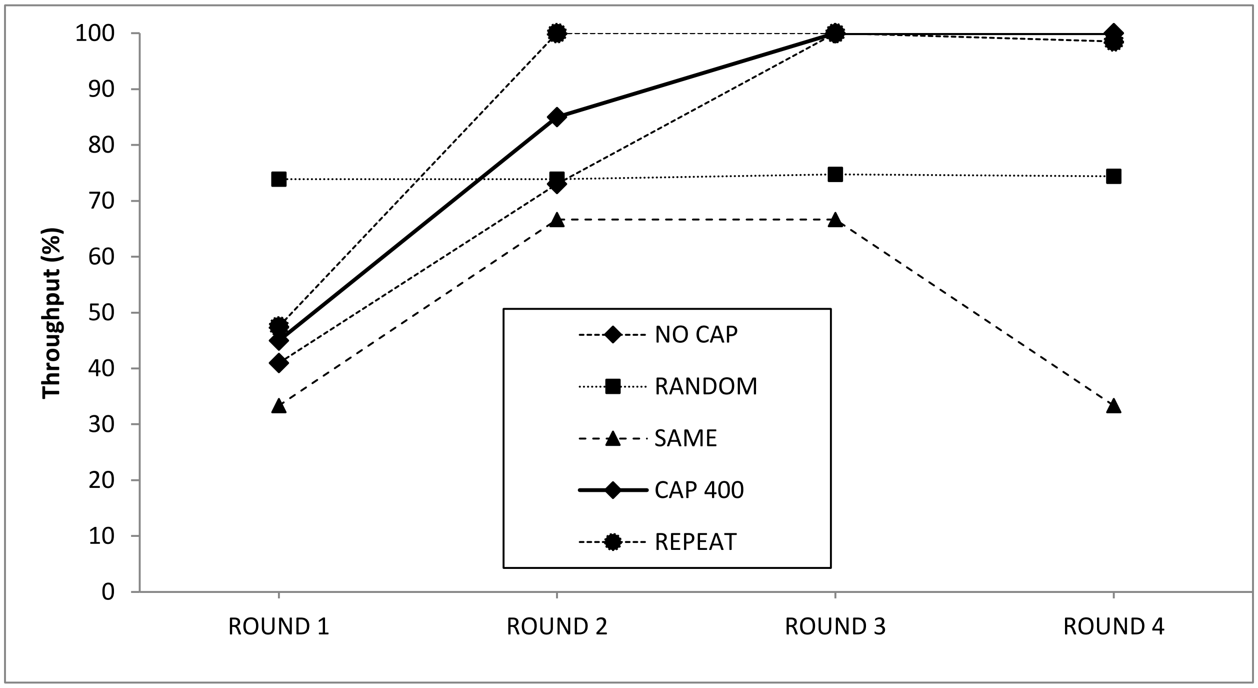
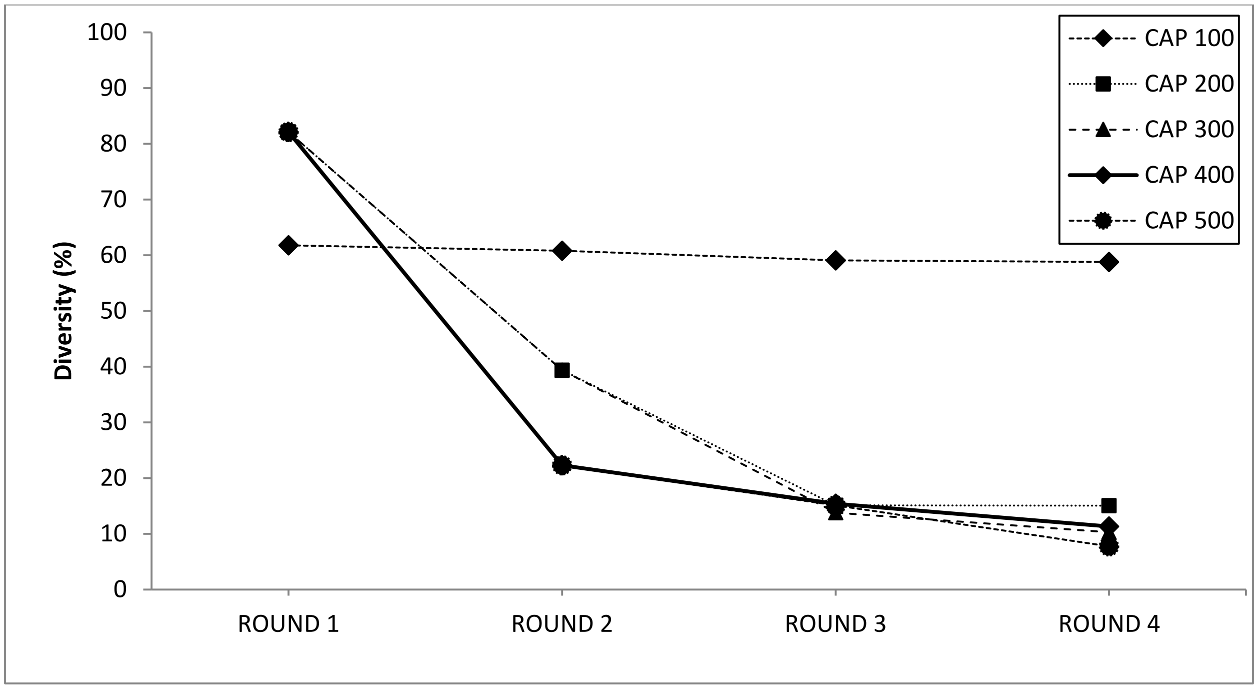


Figure 27 Learning results graphs for different threshold values and content generation approaches



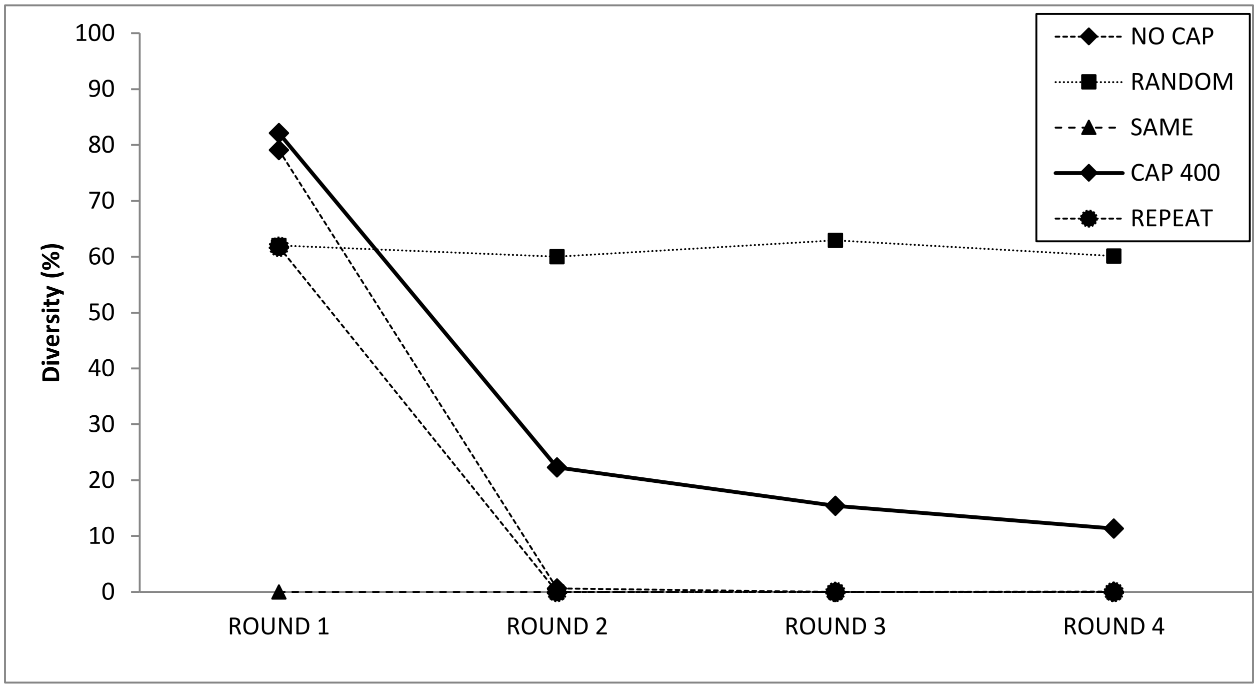
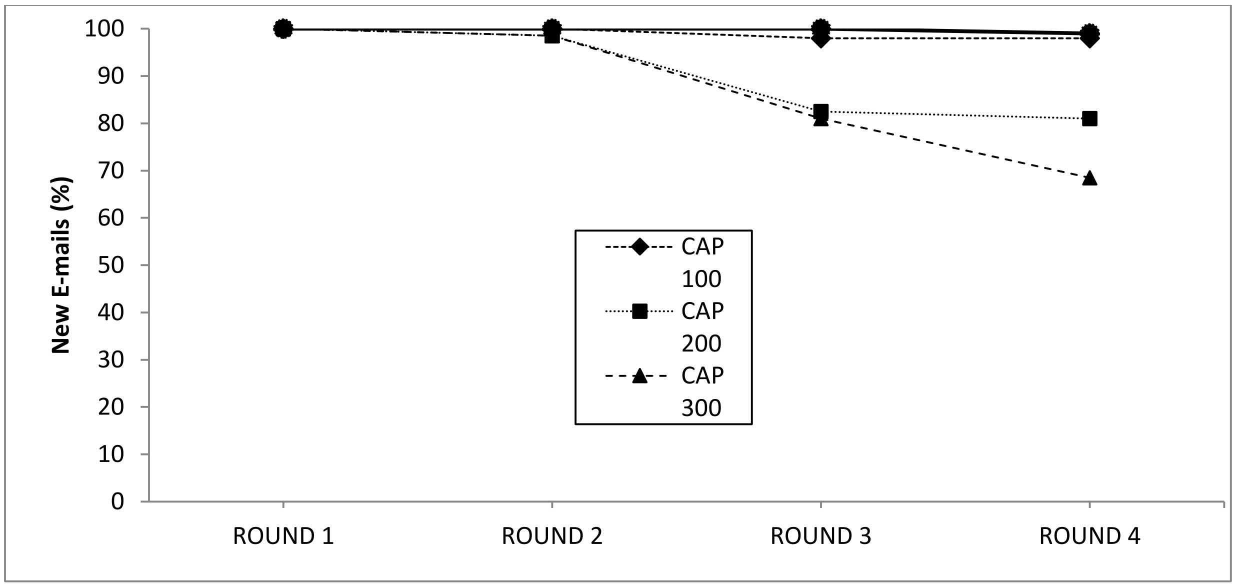


Figure 28 Diversity results graphs for different threshold values and content generation approaches



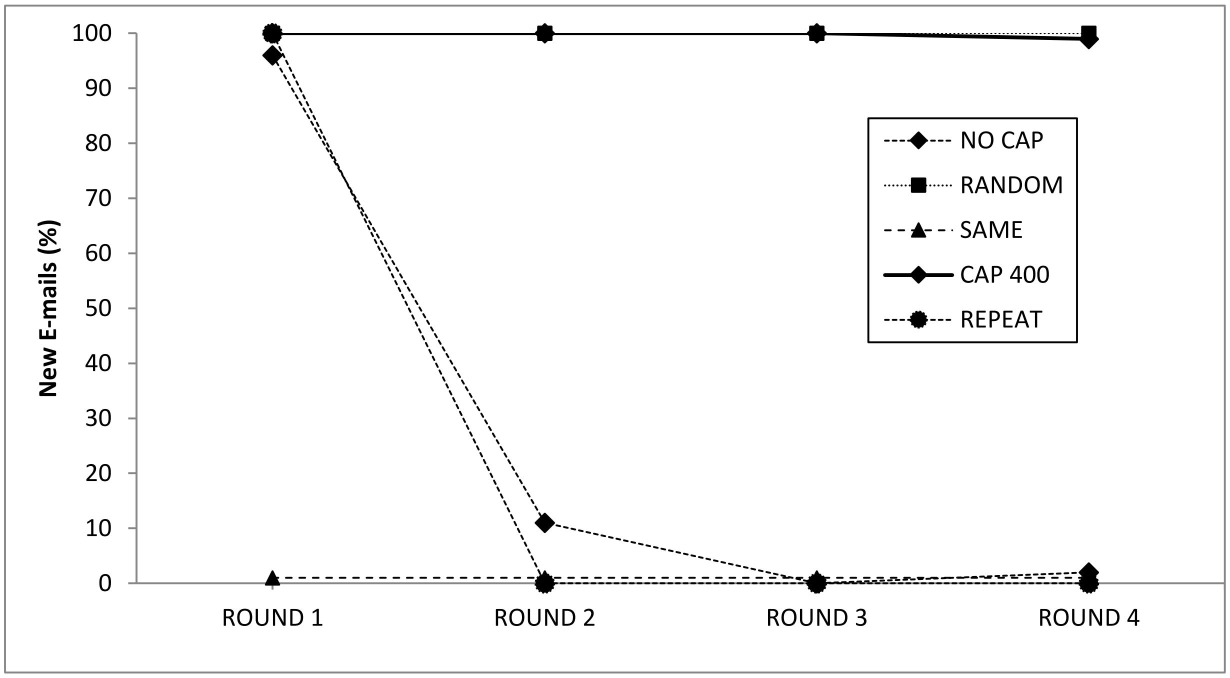


Figure 29 Creation results graphs for different threshold values and content generation approaches

9.5 Analysis and Conclusion

Based on the results of our simulations, we confirmed our hypothesis that without setting an upper-bound on production rule weights, the performance rapidly degrades over the course of several rounds of testing. The data supports the conclusion that, because a small set of rules quickly become dominant, it is the same as replaying a successful e-mail from previous rounds.

For the values we used for initial production rule weights, as well as the amount weights would be decreased or increased, we found that placing an upper-bound of 400 on production rule weights was most effective. More extensive testing would likely be able to determine an optimal value, however the overall improvement in performance would be marginal.

We found that most existing methods tended to perform well in one area, at the cost of the others. For example, while repeating a successful e-mail resulted in an immediate spike in learning for the second round, the diversity and creation metrics dropped to zero as a result. Randomly generating e-mails without taking into account production rule weights resulted in a high number of unique e-mails being generated, but diversity and learning were never able to reach 100%. Our approach consistently achieved higher performance ratings across all three categories.

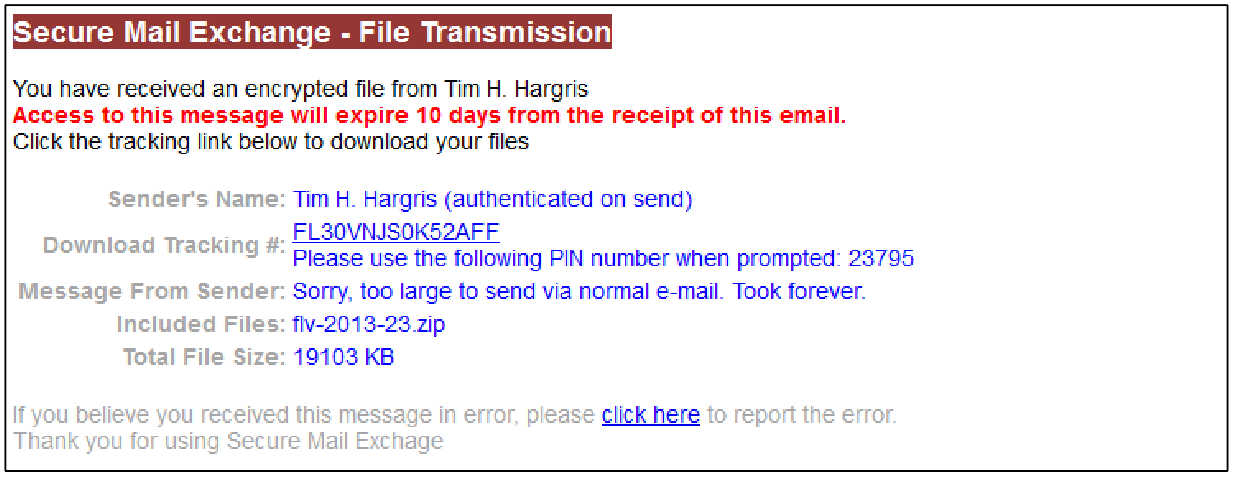
CHAPTER 10:

EVALUATION OF EFFECTIVENESS IN LIVE EXERCISES

While our simulations validated our overall approach to content generation, the simulations only evaluated features based on an optimistic approach that did not include live users. In addition to showing that PhishGen is able to create content, we also needed to show that PhishGen could create content that was as effective in live exercises as existing methods.

10.1 Hypothesis

Our hypothesis was that PhishGen could generate semantically valid and convincing e-mail content that performs as well as manually generated e-mails in live exercises. A total of six types of e-mails were developed for use in the study: a request to download a secure file download request, similar to the one used in our Content Impact study, as shown in Figure 30; a bogus order confirmation from a fictitious online retailer shown in Figure 31; a request to download an electronic fax shown in Figure 32; a request to donate to a relief fund shown in Figure 33; a notification that the user won a free cruise shown in Figure 34; and a request to take an online IQ test shown in Figure 35. For each of these e-mails, a version was manually created by a subject matter expert, while another variation was generated by PhishGen.



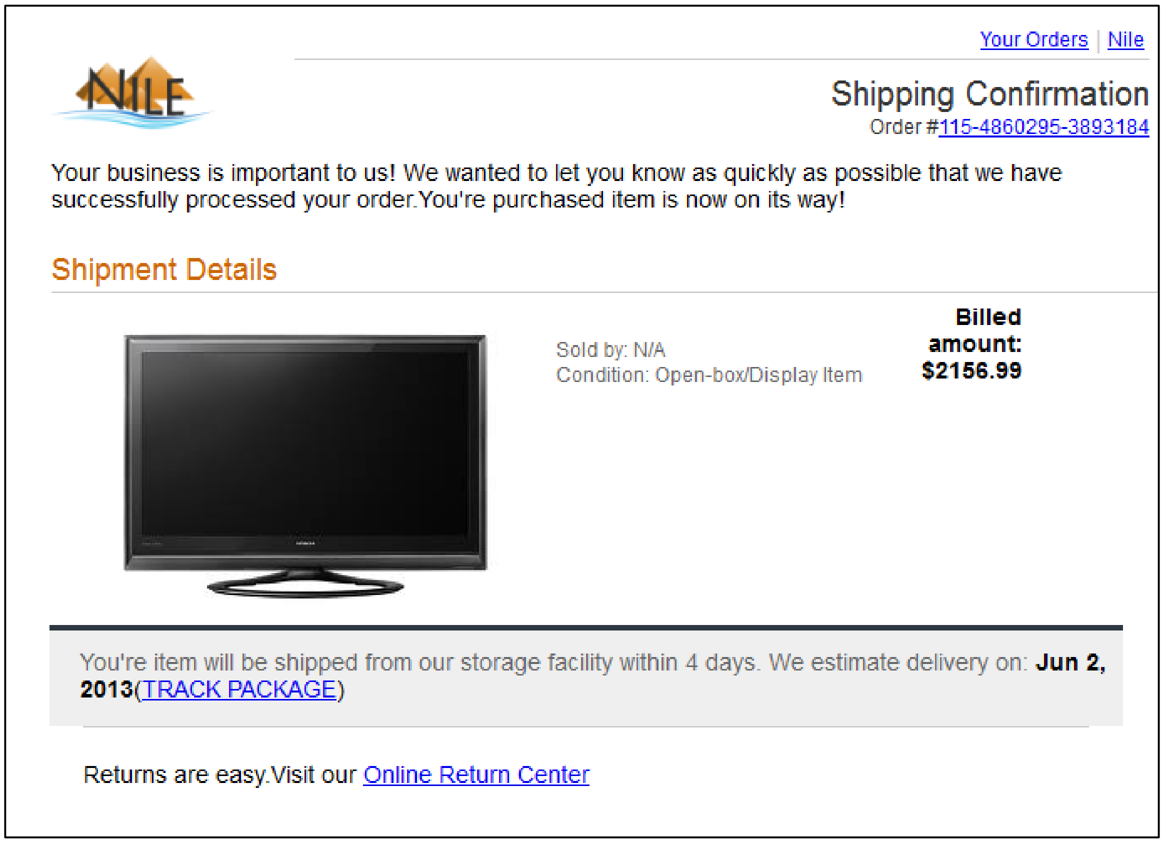
Secure Mail Exchange - File Transmission

You have received an encrypted file from Tim H. Hargris
Access to this message will expire 10 days from the receipt of this email.
Click the tracking link below to download your files

Sender's Name:	Tim H. Hargris (authenticated on send)
Download Tracking #:	FL30VNJS0K52AFF Please use the following PIN number when prompted: 23795
Message From Sender:	Sorry, too large to send via normal e-mail. Took forever.
Included Files:	flv-2013-23.zip
Total File Size:	19103 KB

If you believe you received this message in error, please click here to report the error.
Thank you for using Secure Mail Exchage

Figure 30 Secure e-mail generated by PhishGen


Your Orders | Nile

NILE

Shipping Confirmation
Order #115-4860295-3893184

Your business is important to us! We wanted to let you know as quickly as possible that we have successfully processed your order.You're purchased item is now on its way!

Shipment Details

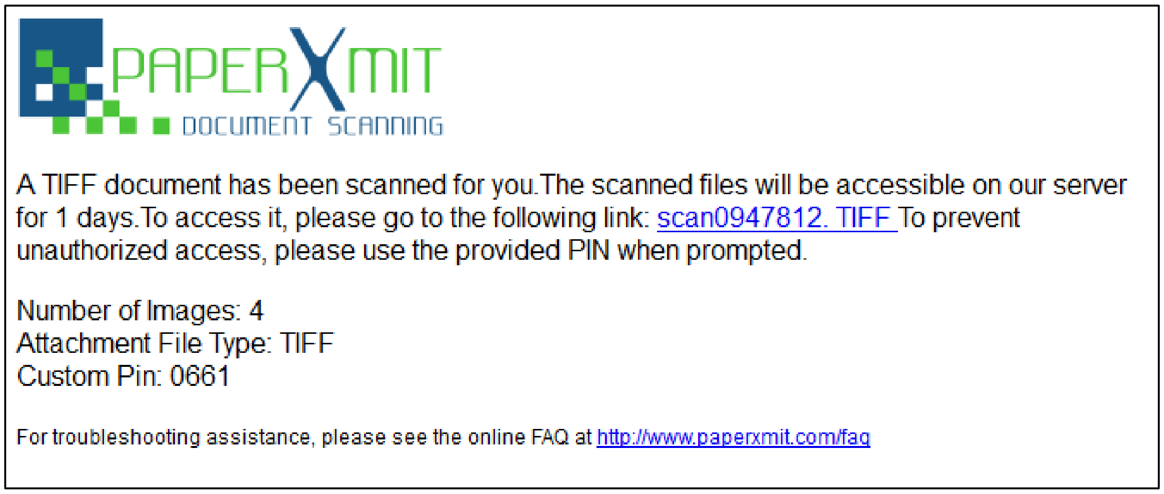
Sold by: N/A
Condition: Open-box/Display Item

Billed amount: $2156.99

You're item will be shipped from our storage facility within 4 days. We estimate delivery on: **Jun 2, 2013**(TRACK PACKAGE)

Returns are easy.Visit our Online Return Center

Figure 31 Order confirmation e-mail generated by PhishGen

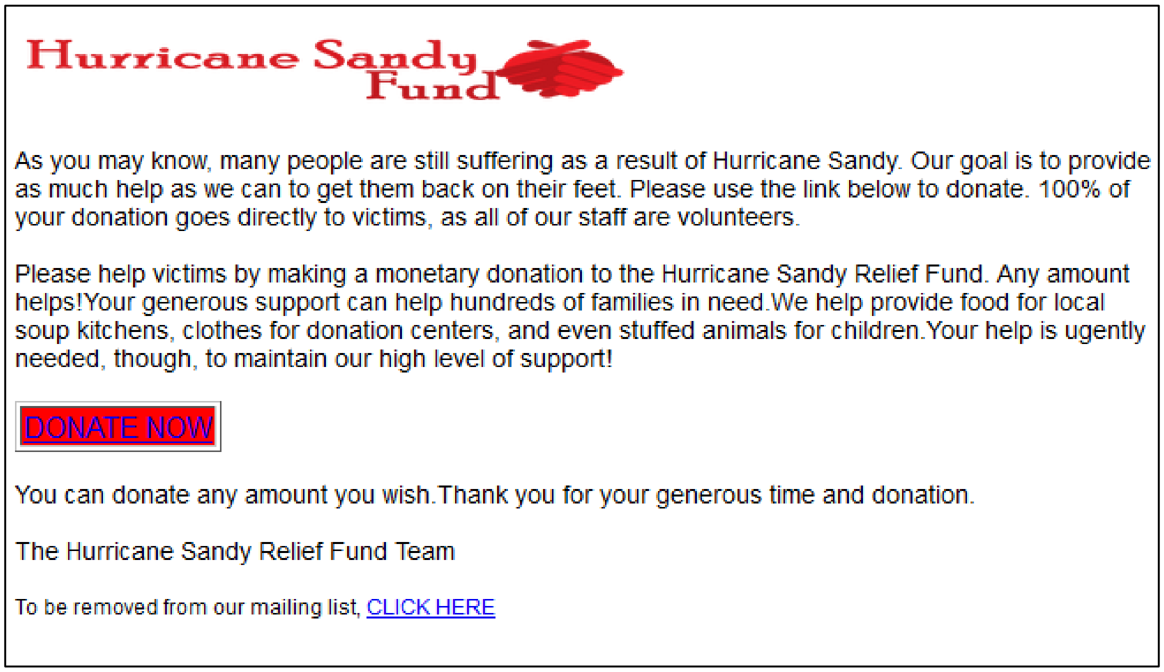

PAPERXMIT
DOCUMENT SCANNING

A TIFF document has been scanned for you.The scanned files will be accessible on our server for 1 days.To access it, please go to the following link: scan0947812. TIFF To prevent unauthorized access, please use the provided PIN when prompted.

Number of Images: 4
Attachment File Type: TIFF
Custom Pin: 0661

For troubleshooting assistance, please see the online FAQ at http://www.paperxmit.com/faq

Figure 32 Electronic fax e-mail generated by PhishGen



As you may know, many people are still suffering as a result of Hurricane Sandy. Our goal is to provide as much help as we can to get them back on their feet. Please use the link below to donate. 100% of your donation goes directly to victims, as all of our staff are volunteers.

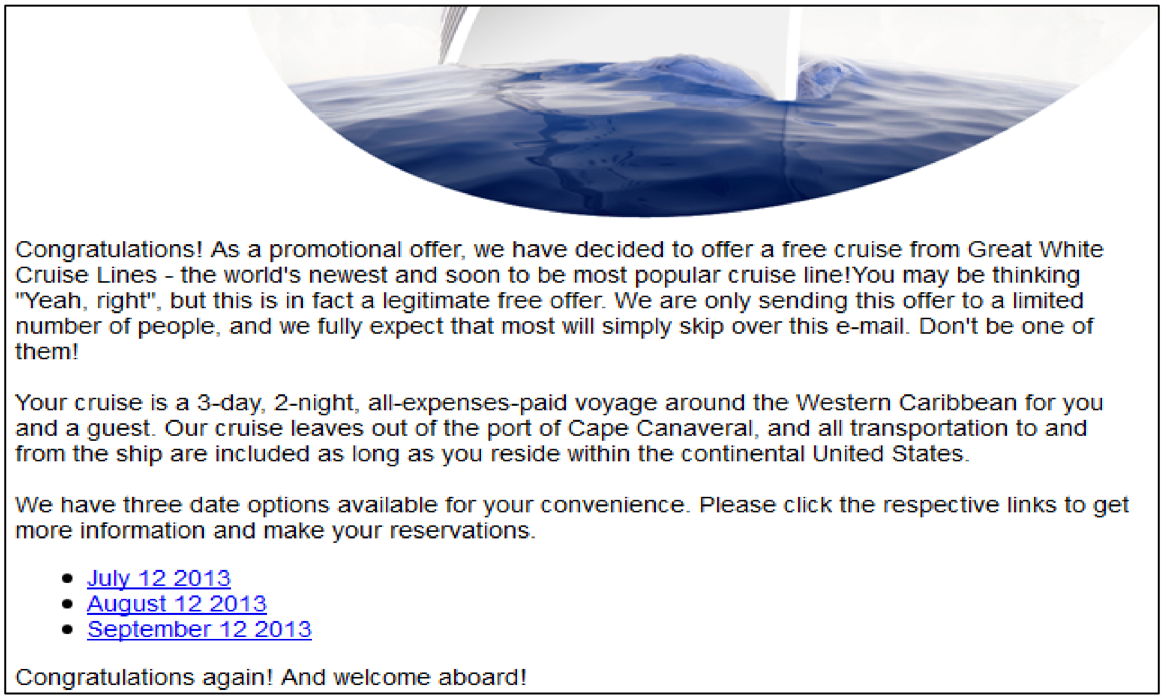
Please help victims by making a monetary donation to the Hurricane Sandy Relief Fund. Any amount helps!Your generous support can help hundreds of families in need.We help provide food for local soup kitchens, clothes for donation centers, and even stuffed animals for children.Your help is ugently needed, though, to maintain our high level of support!

You can donate any amount you wish.Thank you for your generous time and donation.

The Hurricane Sandy Relief Fund Team

To be removed from our mailing list, CLICK HERE

Figure 33 Relief fund e-mail generated by PhishGen



Congratulations! As a promotional offer, we have decided to offer a free cruise from Great White Cruise Lines - the world's newest and soon to be most popular cruise line!You may be thinking "Yeah, right", but this is in fact a legitimate free offer. We are only sending this offer to a limited number of people, and we fully expect that most will simply skip over this e-mail. Don't be one of them!

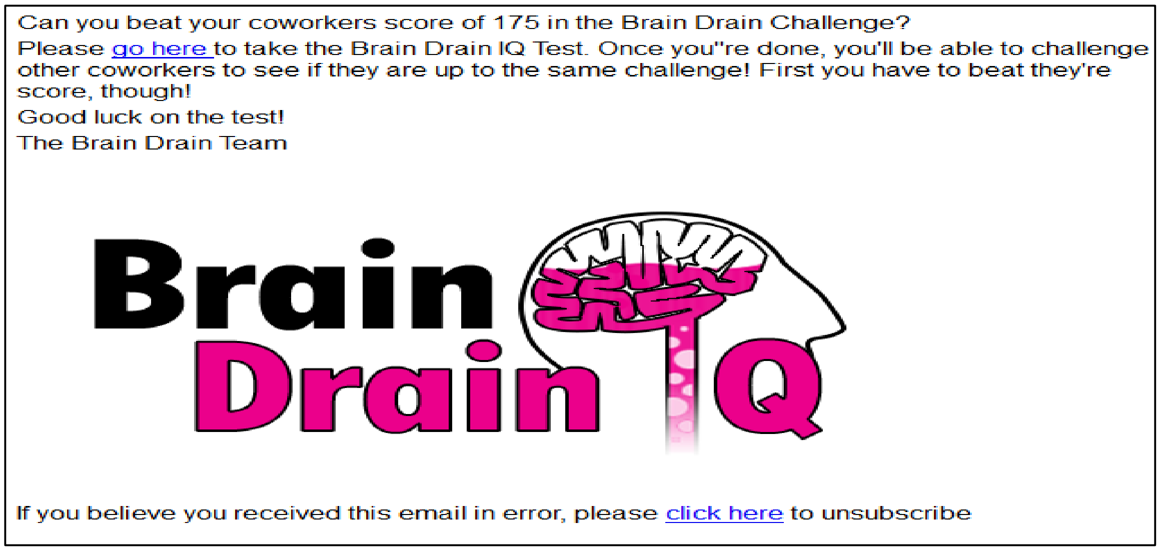
Your cruise is a 3-day, 2-night, all-expenses-paid voyage around the Western Caribbean for you and a guest. Our cruise leaves out of the port of Cape Canaveral, and all transportation to and from the ship are included as long as you reside within the continental United States.

We have three date options available for your convenience. Please click the respective links to get more information and make your reservations.

- July 12 2013
- August 12 2013
- September 12 2013

Congratulations again! And welcome aboard!

Figure 34 Free cruise e-mail generated by PhishGen


Can you beat your coworkers score of 175 in the Brain Drain Challenge?

Please go here to take the Brain Drain IQ Test. Once you''re done, you'll be able to challenge other coworkers to see if they are up to the same challenge! First you have to beat they're score, though!

Good luck on the test!

The Brain Drain Team

If you believe you received this email in error, please click here to unsubscribe

Figure 35 IQ Test e-mail generated by PhishGen

10.2 Experimental Design

To test the hypothesis we developed a comparison study involving 2,253 users chosen at random from a commercial company that had agreed to allow the study, similar to the previous Content and Training Study. The participant list was divided into 7 cohorts by the system owner, and each cohort received four rounds of e-mails. In all, 28 exercises were run, involving 8,786 e-mails. The timing of each round was spaced out over the course of several months, which allowed participants time to return to normal operations after each campaign. Approval was received by the systems owner to test users without notification, and the protocols were also reviewed and approved by the

George Mason Human Subjects Review Board. Internal incident response teams were notified and assisted in gathering data on users that reported the phishing e-mails.

Participants in the study received the simulated phishing e-mails at their normal work address, mixed in with normal e-mail traffic. If a user clicked a link, they were taken to an external server and redirected to an internal training site where the participant received focused training on the specific e-mail they just fell for. For the purposes of these exercises, all e-mails were white-listed to ensure they arrived at the user Inbox.

10.3 Results

Table 12 shows the click-through rates for the study broken out by cohort. In order for this analysis to be relevant, the various cohorts needed to be compared based on the content that was sent for each round, as well as which round the content was used for. For example, it would be unfair to compare the results of a cohort that received an e-mail crafted by an SME with an e-mail generated by PhishGen, were the two e-mails to have drastically different subject matters. As we saw our previous studies, selection bias could easily be used to make our system appear to be more or less effective than a manual approach, however the results would be more a reflection of the users' response to different types of e-mails. Similarly, to compare the same e-mail from different rounds would be equally unfair, as the users in those cohorts would have been through previous exercises and may have been more suspicious of any e-mail. The results of extended exercises, and the apparent decrease in response rates over time, has been noted in multiple other studies as well [57][99]. Table 13 shows the cohorts that were used for comparison for each round, based on the subject matter of the e-mail that was sent.

Table 12 Click-through statistics for Comparison Study

Cohort	*Round*	*E-Mail*	*Generator*	*Responses*	*Sent*	*Click-through Rate*
A	1	Secure E-mail	PhishGen	15	114	13.16%
A	2	Online Order	SME	32	114	28.07%
A	3	Electronic Fax	SME	17	114	14.91%
A	4	Relief Fund	PhishGen	0	113	0.00%
B	1	Secure E-mail	SME	10	74	13.51%
B	2	Online Order	SME	14	74	18.92%
B	3	Electronic Fax	PhishGen	5	72	6.94%
B	4	Free Cruise	SME	0	70	0.00%
C	1	IQ Test	PhishGen	64	492	13.01%
C	2	Free Cruise	SME	31	483	6.42%
C	3	Relief Fund	PhishGen	1	467	0.21%
C	4	Electronic Fax	SME	126	467	26.98%
D	1	Secure E-mail	SME	8	75	10.67%
D	2	Electronic Fax	SME	10	73	13.70%
D	3	Free Cruise	SME	0	71	0.00%
D	4	Online Order	SME	17	69	24.64%
E	1	Secure E-mail	SME	60	499	12.02%
E	2	Online Order	SME	57	493	11.56%
E	3	Electronic Fax	SME	71	482	14.73%
E	4	Relief Fund	SME	1	478	0.21%
F	1	IQ Test	SME	50	499	10.02%
F	2	Free Cruise	SME	17	491	3.46%
F	3	Relief Fund	SME	5	481	1.04%
F	4	Electronic Fax	SME	137	480	28.54%
G	1	Secure E-mail	PhishGen	62	500	12.40%
G	2	Electronic Fax	PhishGen	66	480	13.75%
G	3	Free Cruise	PhishGen	12	481	2.49%
G	4	Online Order	PhishGen	63	480	13.13%

Table 13 Selected cohorts for comparison based on e-mail received in specific rounds of testing

Round	*E-Mail*	*SME Cohorts*	*PhishGen Cohorts*

1	Secure E-mail	B,D,E	A,G
1	IQ Test	F	C
2	Electronic Fax	D	G
3	Electronic Fax	A,E	B
3	Free Cruise	D	G
3	Relief Fund	F	C
4	Relief Fund	E	A
4	Online Order	D	G

10.4 Analysis

To determine whether there was a statistically significant difference between the click-through rates, we used the same two-tailed Z test of 2 population proportions as before. Table 14 shows the click-through rates for the selected cohorts, as well as the z-score and p-value for each of the samples. At $p < 0.01$ the null hypothesis could not be rejected for any of the samples, however at lower levels of confidence the last sample is less conclusive. At $p < 0.05$ our data shows a significant difference in favor of the manually generated e-mail for that sample.

To determined overall performance, we calculated the overall click-through rate from all of the cohorts that were selected for comparison. We wanted to see if either approach was significantly different overall, as opposed to being compared on the basis of a single e-mail. Table 15 shows the total number of e-mails sent and unique responses captured for all SME generated content in the selected cohorts, while Table 16 shows the same data for all PhishGen generated e-mails for the selected cohorts. The overall click-through rate for SME content was 8.5%, with 249 unique responses detected from 2915 generated e-mails. The overall click-through rate for PhishGen content was 9%, with 288 unique responses detected from 3199 generated e-mails.

While cumulative analysis of the response data showed no significant difference between PhishGen and SME generated e-mails, our analysis of the study in Chapter 7 provided an additional analysis tool in the form of normalized gains. Because the analysis may include some level of bias due to the discrepancy in the number of e-mails deployed by each generator type, we used the data from Table 12 to calculate the normalized gain for the two approaches. As a result, we would be able to compare the two approaches without a complexity bias. Figure 36 shows the results of this analysis. We can see the same cumulative decrease that was demonstrated in our previous study, and comparing the two difference approaches we see that the PhishGen approach outperforms the SME approach in the first three rounds. Due to the limited number of rounds, though, we hesitate to claim this data supports that PhishGen consistently outperforms the SME approach. Rather, the data supports the conclusion that PhishGen performs at least as well as the SME approach.

Table 14 Click-through rates and significant difference for selected cohorts

Round	*E-Mail*	*SME*	*PhishGen*	*z-score*	*p-value*
1	Secure E-mail	12.04%	12.54%	-1.0041	0.31732
1	IQ Test	10.02%	13.01%	-1.474	0.14156
2	Electronic Fax	13.70%	13.75%	-0.0119	0.99202
3	Electronic Fax	14.77%	6.94%	1.8107	0.0703
3	Free Cruise	0.00%	2.49%	-1.3456	0.17702
3	Relief Fund	1.04%	0.21%	1.6021	0.1096
4	Relief Fund	0.21%	0.00%	0.4866	0.62414
4	Online Order	24.64%	13.13%	2.5344	0.0114

Table 15 Summary of results for SME generated content in selected cohorts

Round	*E-Mail*	*E-mails Sent*	*Responses Detected*
1	Secure E-mail	648	78
1	IQ Test	499	50
2	Electronic Fax	73	10
3	Electronic Fax	596	88
3	Free Cruise	71	0
3	Relief Fund	481	5
4	Relief Fund	478	1
4	Online Order	69	17
	TOTAL	**2915**	**249**

Table 16 Summary of results for PhishGen generated content in selected cohorts

Round	*E-Mail*	*E-mails Sent*	*Responses Detected*
1	Secure E-mail	614	77
1	IQ Test	492	64
2	Electronic Fax	480	66
3	Electronic Fax	72	5
3	Free Cruise	481	12
3	Relief Fund	467	1
4	Relief Fund	113	0
4	Online Order	480	63
	TOTAL	**3199**	**288**

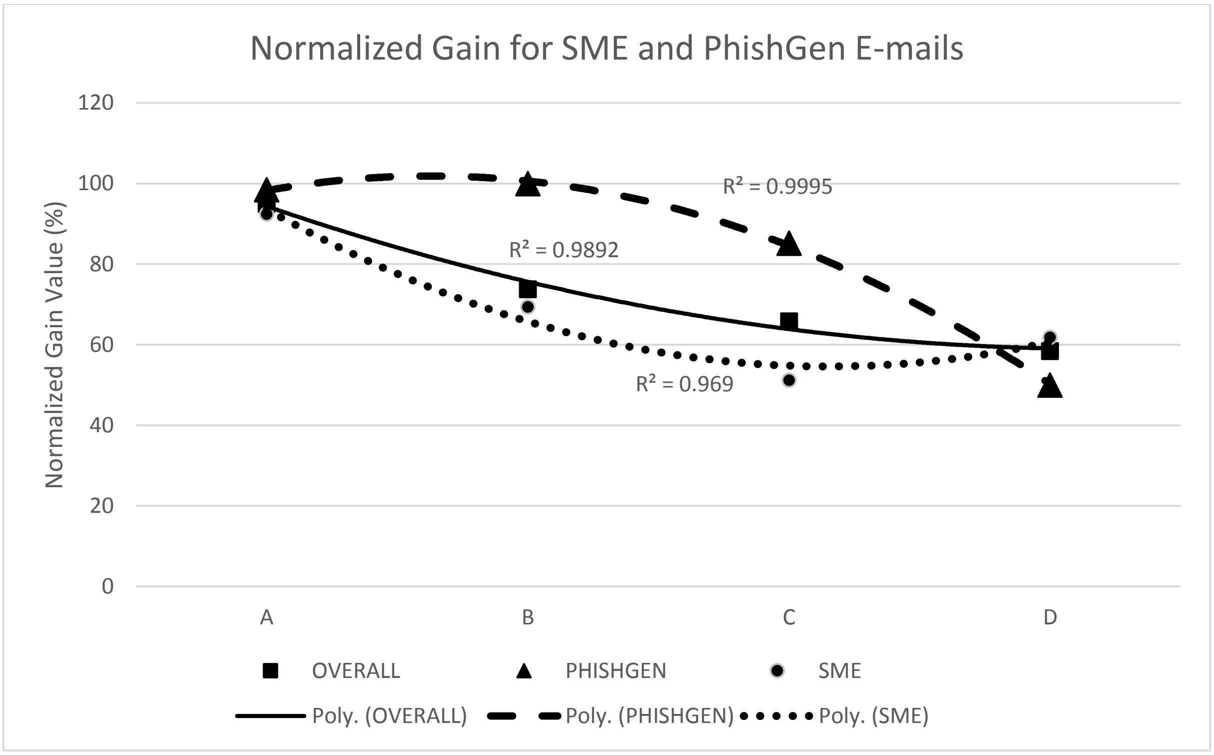


Figure 36 Normalized Gain for SME and PhishGen e-mails

10.5 Conclusions

Our results support the conclusion that PhishGen is able to generate semantically valid and convincing e-mail content that performs as well in live exercises as content manually created by an expert. Our direct comparison of selected cohorts shows no significant difference when comparing the responses to the same e-mail in the same exercise round, while our normalized gains analysis also shows PhishGen to be as effective. This means that the e-mails created by PhishGen, when interpreted by a person, are sufficiently convincing to elicit a response, and so could be used effectively in live exercises.

CHAPTER 11:

IMPROVEMENT OF EXISTING MODELS

Our previous exercises and simulations demonstrated that PhishGen is able to produce dynamic and useful content that can be used in live phishing exercises, however we also needed to show that our approach has novel applications outside of live exercises. For example, we wanted to show that PhishGen can be used to improve existing detection models. The ability to determine specific weaknesses in filters is a major feature of PhishGen, and a similar approach has already been validated by other researchers. In previous research, synonyms and basic obfuscation techniques were used to get spam e-mails passed Bayesian filters [66]. However, our approach has the added benefit of being automated, and so we are able to utilize PhishGen as a fuzzing tool to identify gaps in e-mail filters.

11.1 Filter Evasion and Hardening

During the Content Characteristic simulations, one of the benefits found was that PhishGen is able to learn which production rules are effective in getting past detection measures. More importantly, to accomplish that goal PhishGen keeps track of which chains of production rules may by filtered, and this information can then be analyzed to improve filters.

While Bayesian filters have been shown to be vulnerable to bypass using synonyms or other visual modifications, PhishGen can be used to identify these synonyms in order to more tightly control the filters. Based on feedback, a list of potentially bad production rule combinations are used to overrule decisions within the PCFG when it makes choices between two rules with the same weight. This list of bad production rule combinations is accomplished by doing an n-gram analysis of rule combinations based on positive and negative feedback from the exercise environment. Figure 37 illustrates the process of discriminating good and bad sequences of production rule using n-gram analysis, and updating production rule weights accordingly. When a decision is made to choose one production rule over another, PhishGen looks at the list of potentially bad rule combinations to see if there is an optimal choice, and makes its decision based on that data.

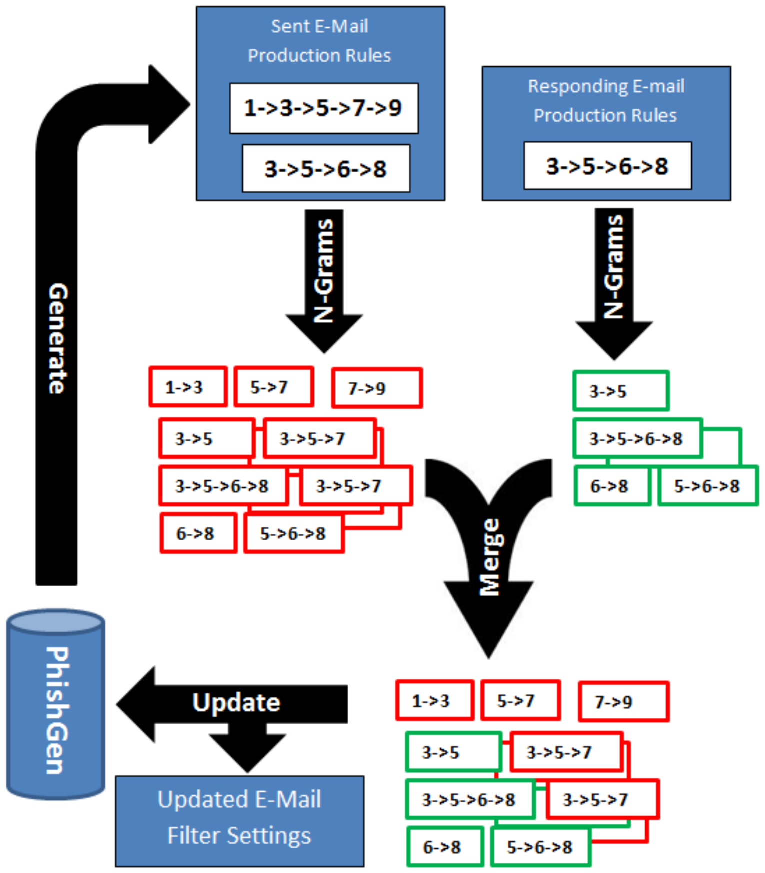


Figure 37 Process overview for using n-gram analysis to update filters and PCFG rule weights

While there is no guarantee that the rule combination is actually bad or being filtered, it could just have been sent to a participant that was on vacation, PhishGen assumes the worst and favors rule combinations that are either known to be good, or at least are not included in the bad combination list. This does affect diversity to some extent, however as has been shown by our previous simulations, the system still has greater diversity in the sets of generated e-mails than traditional methods. There can be no guarantee that filters would be able to catch everything, however we do show that PhishGen can be used to identify coverage gaps for hardening existing filters [12].

11.2 Hypothesis

Our initial hypothesis is that the adjusting of PCFG production rules combined with feedback from testing environments can be used to identify specific use cases that are not detected by the model being evaluated. As all production rules used to create an e-mail are recorded and tracked, the same process can be used to identify production rules that should be added to existing detection frameworks.

11.3 Experimental Design

To test the fuzzing capability of PhishGen, we developed a series of simulations that would measure the ability to bypasses existing countermeasures. E-mail content for each simulation was generated by PhishGen using a set of production rules that had been previously validated in other research [11]. Specifically, we utilized a template that contained a series of production rules to generate an e-mail asking users to click links in order to update a password or user profile. This e-mail template was chosen because it contained some rules that would generate clearly suspicious content and would presumably trigger existing detection capabilities, but it had also been demonstrated as very effective in previous live exercises. In all, the production rules template contained 129 individual production rules, and took less than an hour to develop.

In many cases, templates that we developed for testing went completely undetected in the various environments that were being tested, and so were excluded from the results that we analyzed. The focus of the simulations was to demonstrate a learning capability that supports fuzzing, which is best illustrated when e-mails are initially detected.

For each simulation, PhishGen was first used to create 100 random e-mails, and these e-mails were then sent through to the specific environment being evaluated. All e-mails for all rounds of testing were generated from the same set of production rules for consistency. Information about the emails that were not filtered was then fed back into PhishGen so that it could adjust the weights on production rules and update the n-gram analysis tables for the next round of testing. Each round of simulations utilized 4 sets of 100 e-mails, with a total of 3,200 generated e-mails being sent overall.

The first environment that we used for testing our fuzzing approach was a corporate environment. Specifically, we were allowed to run simulations within the production e-mail environment for a large commercial firm, given that our simulations were not targeting users. This environment supported business operations for over 20,000 users, and was configured with multiple phishing and spam detection measures. This environment primarily utilized a signature-based approach to detection, but also incorporated commercial solutions like Cisco IronPort spam filtering. There were likely additional detection measures in place that we were unaware of, however this illustrates how our fuzzing approach can still be used in black-box environments. For our e-mail client, we used the same configuration used by the company user population, which was a Microsoft Outlook 2013 installation with McAfee E-mail Security installed as an add-in. We did modify the base configuration to have the Junk E-mail Options set to high, so that we could provide as restrictive an environment as possible. This environment was representative of the types of e-mail environments we have seen at other companies.

The second environment utilized SpamAssassin with Bayesian-style probabilistic classification, as an alternative to a signature-based approach. SpamAssassin was trained on 795,092 spam and phishing e-mails sourced from a spam codex maintained by Untroubled Software (http://untroubled.org/spam/) and a scam e-mails database maintained by Scamdex (http://www.scamdex.com).

Initially, we were interested in testing our approach against online e-mail systems like Gmail. However, after testing several types of e-mails and production rules, we determined that these environments were not useful for demonstrating our fuzzing approach because they failed to identify any of our generated e-mails as suspicious. For example, the example e-mails that were detected by SpamAssassin and the corporate e-mail environment were not filtered when sent to a Gmail account. We believe the most important reason for this is the lack of exposure. Gmail looks for viruses and malware signatures, patterns within e-mails, and most importantly it learns from user-reported e-mails [100]. For example, Gmail will provide warnings for messages with similarity to known suspicious messages, detected spoofed sender addresses, known phishing attacks, and even empty message contents. However, Gmail relies heavily on reports of phishing and spam by users, and our e-mails were never sent in the wild. Our e-mails were not sent with malicious attachments or virus signatures, but surprisingly they did not meet any thresholds that would classify the e-mails as suspicious either. While this does not mean Gmail cannot detect any e-mail generated by PhishGen, it does indicate the difficulty in identifying "zero-day" e-mails in general.

After all simulations were completed, we then provided an updated rule to try and catch generated e-mails, in order to determine whether the n-gram analysis could be useful for mitigation. We then resent a set of 100 e-mails through to the client to identify whether the mitigations was successful.

11.4 Results

For our testing we had PhishGen working off of perfect information from the exercise environment. Any e-mail that made it to the Inbox without modification was considered "successful" and provided as feedback to the software. In a live environment, not every e-mail would result in a human response, and so the software would have less information per round of e-mails on which to base weight adjustments. Also, PhishGen would assume that a non-response was due to filtering, rather than through failure to coerce a human response. This is a unique capability of PhishGen, because in a live environment PhishGen will ultimately favor production rules that trigger human responses as well. For the purposes of our research to show how PhishGen can be used as an intelligent fuzzer, though, we removed the human variable from the simulations.

Table 17 shows the percentage of detected e-mails for both environments across multiple simulations. In all, we ran four sets of simulations in each environment, with each simulation utilizing 4 rounds of 100 e-mails. In both environments, PhishGen was able to get 100\% of e-mails through the filters within 3 rounds of e-mails being sent. This indicates that the n-gram analysis approach was able to identify production rules that were resulting in e-mails being filtered, which then informed the subsequent rounds of e-mail generation.

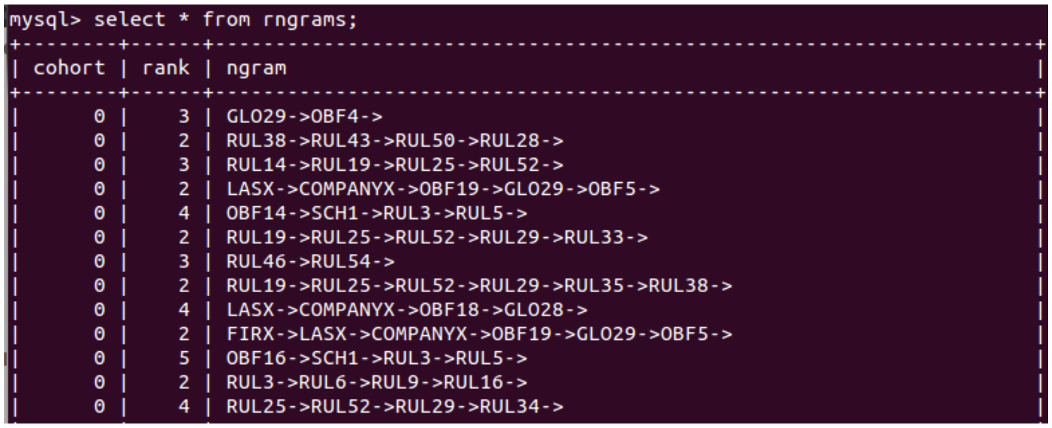
Table 17 Detection rates for multiple simulations across two test environments

	Simulation Round			
Environment	*1*	*2*	*3*	*4*
Corporate	19%	12%	0%	0%
Corporate	14%	0%	0%	0%
Corporate	18%	1%	0%	0%
Corporate	15%	1%	1%	0%
SpamAssassin	15%	1%	0%	0%
SpamAssassin	9%	1%	0%	0%
SpamAssassin	10%	0%	0%	0%
SpamAssassin	20%	1%	0%	0%

11.5 Analysis and Conclusions

Overall, there was very little difference in detection capabilities between the SpamAssassin and Corporate test environments. Both appeared to catch around 15% of the e-mails in the first round, with a significant drop in detection for the second round. In one simulation, the Corporate environment was able to catch more than 10% of the second round e-mails, but then the detection rate dropped to zero in the third round of e-mails. While this only occurred once, it does show that the approach can sometimes misinterpret the feedback from the exercise environment, or employ previously unselected production rules that result in filtering.

PhishGen keeps track of every generated e-mail, and the series of production rules used to generate them. Figure 38 shows a listing from the database that tracks the various n-grams and associated ranks for production rules that were caught by filters. With this information, we isolated several sets of production rules that generated successful e-mails, and added those sentence fragments to existing filtering rules within the local e-mail client. With these new filtering rules in place, all of the e-mails that were previously successful were caught by the filters and moved to the Junk E-Mail folder.



```
mysql> select * from rngrams;
+--------+------+-------------------------------------------------------------------+
| cohort | rank | ngram                                                             |
+--------+------+-------------------------------------------------------------------+
|      0 |    3 | GLO29->OBF4->                                                     |
|      0 |    2 | RUL38->RUL43->RUL50->RUL28->                                      |
|      0 |    3 | RUL14->RUL19->RUL25->RUL52->                                      |
|      0 |    2 | LASX->COMPANYX->OBF19->GLO29->OBF5->                              |
|      0 |    4 | OBF14->SCH1->RUL3->RUL5->                                         |
|      0 |    2 | RUL19->RUL25->RUL52->RUL29->RUL33->                               |
|      0 |    3 | RUL46->RUL54->                                                    |
|      0 |    2 | RUL19->RUL25->RUL52->RUL29->RUL35->RUL38->                        |
|      0 |    4 | LASX->COMPANYX->OBF18->GLO28->                                    |
|      0 |    2 | FIRX->LASX->COMPANYX->OBF19->GLO29->OBF5->                        |
|      0 |    5 | OBF16->SCH1->RUL3->RUL5->                                         |
|      0 |    2 | RUL3->RUL6->RUL9->RUL16->                                         |
|      0 |    4 | RUL25->RUL52->RUL29->RUL34->                                      |
```

Figure 38 PhishGen tracks specific good and bad n-grams which can be used to update filters

Our results support the claim that the feedback from successful responses can be used by PhishGen to adjust production rules and create more effective content. However the result also show that this approach is effective in identifying gaps in detection filters, which can then be used to create more effective rules upon analysis. The same approach can be used to modify global filtering rules, rather than focusing on local filters in the e-mail client.

An interesting aspect of our approach is that many elements in the generated e-mails are randomized, and will change each time they are used. This is beneficial for generating test cases for training algorithms against. Since the dataset will be unique each time it is generated, the algorithm design phase can go through multiple testing rounds on new data, ensuring that the system is not simply focusing on detecting static strings.

CHAPTER 12:

CONCLUSIONS AND FUTURE WORK

This thesis seeks to improve existing phishing detection and prevention methods by providing a novel method of supplementing existing phishing content that can be used to build training datasets and live exercise content. To support this objective, we developed PhishGen, a novel approach to content generation using generative grammars. At the beginning of this dissertation, we presented the following thesis statement:

Generative grammars, when used in conjunction with a means to ensure semantic consistency, can be used to create large sets of diverse, realistic and effective phishing e-mails that can supplement training datasets and live phishing exercises.

We next describe how we support this thesis by addressing the fundamental contributions of this research.

12.1 Fundamental Contributions Overview

The fundamental contributions of this dissertation include best practices for live phishing exercises, empirical data regarding the impact of content on click-through rates, and novel approaches to content development using generative grammars. The following sections summarize how these contributions support our thesis claim.

12.1.1 Live Phishing Exercise Capabilities and Best Practices

When properly designed, live phishing exercises provide an excellent way to evaluate the impact of different variables on phishing response rates. In our case, we were able to establish an effective process for coordinating these exercises, and by coordinating closely with the system owners we were able to gather reliable data to support our claims. Our research utilized several live phishing exercises, collecting response data from thousands of individuals over the course of several months.

In addition to validating our live exercise methodology, our research also supported the claim that PhishGen is able to generate large datasets of test cases. Over the course of our research, we were required to generate a large amount of test cases. The number of test cases that were used in live exercises and simulations numbered in the tens of thousands, although this number is actually much larger given the test cases generated in development, and provided as examples to system owners in order to demonstrate capabilities.

12.1.2 Content Impact Analysis Results

Several of our studies focused in the impact of e-mail content on click-through rates. In the first motivational study, we utilized different types of e-mails with varying semantic complexity to show that the type or complexity of content in a phishing e-mail is important. We concluded that content is an extremely relevant factor in the effectiveness of phishing e-mails, and that arbitrary or random content does not perform as well. We also conclude based on our results from the second motivational study that reusing the same content, even if it is initially very effective, has decreased utility over time. Our larger study confirmed that variations in content complexity can result in significant differences in click-through rates, and that training received in between rounds of testing also affects response characteristics.

12.1.3 Semantic Consistency Engine

After demonstrating the importance of content, we also demonstrated that our system is able to generate content that is able to perform as well as content created by a subject matter expert using existing methods. Combined with the results of our previous studies indicating lower click-through rates for less complex or random content, we can conclude that the content generated by PhishGen is roughly equivalent in complexity to content crafted by traditional methods.

12.1.4 Diverse Dataset Generation

While our live phishing exercises demonstrated that our content could perform as well as manually created content, our e-mail characteristic simulations also confirmed that our approach was able to generate larger datasets that maintained better overall characteristics with regards to diversity than existing methods. We conclude then that our approach is able to generate dynamic content, and can be used to create datasets for training purposes.

12.1.5 Model Improvement Using Iterative Feedback Testing

With our comparison study, we demonstrated that our generated content is able to coerce responses from actual users, however we also set out to provide a method to improve existing models that are used for phishing e-mail detection. Based on the results of our fuzzing approach, we conclude that PhishGen is able to provide dynamic content that can be used to improve existing filter settings. The same iterative process could easily be implemented into algorithm training phases to provide an increased level of difficulty, forcing more robust algorithms.

12.2 Future Work

Several additional areas of research resulted from this project, as well as areas where we feel additional work needs to be performed to improve the utility of PhishGen. These include additional applications for filter testing, as well as additional ideas for phishing exercises to improve results analysis. Some of these additional research areas are provided here.

12.2.1 Codex Generation

While the web interface in PhishGen is primarily built for handling live exercises and simulations, PhishGen also supports codex generation. The same process used for importing templates and modifying rules can be used via the web interface, but instead of going through the online generation process the system has a local script for creating batches of test cases.

When generating a set of offline test cases, we see the major benefits of using CFGs. Even for a small CFG, the growth can be phenomenal, which provides the capability of generating millions of potential test cases. By comparison, the Enron corpus contains 88792 messages [101]. Instead of using a single example from a known phishing e-mail, which could lead to overly tight filters that would not detect variations in the attack, PhishGen can instead build test cases that use combinations of various techniques to ensure that detection algorithms are focused on the underlying features that are important for detection.

An additional area of research that we feel might be useful is to go through and generate a massive library of test cases, as well as supporting grammar rules, so that researchers can simply download a set of test cases and use them as training datasets. Additionally, this could be supported by creating a publically accessible PhishGen instance that could safely import new rules from external sources and supplement a central library of test cases.

12.2.2 Additional Phishing Exercises

While the live exercises we performed provided very useful information about response characteristics, we feel that a more extensive study might provide more interesting data. Given the mercurial nature of exercise participants, it would be interesting to extend the same exercise methodologies over a larger number of exercise rounds, or to other organizations with different user demographics.

12.2.3 Improved Features and Grammar Rule Formats

The current implementation of PhishGen was not built for aesthetics, and we understand that this type of rapid prototyping results in a non-intuitive user interface. One of our future projects is to spend some time on the interface and deployment processes to make PhishGen easier to use, and easier to install in various environments.

Additionally, we feel that our format and encoding scheme for the various production rules could be improved. We found that the format of rules in our system can get very complicated, especially when trying to maintain context between variables that reference other variables, or relational variables that are embedded in other relational variables. The more complicated this encoding becomes, the more arduous it is for an exercise designer to create valid templates. We would like to make content creation more efficient, not less efficient, and so our web interface offloaded some of the more complex formatting issues onto the system rather than the user. We decided that the structure we had was sufficient for generating useful content, given the results of our study, however formalizing an encoding structure could be useful for future iterations.

12.3 Final Remarks

It is unlikely that there will ever be a 100% effective mitigation to phishing attacks, but luckily a complete solution is not always necessary. Instead, network defenders only need to make it sufficiently improbable that an attack would succeed. To that end, any improvement is welcome, and we hope that PhishGen can be useful to network defenders by providing an additional tool to support phishing mitigation efforts and increase the effectiveness of detection capabilities.

REFERENCES

[1] N. D. Schwartz and C. Drew, "RSA Security Faces Angry Users Over Breach," *The New York Times*, 07-Jun-2011.

[2] "New York Times hacked, Syrian Electronic Army suspected - NBC News.com." [Online]. Available: http://www.nbcnews.com/tech/internet/new-york-times-hacked-syrian-electronic-army-suspected-f8C11016739. [Accessed: 07-Sep-2014].

[3] B. P. 13, 2014, and 12:21 Pm, "Target breach may have started with email phishing." [Online]. Available: http://www.cbsnews.com/news/target-breach-may-have-started-with-email-phishing/. [Accessed: 12-Sep-2014].

[4] "Email Attack on Vendor Set Up Breach at Target — Krebs on Security." [Online]. Available: http://krebsonsecurity.com/2014/02/email-attack-on-vendor-set-up-breach-at-target/. [Accessed: 07-Sep-2015].

[5] "Hacks of OPM databases compromised 22.1 million people, federal authorities say - The Washington Post." [Online]. Available: http://www.washingtonpost.com/blogs/federal-eye/wp/2015/07/09/hack-of-security-clearance-system-affected-21-5-million-people-federal-authorities-say/. [Accessed: 07-Sep-2015].

[6] "OPM Breach Dates Back to December | Threatpost | The first stop for security news." [Online]. Available: https://threatpost.com/opm-breach-dates-back-to-december/113361/. [Accessed: 07-Sep-2015].

[7] "Official: Russia eyed in Joint Chiefs email intrusion - CNNPolitics.com." [Online]. Available: http://www.cnn.com/2015/08/05/politics/joint-staff-email-hack-vulnerability/. [Accessed: 07-Sep-2015].

[8] "Russia believed to be behind Pentagon's Joint Staff email breach - CBS News." [Online]. Available: http://www.cbsnews.com/news/russia-believed-to-be-behind-pentagons-joint-staff-email-breach/. [Accessed: 07-Sep-2015].

[9] "View All Publications." [Online]. Available: http://www.mcafee.com/us/apps/view-all/publications.aspx. [Accessed: 12-Sep-2014].

[10] "Phishing continues to be effective, McAfee Labs report shows," *TechWorm*. .

[11] S. Palka and D. McCoy, "Dynamic phishing content using generative grammars," in *Software Testing, Verification and Validation Workshops (ICSTW), 2015 IEEE Eighth International Conference on*, 2015, pp. 1–8.

[12] S. Palka and D. McCoy, "Fuzzing E-mail Filters with Generative Grammars and N-Gram Analysis."

[13] K. Rekouche, "Early phishing," *ArXiv Prepr. ArXiv11064692*, 2011.

[14] J. Felix and C. Hauck, "System security: a hacker's perspective," *1987 Interex Proc.*, vol. 1, no. 6, 1987.

[15] "Version 3 Unix mail man page." [Online]. Available: http://minnie.tuhs.org/cgi-bin/utree.pl?file=V3/man/man1/mail.1. [Accessed: 05-Jul-2014].

[16] V. Sobeslav, "Computer networking and sociotechnical threats," in *Proc. of the International Conference on Applied, Numerical and Computational Mathematics–ICANCM'11, and Proc. of the International Conference on Computers, Digital Communications and Computing–ICDCC*, 2011, vol. 11, pp. 75–79.

[17] D. Irani, S. Webb, J. Giffin, and C. Pu, "Evolutionary study of phishing," in *eCrime Researchers Summit, 2008*, 2008, pp. 1–10.

[18] R. Dazeley, J. L. Yearwood, B. H. Kang, and A. V. Kelarev, "Consensus clustering and supervised classification for profiling phishing emails in internet commerce security," in *Knowledge Management and Acquisition for Smart Systems and Services*, Springer, 2010, pp. 235–246.

[19] R. Dhamija, J. D. Tygar, and M. Hearst, "Why phishing works," in *Proceedings of the SIGCHI conference on Human Factors in computing systems*, 2006, pp. 581–590.

[20] C. Jackson, D. R. Simon, D. S. Tan, and A. Barth, "An evaluation of extended validation and picture-in-picture phishing attacks," in *Financial Cryptography and Data Security*, Springer, 2007, pp. 281–293.

[21] M. Jakobsson, "Modeling and preventing phishing attacks," in *Financial Cryptography*, 2005, vol. 5.

[22] H. Kalidasu, B. P. Kumar, and K. A. Kumar, "Battle against for Phishing Based on Captcha Security," *Int. J. Innov. Res. Dev.*, vol. 1, no. 5, pp. 497–519, 2012.

[23] A. Almomani, B. B. Gupta, T. Wan, A. Altaher, and S. Manickam, "Phishing Dynamic Evolving Neural Fuzzy Framework for Online Detection Zero-day Phishing Email," *ArXiv Prepr. ArXiv13020629*, 2013.

[24] A. Bergholz, J. De Beer, S. Glahn, M.-F. Moens, G. Paaß, and S. Strobel, "New filtering approaches for phishing email," *J. Comput. Secur.*, vol. 18, no. 1, pp. 7–35, 2010.

[25] A. Bergholz, J. H. Chang, G. Paaß, F. Reichartz, and S. Strobel, "Improved Phishing Detection using Model-Based Features.," in *CEAS*, 2008.

[26] A. Bergholz, G. Paaß, L. D'Addona, and D. Dato, "A real-life study in phishing detection," in *Proceedings of the Conference on Email and Anti-Spam (CEAS)*, 2010, vol. 1, pp. 1–10.

[27] I. Fette, N. Sadeh, and A. Tomasic, "Learning to detect phishing emails," in *Proceedings of the 16th international conference on World Wide Web*, 2007, pp. 649–656.

[28] H. Huang, L. Qian, and Y. Wang, "A SVM-based technique to detect phishing URLs," *Inf. Technol. J.*, vol. 11, no. 7, pp. 921–925, 2012.

[29] F. Toolan and J. Carthy, "Feature selection for spam and phishing detection," in *eCrime Researchers Summit (eCrime), 2010*, 2010, pp. 1–12.

[30] R. Cohen and A. Mergi, *Method for Detecting and Blocking Phishing Attacks*. Google Patents, 2010.

[31] M. He, S.-J. Horng, P. Fan, M. K. Khan, R.-S. Run, J.-L. Lai, R.-J. Chen, and A. Sutanto, "An efficient phishing webpage detector," *Expert Syst. Appl.*, vol. 38, no. 10, pp. 12018–12027, 2011.

[32] H. Jiang, D. Zhang, and Z. Yan, "A Classification Model for Detection of Chinese Phishing E-Business Websites," 2013.

[33] G. Liu, B. Qiu, and L. Wenyin, "Automatic detection of phishing target from phishing webpage," in *Pattern Recognition (ICPR), 2010 20th International Conference on*, 2010, pp. 4153–4156.

[34] M. Aburrous, M. A. Hossain, K. Dahal, and F. Thabtah, "Predicting phishing websites using classification mining techniques with experimental case studies," in *Information Technology: New Generations (ITNG), 2010 Seventh International Conference on*, 2010, pp. 176–181.

[35] M. I. A. Ajlouni, W. 'el Hadi, and J. Alwedyan, "Detecting Phishing Websites Using Associative Classification," *Eur. J. Bus. Manag.*, vol. 5, no. 15, pp. 36–40, 2013.

[36] P. Soni, S. Firake, and B. B. Meshram, "A phishing analysis of web based systems," in *Proceedings of the 2011 International Conference on Communication, Computing & Security*, 2011, pp. 527–530.

[37] V. Ramanathan and H. Wechsler, "Phishing website detection using Latent Dirichlet Allocation and AdaBoost," in *Intelligence and Security Informatics (ISI), 2012 IEEE International Conference on*, 2012, pp. 102–107.

[38] R. Dhamija and J. D. Tygar, "The battle against phishing: Dynamic security skins," in *Proceedings of the 2005 symposium on Usable privacy and security*, 2005, pp. 77–88.

[39] C.-Y. Huang, S.-P. Ma, W.-L. Yeh, C.-Y. Lin, and C.-T. Liu, "Mitigate web phishing using site signatures," in *TENCON 2010-2010 IEEE Region 10 Conference*, 2010, pp. 803–808.

[40] B. Parno, C. Kuo, and A. Perrig, *Phoolproof phishing prevention*. Springer, 2006.

[41] S. Egelman, L. F. Cranor, and J. Hong, "You've been warned: an empirical study of the effectiveness of web browser phishing warnings," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2008, pp. 1065–1074.

[42] M. Jakobsson and S. Myers, *Phishing and countermeasures: understanding the increasing problem of electronic identity theft*. Wiley. com, 2006.

[43] A. Almomani, B. Gupta, S. Atawneh, A. Meulenberg, and E. Almomani, "A Survey of Phishing Email Filtering Techniques."

[44] Y. Zhang, S. Egelman, L. Cranor, and J. Hong, "Phinding phish: Evaluating anti-phishing tools," 2006.

[45] T. Ronda, S. Saroiu, and A. Wolman, "Itrustpage: a user-assisted anti-phishing tool," in *ACM SIGOPS Operating Systems Review*, 2008, vol. 42, pp. 261–272.

[46] J. S. Downs, M. B. Holbrook, and L. F. Cranor, "Decision strategies and susceptibility to phishing," in *Proceedings of the second symposium on Usable privacy and security*, 2006, pp. 79–90.

[47] S. Sheng, M. Holbrook, P. Kumaraguru, L. F. Cranor, and J. Downs, "Who falls for phish?: a demographic analysis of phishing susceptibility and effectiveness of

interventions," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2010, pp. 373–382.

[48] A. Vishwanath, T. Herath, R. Chen, J. Wang, and H. R. Rao, "Why do people get phished? Testing individual differences in phishing vulnerability within an integrated, information processing model," *Decis. Support Syst.*, vol. 51, no. 3, pp. 576–586, 2011.

[49] R. T. Wright and K. Marett, "The influence of experiential and dispositional factors in phishing: An empirical investigation of the deceived," *J. Manag. Inf. Syst.*, vol. 27, no. 1, pp. 273–303, 2010.

[50] C. Laorden, B. Sanz, I. Santos, P. Galán-García, and P. G. Bringas, "Collective classification for spam filtering," *Log. J. IGPL*, p. jzs030, 2012.

[51] N. A. G. Arachchilage, S. Love, and M. Scott, "Designing a Mobile Game to Teach Conceptual Knowledge of Avoiding 'Phishing Attacks,'" *Int. J. E-Learn. Secur.*, vol. 2, no. 2, pp. 127–132, 2012.

[52] C. B. Mayhorn and P. G. Nyeste, "Training users to counteract phishing," *Work J. Prev. Assess. Rehabil.*, vol. 41, pp. 3549–3552, 2012.

[53] S. Sheng, B. Magnien, P. Kumaraguru, A. Acquisti, L. F. Cranor, J. Hong, and E. Nunge, "Anti-phishing phil: the design and evaluation of a game that teaches people not to fall for phish," in *Proceedings of the 3rd symposium on Usable privacy and security*, 2007, pp. 88–99.

[54] "Patent US8793799 - Systems and methods for identifying and mitigating information security risks - Google Patents." [Online]. Available: http://www.google.com/patents/US8793799. [Accessed: 23-Sep-2014].

[55] P. Kumaraguru, J. Cranshaw, A. Acquisti, L. Cranor, J. Hong, M. A. Blair, and T. Pham, "School of phish: a real-world evaluation of anti-phishing training," in *Proceedings of the 5th Symposium on Usable Privacy and Security*, 2009, p. 3.

[56] P. Kumaraguru, Y. Rhee, A. Acquisti, L. F. Cranor, J. Hong, and E. Nunge, "Protecting people from phishing: the design and evaluation of an embedded training email system," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, 2007, pp. 905–914.

[57] D. Bliton, A. Norwood, and S. Palka, "Unannounced Phishing Exercises and Targeted Training: Results and Lessons Learned," in *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)*, 2011, vol. 2011.

[58] "Home | Phishing Training | Security Awareness | PhishMe, Inc.," *PhishMe*. [Online]. Available: http://phishme.com/. [Accessed: 19-Aug-2013].

[59] M. Jakobsson and J. Ratkiewicz, "Designing ethical phishing experiments: a study of (ROT13) rOnl query features," in *Proceedings of the 15th international conference on World Wide Web*, 2006, pp. 513–522.

[60] G. Ollmann, "The evolution of commercial malware development kits and colour-by-numbers custom malware," *Comput. Fraud Secur.*, vol. 2008, no. 9, pp. 4–7, 2008.

[61] N. Pavkovic and L. Perkov, "Social Engineering Toolkit—A systematic approach to social engineering," in *MIPRO, 2011 Proceedings of the 34th International Convention*, 2011, pp. 1485–1489.

[62] D. Maynor, *Metasploit Toolkit for Penetration Testing, Exploit Development, and Vulnerability Research*. Elsevier, 2011.

[63] "pentestgeek/phishing-frenzy," *GitHub*. [Online]. Available: https://github.com/pentestgeek/phishing-frenzy. [Accessed: 12-Jul-2014].

[64] M. Motoyama, D. McCoy, K. Levchenko, S. Savage, and G. M. Voelker, "An analysis of underground forums," in *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, 2011, pp. 71–80.

[65] M. Yip, "An investigation into Chinese cybercrime and the underground economy in comparison with the West," University of Southampton, 2010.

[66] C. Karlberger, G. Bayler, C. Kruegel, and E. Kirda, "Exploiting redundancy in natural language to penetrate Bayesian spam filters," in *First USENIX Workshop on Offensive Technologies (WOOT'07), Boston, MA*, 2007.

[67] "Gone phishing: Army uses Thrift Savings Plan in fake e-mail to test cybersecurity awareness - The Washington Post." [Online]. Available: http://www.washingtonpost.com/politics/gone-phishing-army-uses-thrift-savings-plan-in-fake-email-to-test-cybersecurity-awareness/2014/03/13/8ad01b84-a9f3-11e3-b61e-8051b8b52d06_story.html. [Accessed: 16-Sep-2014].

[68] "Too Many Phishing Penetration Tests Go Wrong - PhishGuru Product Designed to Avoid Common Pitfalls and Train Employees at the Same Time | Wombat Security." [Online]. Available: http://www.wombatsecurity.com/too-many-phishing-penetration-tests-go-wrong-phishguru-product-designed-to-avoid-common-pitfalls-and-train-employees-at-the-same-time. [Accessed: 23-Sep-2014].

[69] "Security test prompts federal fraud alert | Network World." [Online]. Available: http://www.networkworld.com/article/2247922/security/security-test-prompts-federal-fraud-alert.html. [Accessed: 23-Sep-2014].

[70] P. Finn and M. Jakobsson, "Designing ethical phishing experiments," *Technol. Soc. Mag. IEEE*, vol. 26, no. 1, pp. 46–58, 2007.

[71] C. Yue and H. Wang, "Anti-phishing in offense and defense," in *Computer Security Applications Conference, 2008. ACSAC 2008. Annual*, 2008, pp. 345–354.

[72] M. Jakobsson, "The human factor in phishing," *Priv. Secur. Consum. Inf.*, vol. 7, pp. 1–19, 2007.

[73] T. Dimkov, A. Van Cleeff, W. Pieters, and P. Hartel, "Two methodologies for physical penetration testing using social engineering," in *Proceedings of the 26th Annual Computer Security Applications Conference*, 2010, pp. 399–408.

[74] T. N. Jagatic, N. A. Johnson, M. Jakobsson, and F. Menczer, "Social phishing," *Commun. ACM*, vol. 50, no. 10, pp. 94–100, 2007.

[75] J. G. Mohebzada, A. El Zarka, A. H. BHojani, and A. Darwish, "Phishing in a university community: Two large scale phishing experiments," in *Innovations in Information Technology (IIT), 2012 International Conference on*, 2012, pp. 249–254.

[76] P. Kumaraguru, S. Sheng, A. Acquisti, L. F. Cranor, and J. Hong, "Lessons from a real world evaluation of anti-phishing training," in *eCrime Researchers Summit, 2008*, 2008, pp. 1–12.

[77] M. Jakobsson, P. Finn, and N. Johnson, "Why and how to perform fraud experiments," *Secur. Priv. IEEE*, vol. 6, no. 2, pp. 66–68, 2008.

[78] “US Air Force phishing test transforms into a problem | Network World.” [Online]. Available: http://www.networkworld.com/article/2208344/data-center/us-air-force-phishing-test-transforms-into-a-problem.html. [Accessed: 23-Sep-2014].

[79] “Cyber Security Training, Simulated Attacks, and Phishing Filters | Wombat Security.” [Online]. Available: http://wombatsecurity.com/. [Accessed: 19-Aug-2013].

[80] V. P. Coletta, J. A. Phillips, and J. J. Steinert, “Interpreting force concept inventory scores: Normalized gain and SAT scores,” *Phys. Rev. Spec. Top.-Phys. Educ. Res.*, vol. 3, no. 1, p. 010106, 2007.

[81] R. R. Hake, “Interactive-engagement versus traditional methods: A six-thousand-student survey of mechanics test data for introductory physics courses,” *Am. J. Phys.*, vol. 66, no. 1, pp. 64–74, 1998.

[82] E. Blom and S. de Korte, “Dummy auxiliaries in child and adult second language acquisition of Dutch,” *Lingua*, vol. 121, no. 5, pp. 906–919, 2011.

[83] A. Radford and H. Yokota, “On the Acquisition of Universal and Parameterised Goal Accessibility Constraints by Japanese Learners of English,” *Essex Res. Rep. Linguist. Mar*, 2011.

[84] L. Dominguez, M. J. Arche, and F. Myles, “Testing the predictions of the feature-assembly hypothesis: evidence from the L2 acquisition of Spanish aspect morphology,” in *Proceedings of the Boston University Conference on Language Development*, 2011, vol. 35.

[85] J. Stribling, M. Krohn, and D. Aguayo, *Scigen-an automatic cs paper generator*. 2005.

[86] C. Labbé and D. Labbé, “Duplicate and fake publications in the scientific literature: how many SCIgen papers in computer science?,” *Scientometrics*, vol. 94, no. 1, pp. 379–396, 2013.

[87] A. W. Aho and J. D. Ullman, *Introduction to automata theory, languages and computation*. Addison-Wesley, Reading, MA, 1979.

[88] N. Chomsky and M. P. Schützenberger, “The algebraic theory of context-free languages,” *Stud. Log. Found. Math.*, vol. 26, pp. 118–161, 1959.

[89] M. R. Bridson and R. H. Gilman, “Context-free languages of sub-exponential growth,” *J. Comput. Syst. Sci.*, vol. 64, no. 2, pp. 308–310, 2002.

[90] “Context-free grammar,” *Wikipedia, the free encyclopedia*. 13-Jul-2014.

[91] M. Sipser, *Introduction to the Theory of Computation*. Cengage Learning, 2012.

[92] J. Yearwood, M. Mammadov, and A. Banerjee, “Profiling phishing emails based on hyperlink information,” in *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on*, 2010, pp. 120–127.

[93] A. Blum, B. Wardman, T. Solorio, and G. Warner, “Lexical feature based phishing URL detection using online learning,” in *Proceedings of the 3rd ACM workshop on Artificial intelligence and security*, 2010, pp. 54–60.

[94] S. Chhabra, A. Aggarwal, F. Benevenuto, and P. Kumaraguru, “Phi. sh/$ oCiaL: the phishing landscape through short URLs,” in *Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference*, 2011, pp. 92–101.

[95] “ShmooCon 2009- Presentations.” [Online]. Available: https://www.shmoocon.org/2009/presentations-all.html. [Accessed: 19-Jul-2014].
[96] “Automated Spear-twishing - It was only a matter of time Hack3rcon 3 (Hacking Illustrated Series InfoSec Tutorial Videos).” [Online]. Available: http://www.irongeek.com/i.php?page=videos/hack3rcon3/05-automated-spear-twishing-it-was-only-a-matter-of-time-sean-palka. [Accessed: 19-Jul-2014].
[97] “Cisco Anti-Spam - Cisco.” [Online]. Available: http://www.cisco.com/c/en/us/products/security/email-security-appliance/antispam_index.html. [Accessed: 19-Jul-2014].
[98] “Change the level of protection in the Junk Email Filter - Outlook.” [Online]. Available: http://office.microsoft.com/en-us/outlook-help/change-the-level-of-protection-in-the-junk-email-filter-HP010356454.aspx. [Accessed: 19-Jul-2014].
[99] B. M. Bowen, R. Devarajan, and S. Stolfo, “Measuring the human factor of cyber security,” in *Technologies for Homeland Security (HST), 2011 IEEE International Conference on*, 2011, pp. 230–235.
[100] “Spam and suspicious emails - Gmail Help.” [Online]. Available: https://support.google.com/mail/answer/1366858?hl=en. [Accessed: 30-Jun-2015].
[101] “Data - Csmining Group.” [Online]. Available: http://www.csmining.org/index.php/data.html. [Accessed: 19-May-2014].

BIOGRAPHY

Sean Palka graduated from Westwood High School, Austin, TX, in 1994. He received his Bachelor of Arts from the University of Notre Dame in 1998, and his Master of Science from Marymount University in 2001. For the past 10 years, Mr. Palka has been employed as a penetration tester, and has developed multiple tools to assist in the effective execution of social engineering exercises including phishing. He is a co-inventor of the patented Booz Allen Hamilton STAR*Phish tool (US8793799).