8-5-2020

A Reputation Score Driven E-Mail Mitigation System

Tommy Chin
txc9627@g.rit.edu

A Reputation Score Driven E-Mail Mitigation System

by

Tommy Chin

Submitted to the
B. Thomas Golisano College of Computing and Information Sciences
Department of Computing Security
in partial fulfillment of the requirements for the
Master of Science Degree in
Computer Security and Information Assurance
at the Rochester Institute of Technology

August 5, 2020

Abstract

E-mail inspection and mitigation systems are necessary in today's world due to frequent bombardment of adversarial attacks leverage phishing techniques. The process and accuracy in identifying a phishing attack present significant challenges due to data encryption hindering the ability to conduct signature matching, context analysis of a message, and synchronization of alerts in distributed detection systems. The author recognizes a grand challenge that the increase in the number of data analysis systems corresponds to an overall increase in the delivery time delay of an e-mail message. This work enhances PhishLimiter as a solution to combat phishing attacks using machine learning techniques to analyze 27 e-mail features and Software-Defined Networking (SDN) to optimize network transactions. PhishLimiter uses a two-lane inspection approach of Store-and-Forward (SF) and Forward-and-Inspect (FI) to distinguish whether traffic is held for analysis or immediately forwarded to the destination. The results of the work demonstrated PhishLimiter as a viable solution to combat Phishing attacks while minimizing delivery time of e-mail messages.

M.S. IN COMPUTER SECURITY AND INFORMATION ASSURANCE

ROCHESTER INSTITUTE OF TECHNOLOGY

ROCHESTER, NEW YORK

CERTIFICATE OF APPROVAL

MS DEGREE THESIS

The MS degree thesis of Tommy Chin
has been examined and approved by the
thesis committee as satisfactory for the
thesis required for the
MS degree in Computer Security and Information Assurance

Sumita Mishra, Thesis Advisor

Yin Pan

Kaiqi Xiong

Date

THESIS RELEASE PERMISSION
ROCHESTER INSTITUTE OF TECHNOLOGY
GCCIS M.S. COMPUTER SECURITY AND INFORMATION ASSURANCE

Title of Thesis:
A Reputation Score Driven E-Mail Mitigation System

Acknowledgments

I would like to express my sincere appreciation to Dr. Kaiqi Xiong for his guidance, support, and mentorship. I would also like to thank Ms. Sylvia Perez-Hardy, Dr. Bo Yuan, Dr. Sumita Mishra, Dr. Yin Pan, Ms. Liz Zimmerman, and Ms. Megan Fritts for supporting my graduate studies and providing guidance. Lastly, I would like to thank Dr. Eric Blasch, Dr. Guna Seetharaman, and Dr. Soundararajan Ezekiel for their mentorship.

In dedication to all my friends and family who supported me throughout this endeavor

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Electronic Mail or E-Mail may be considered the norm for a daily regiment where users often and periodically read and review their inboxes for a ingest of correspondence and messages. There are many types of messages a user may receive regarding online bills, newsletters, advertisements, and spam. phishing is a technique often leveraged by threat actors. A user receives an e-mail that consists of a specially crafted message that contains either a malicious link/attachment or a luring statement. The primary objective of a phishing attack is to reveal some form of sensitive data such as Personal Identifiable Information (PII), user credentials, or to entice the recipient to commit an action such as divulge intellectual properties.

In 2019, Verizon reported that phishing attacks impacted 83% of the global community and were the results of 32% data breaches [40]. Several organizations have evaluated methods and techniques to deter phishing attempts from the education approach of Security Education Training and Awareness (SETA) [17] programs in addition to security-driven detection appliances. The technology capabilities to deter phishing attacks and related hazardous e-mails are often mitigated by an inspection and mitigation system such as, but not limited to, Proofpoint [34], PhishLimiter [10], and PhishNet [35]. A concern for many e-mail mitigation systems often focuses on performance drawbacks on receiving and sending a message. Precisely, an e-mail inspection system often holds a message for analysis before forwarding its content to the final destination once all signatures and heuristic-based detection complete the analysis. The time a system analyzes e-mail messages creates an impact for both the recipient(s) and the organization's business operations by delaying the delivery of an e-mail. The complexity of the e-mail message (e.g. web-code in e-mail body, file attachment, or message encryption) increases the inspection delay.

This thesis presents an extended approach to PhishLimiter [10] as an enhancement to address the urgency of e-mail delivery while providing a robust e-mail inspection system to deter phishing attacks. PhishLimter provides a two-lane inspection approach to detect and mitigate phishing attacks known as Store-and-Forward (SF) and Forward-and-Inspect (FI). The SF lane carefully inspects the e-mail communication by storing the message in transit on a virtualized network switch before allowing it to arrive at the recipient's inbox. The FI addresses the urgency of timely e-mail communication by quickly forwarding the message in transit to the recipient while creating a copy of the communication for analysis in a future period. A reputation score of the e-mail communication dictates the lane of inspection where characteristics and features increase or decrease the value. Machine learning is the critical component to PhishLimiter, where 27 types of features are extracted per e-mail message ranging from source and destination e-mail address to file attachment analysis.

The contributions of this research is as follows:

- Enhanced the capabilities and study of PhishLimiter by adding eight new features for detecting phishing attacks to support newer adversarial attack methodologies
- Measured inspection timing and delay of the system using five new datasets with over 500,000 messages for analysis
- Conduct performance analysis of the system through implementation of load balancing

The remainder of this thesis is organized as follows: Chapter 2 describes background information to both phishing and SDN. Chapter 3 describes the architecture design of the system and methodology of conducting experiments while Chapter 4 showcases real-world results. Chapter 5 provides discussion while Chapter 6 concludes the work.

Chapter 2

Background, Challenges, and Assumptions

This chapter describes background information into the areas of phishing and SDN.

2.1 Phishing Indicators, Capabilities, and Controls

Adversaries often leverage e-mail communication as one of many entry points that ultimately end in the total compromise of an enterprise. The MITRE ATT&CK framework outlines e-mail communication as 3 out of 11 (27%) methods for Initial Access (i.e., Spearphishing Attachment, Spearphishing Link, and Spearphishing via Service) [29]. APWG identified 162,155 new phishing websites for Q4 of 2019 and that the majority (30.80%) of phishing targets are in SaaS/Webmail [1]. Organizations adopt and integrate several controls to deter, reduce, or mitigate phishing attacks, summarized into three technical categories: Infrastructure, Application, and User. Table 2.1 outlines three areas to implement phishing detection and migration efforts. Lastly, a top-level view of an organization may include policy-driven controls and governance; however, this thesis focuses directly on the technical works of phishing detection and mitigation efforts.

2.1.1 Phishing Indicators

The most prevalent and common form of Phishing attacks appears in e-mail communication [1, 14]. There are numerous methods and characteristics that

Table 2.1: Comparative review of recent phishing mitigation technologies

Category	**Definition**	**Challenges & Limitation**
Infrastructure [4, 10]	Resides within a network and evaluates or inspects relevant packets related to phishing attacks and e-mail communication	Difficult to evaluate encrypted data without an ability to decrypt messages and presents areas of concern for network congestion and through degradation.
Application [25]	Local to a program or system that interfaces with the end-user to deter or mitigate phishing e-mail and related correspondence	Mitigation limited to endpoint security solutions such as antivirus, anti-malware, and data loss prevention.
User [15, 42]	Training, awareness, and knowledge transfer methods to improve the overall end-users' ability to identify and mitigate phishing attacks when network and application controls fail	Poised to judgement of determining the validity of e-mail communication when all other controls and defeated. Limited by security education and awareness programs.

indicate the purpose of an e-mail message (e.g., normal user communication or malicious intentions). Identifying phishing e-mail communication summarizes into two primary categories: (1) Signature Detection using static signature matching and (2) Dynamic analysis leveraging techniques from machine learning and other automated testing capabilities [10].

Signature Detection

Phishing campaigns often target two groups of individuals: (1) wide mass e-mail communication where an adversary sends a message to any known user and (2) tailored and specifically targeted individuals. The quantity of each phishing victim varies between the adversary's intent, the overall objective, and the level of covertness for a phishing campaign. The first aspects of signature detection for e-mail communication offers a series of benefits to provide known and common phishing attacks. Table 2.2 outlines example areas of an e-mail message where signature detection applies.

Machine Learning

String matching and Deep Packet Inspection (DPI) offers the ability to identify known and frequent phishing attacks but lack the ability to detect new and emerging threats from sophisticated attackers [10, 11, 26]. Machine learning offers the ability to analyze e-mail communication through detailed analysis such as lexical features [4], predictive blacklisting [35], and keyword analysis on web code [16]. Several challenges emerge from leveraging string matching techniques for scenarios where an adversary implements evasive techniques such as data encoding and obfuscation [39] where detection systems are unable to identify and match the phishing attack.

2.1.2 Adversarial Capabilities and Mitigation Controls

A successful phishing campaign leads to several outcomes where a threat actor often achieves their primary objective or further their progress (i.e., persistent foothold [29]) to achieving the end goal—the following present two cases of outcomes from Phishing attacks.

Data Loss

A common result of a successful phishing campaign focuses on data loss. User credentials, trade secrets, and Personal Identifiable Information (PII) are some of the many outcomes from a successful phishing campaign. Additional layers of protection or compensating controls may be included in an environment to protect computer systems from data loss. Some technologies for data loss prevention include context-aware cloud [31] and text-classification [24] approaches. One challenge to the overall approaches lacks the ability to adapt the analysis of data encryption where an adversary may ultimately exfiltrate data over an encrypted medium (channel).

Browser Exploitation to Code Execution

A highly motivated and sophisticated threat actor would attempt to gain access to the underlying computer system that reviewed a phishing e-mail. Gaining access to the computer system through exploitation techniques often leverages 0-Day or n-day style tools. An outdated web browser presents a potential area for exploitation where the known vulnerability may be leveraged to gain code execution on the underlying computer system. There are several publicly accessible tools to compromise outdated browsers [28], where phishing attacks may be an initial pathway to gain access to a computer system. A URL in an e-mail message presents a significant concern as a susceptible user may haphazardly click on the link, thereby allowing their computer system to browse to a website that hosts exploitation code. In-depth analysis using machine learning on URLs [10] and domain names [11, 26] offers the ability to identify malicious links. User education and training are also necessary as a method to reduce the potential risk of a link being clicked. Arachchilage et al. [2] sampled a set of participants (n=20) in their ability to identify and avoid phishing attacks using a mobile application. Participants improved 28% in identifying suspicious messages after receiving training indicating a partially effective solution.

2.2 Software-Defined Networking Communication

Several researched works [7, 8, 10, 11, 26] for detecting and deterring malicious activities identified effective solutions. A critical component to SDN over traditional networking efforts involves the programmability aspects in which customize development solutions interfaces with the overall environment [20]. SSDN presents the case where modifications of network flows and traffic allow for increased performance through optimization efforts [9]. Leveraging framework approaches [22] allows for an improved security posture and the ability to defend against phishing attacks, thereby protecting the overall organization. This work aims to leverage SDN capabilities to optimize further and deter Phishing attacks.

2.3 Research Challenges

Data encryption is often the primary contender to circumvent and defeat detection systems [10]. Several research [6,10] offers robust approaches to providing a method for deep packet inspection for encrypted communication channels—the following present two challenges in this work.

E-Mail Communication Emulation

There are several datasets [19,27,32,36,37] that offer a vast quantity of e-mail messages ranging from normal, spam, and phishing data types. A challenge in conducting experiments involve involves disseminating messages within each data set, emulating the vast number of clients, and replaying the message in conformance with e-mail communication techniques using Internet Message Access Protocol (IMAP) and Simple Mail Transfer Protocol (SMTP). Moreover, the use of secure communication variants such as IMAPS and SMTPS presents challenges by themselves for inspecting e-mail messages as traditional detection systems do not present the ability to conduct signature matching without the use of certificate pinning. This work describes techniques to emulate e-mail messages and handling secure communication in Sub Sections 3.3.2 and 3.3.3, respectively.

SFBuffer Allocation and Constraints

PhishLimiter leverages a two-lane approach in analyzing and inspecting network traffic. The SF lane for inspection requires a queuing process to hold messages for analysis, depending on the system resource specifications. The inspection system's lifespan may endure a situation where a burst of e-mail messages enter the overall detection system, thereby presenting a queuing process for analysis due to resource constraints. The queue of messages presents challenges for situations of a mailing list; spam messages addressed to a significant number of users, or automated systems providing e-mail-based alerting. The process of holding a message may significantly detriment the ability to effectively conduct analysis and inspection for the presence of high urgency situations. Sub Section 3.1.2 addresses these challenges to support the overall function of PhishLimiter.

2.4 Research Assumptions

This work considers one assumption to address the reality of specialized e-mail communication methods that circumvents the detection capabilities of PhishLimiter.

Pretty Good Privacy

E-mail communication using Pretty Good Privacy (PGP) [21] is trusted and safe as both the sender and recipient have conducted a key exchange to validate one another for secure e-mail communication. Adding a “third key” for e-mail correspondence violates the intention of PGP and that the information is visible for both parties and the organization who manages the detection system. For this case of PGP, either party needs a private key or computer system with the private key compromise for a potential avenue of a Phishing attack.

Table 2.2: Areas of an e-mail message for signature detection

Signature	**Description**	**Challenges**
Sender E-Mail Address	The originating source e-mail address ("from" field) of the message	Adversary may spoof the information to trick or mislead a susceptible user
Subject Line	Short term phrases that elevate the urgency of an e-mail	The presence of HTML code in the subject line may change the appearance such as italic format which may confuse static string matching approaches
Message Body	Content of the message containing HTML code, hyperlinks to potential malicious websites, or a convincing story to trick the user to commit an action as part of a social engineering campaign	Numerous methods and technique are necessary to identify malicious activities. Matching URL presents a challenge where mixture of web code and hyperlink data to purposely evade detection for analysis
File Attachments	An accompanying file that arrives with an e-mail message where the user may subconsciously execute malware or malicious code	Significant variation of files and methods of execution from an executable, office macro, or documents that leverages an exploit. File encryption may create false positives due to misinterpretation of data analysis. Size of file varies between e-mail environment and defined by e-mail server (e.g., file attachment quota)
E-Mail Header	The metadata information of the e-mail message where data such as originating server address and whether particular security features are enabled (i.e., Sender Policy Framework (SPF), and DomainKeys Identified Mail (DKIM))	Corresponding an e-mail message to the originating server may present a challenge for areas of e-mail spoofing. False positives are plausible for cases where an organization utilizes an e-mail gateway or third-party filtering service [34]

Chapter 3

Methodology

Detecting and mitigating phishing attacks rely on three categories of detection: *Infrastructure*, *Application*, and *User*, whereas this thesis primarily focuses on the Infrastructure perspective to address the implementation and effectiveness of PhishLimiter. This chapter describes the overall architecture of PhishLimiter and the techniques to address load balancing, scalability, and inspection of encrypted e-mail communication. Figure 3.1 shows an overview of PhishLimiter's design where (1) a user sends an e-mail, (2) PhishLimiter designates a path for inspection using a *reputation score* (see section 3.1), (3) the message undergoes evaluation for maliciousness, (4) sent to an e-mail server for storage, and finally (5) the recipient retrieves the message.

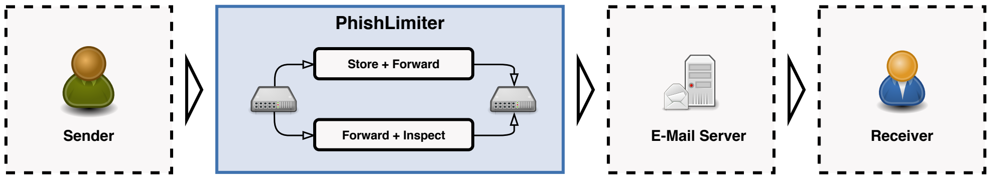


Figure 3.1: A high level overview of PhishLimiter's architecture [10]

3.1 Inspection Lanes

PhishLimiter directs network traffic to traverse through one of two paths for inspection. EEach path adopts separate ideologies and motivations to address an organization's performance demands while maintaining a security level. The Store and Forward (SF) path is the de facto selection when new, never before seen, communication occurs in addition to users who have a low reputation score. The SF path holds messages into a queue within PhishLimiter, inspects the message for maliciousness, and then forwards it to the appropriate destination if deemed safe. The Forward and Inspect (FI) pathway adopts a separate approach where messages are immediately forwarded to the destination upon arrival. At the same time, PhishLimiter generates a copy of the message for future analysis. The FI pathway requires a reputation score that exceeds an *administrative defined threshold* with a *positive rating*. Lastly, both pathways leverage a kernel module to handle and analyze traffic on virtual networking switches.

3.1.1 Virtual Switch Kernel Module

PhishLimiter employs two pathways for inspection. The handling of e-mail communication requires PhishLimiter to hold and replicate messages in transit for SF and FI, respectively. Netfilter [41] provides the ability to process data as it enters a network interface. Figure 3.2 shows an overview of the kernel module with PhishLimiter for data inspection. Step (1) indicates network

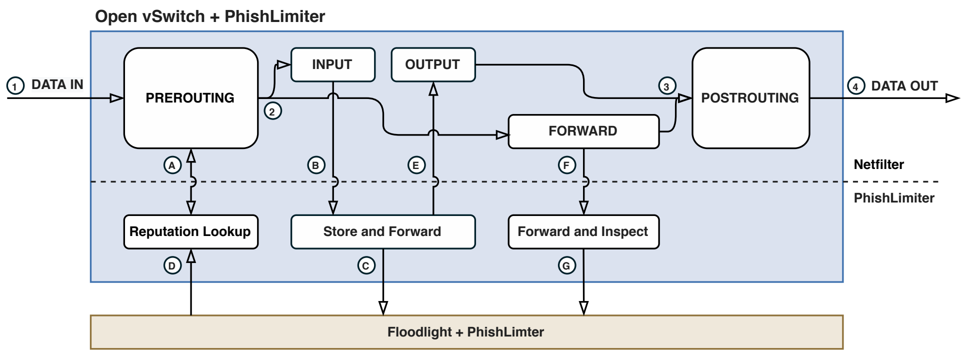


Figure 3.2: An overview of the netfilter module for SF and FI inspection

traffic entering the device for inspection, such as a virtual switch. The traffic consists of both regular network data such as web and e-mail communication (phishing and regular correspondence). A *hook event* (a Netfilter function call to conduct custom processing of information) occurs in Netfilter that queries the reputation of the traffic to determine the path selection, as indicated in Step (2). Traffic with inadequate or lack of a reputation score undergoes inspection in the SF lane while the remaining goes to FI. Step (B) and (C) present the SF process in which a delay or latency impacts the e-mail delivery time due to the step to hold a message. Step (D) denotes the process to analyze and identify the reputation of the traffic where benign (positive) messages proceed to Step (E). PhishLimiter drops messages with poor reputation scores (not shown in the figure). An alternative solution presents the case where an e-mail inspection system sends messages to a quarantine location for user-review. At the same time, this work does not evaluate such a case to focus primarily on the detection accuracy and performance of PhishLimiter. Step (3) shows sending the message beyond the Netfilter module and onward to the destination (4). Under the FI approach of PhishLimiter, Step (F) and (G) represent the process where traffic replication occurs for analysis. At the same time, the module immediately forwards the original message to the destination via Step (3) and (4). A lane selection process determines whether traffic goes to SF or FI as follows.

3.1.2 Lane Selection Process

PhishLimiter depends on a *reputation score* and an *administrative threshold* to determine the pathway selection for network traffic. Feature extraction through machine learning provides insight in determining the reputation or trustworthiness of a network flow. Malicious traffic decreases the overall score of the network flow, while non-malicious increases the overall reputation. The *administrative threshold* indicates the fundamental trigger on whether traffic enters SF or FI.

Reputation Score

The trustworthiness of an e-mail message depends on a combination of user judgment and signature matching. Malicious e-mail communication negatively affects the overall reputation score. The previous work [10] indicated the method to calculate the reputation score while Algorithm 1 provides a brief pseudo code.

Algorithm 1 Pseudo code indicating reputation score calculations

```
$F$: An SDN flow
$pkt$: A network packet of any size
$S_o$: The original score prior to inspection
$S_u$: The updated PhishLimiter Score
$M = ()$: A set of e-mail messages for analysis
$c = 0$: A numerical counter
$E()$: A function to extract payload data from a packet and returns raw text
$P()$: A function for PhishLimiter feature extractions that returns a 1 or -1 for benign or malicious, respectively
procedure Score Analysis($S_{o,i}$,$F_i$)
    for each $pkt$ in $F$ do
        $M_c = M_c + E(pkt)$;
        if $pkt$ = END_OF_STREAM then
            $c = c + 1$
        end if
    end for
    $S_u = S_o$
    for $i = 0$, $i < length(M)$, $i$++ do
        $S_u = S_u + P(M_i)$
    end for
    return $S_u$
end procedure
```

The transaction of network traffic is grouped into SDN flows as designated by F. Multiple flows exist in the environment hence F_i and may contain multiple messages M. The END_OF_STREAM indicates the end of a message or packet *pkt* either by the termination or timeout of a TCP stream. Function $E()$ indicates the extraction of the payload of *pkt* where a partial segment of a message exist (M_c). The entire message undergoes a process of feature extraction where the results indicate the outcome of the maliciousness of the flow. The outcome returns an updated score, thereby indicating the decision of future traffic to go through SF or FI and defined by an administrative threshold.

Administrative Threshold

The underlying decision making or choice to determine whether network traffic traverses through SF or FI depends on the reputation score compared to the administrative threshold. A network flow utilizes the SF lane if the reputation score goes below the administrative threshold. Likewise, positive and normal e-mail communication increases the score to a point where PhishLimiter allows FI's usage. The administrative threshold value varies between network topology, the amount of daily e-mail traffic, and compensating controls if a malicious e-mail arrives at the end user's inbox from the FI lane. The threshold value varies base on the use case as do the upper and lower bounds of the overall score base on the number of false positives generated or performance behaviors in the environment.

Store and Forward (SF)

Storing a message requires system resources on an inspection device to hold the information in a temporary location. The SF lane leverages the `PREROUTING` mechanic in Netfilter to store a message in memory while PhishLimiter processes and analyze the data to determine the maliciousness of an e-mail. The demand and volume of e-mail messages may significantly increase to a point where the resource requirements exceed the amount of memory available. Section 3.2 addresses the scalability and the sudden burst of messages in detail while Figure 3.3 expresses the logic on SF. The architecture of PhishLimiter stores queued e-mails for analysis in the *SFBuffer*, where the resource location operates on the virtual network switch. A concern for the SFBuffer involves the threat of a *buffer overflow attack* as e-mail messages vary in quantity. E-mail messages are often sent using Transmission Control Protocol (TCP) for data reliability and handling. Network transmission throughput is bounded by inter-network devices and the Maximum Transmission Unit (MTU) per system. The constraints of TCP and MTU splits a single e-mail message into smaller segments, thereby, presents a challenge for PhishLimiter to conduct analysis. Precisely, packets arrive into the SFBuffer within the SF lane in real-time, where PhishLimiter reforms the entire message for analysis. A large e-mail message requires packet segmentation due to the restrictions bounded by the MTU and inter-network devices. The arrival of each segment of a message differs from network congestion, packet loss, or issues from packets

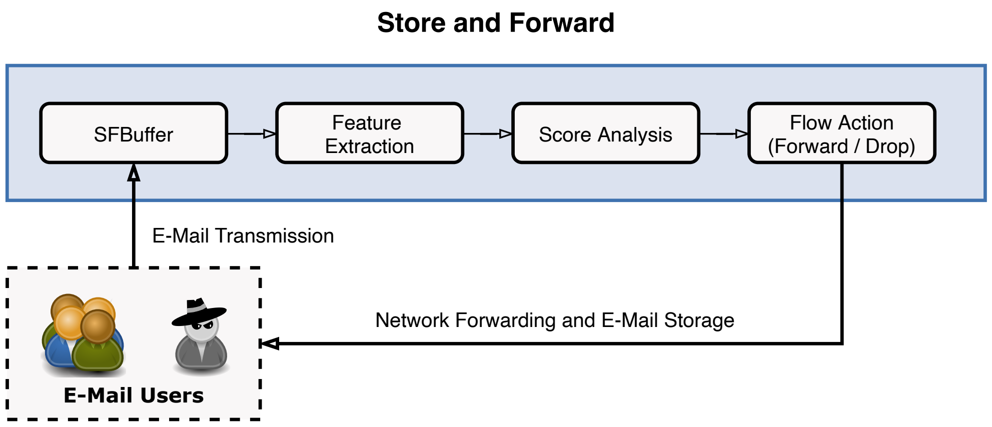


Figure 3.3: Store and Forward (SF) logic diagram

arriving out of order. The SFBuffer factors such situations by holding a message in queue until a timeout occurs by the design of TCP. Analysis of the message occurred upon the completion of the entire transmission and that the SFBuffer concurrently holds messages in real-time. *Feature Extraction* occurs once an e-mail message ultimately arrives into the SFBuffer, where PhishLimiter carefully inspects and analyzes the information for any indicators of malicious activity. The outcome of the analysis either increases or decreases the overall reputation score in *Score Analysis*. The concluding step of SF permits the e-mail to arrive at the recipient or drops the message to protect the user. Positive rated reputation scores allow the user to leverage the FI lane to increase performance on e-mail handling and communication.

Forward and Inspect (FI)

There are certain situations where e-mail messages have a level of urgency that requires minimal delivery time, such as emergency response, executive-level decision making, or time-sensitive collaboration work. The SF lane holds messages in transit for inspection but presents the concern of increase latency when the detection system becomes resource constraint or overburden by events. FI factors a level of trustworthiness for the user by quickly forwarding the message to the destination upon arrival to the inspection system. PhishLimiter replicates the message for analysis at a later time in the `PREROUTING` mechanic of Netfilter. Figure 3.4 shows the logic behind FI. A user transmits an e-mail

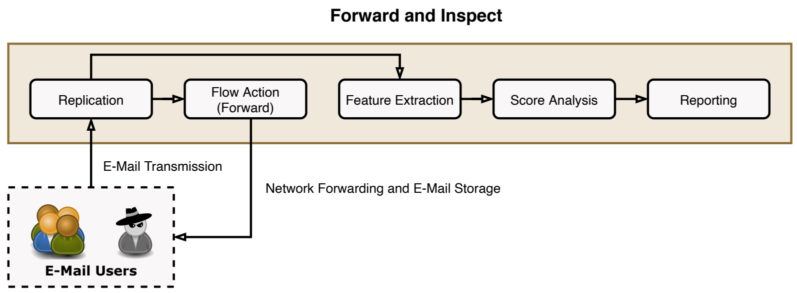


Figure 3.4: Forward and Inspect (FI) logic diagram

message where the reputation of the flow has a positive rating. PhishLimiter replicates and forwards the message upon arrival to the destination when using Forward-and-Inspect (FI). Feature Extraction occurs on the replicated message to determine whether any indications of malicious activities such as malware or exploitation code. The overall reputation score of the network flow decreases when PhishLimiter identifies malicious activities to inspect the data. The comparison between SF and FI presents one area of concern: malicious e-mail messages are automatically forwarded to the destination as the overall trade-off to decrease delivery time. The forwarding of a potentially malicious e-mail message ultimately presents a risk when adopting a scenario of PhishLimiter. At the same time, many endpoint systems utilize compensating controls and technologies (e.g., endpoint security) to protect the user from malware cases.

3.2 Load Balancing at Scale

System resources often limit inspection and detection systems at any scale due to the rate and quantity of network traffic. Many commercial products commonly provide statistics [23] on their detection system (appliance) with a standardized *Packets Per Second (PPS)* value that defines the maximum throughput allowed before a delay occurs. Combating the limitation of systems resources requires a set of distributed computing and load balancing efforts. Load balancing configurations often endure the challenge of synchronization of analysis and alerting, whereas in the case of PhishLimiter—a reputation score must be maintained across all devices. Figure 3.5 shows an example case of three types of load balancing efforts with PhishLimiter. The first

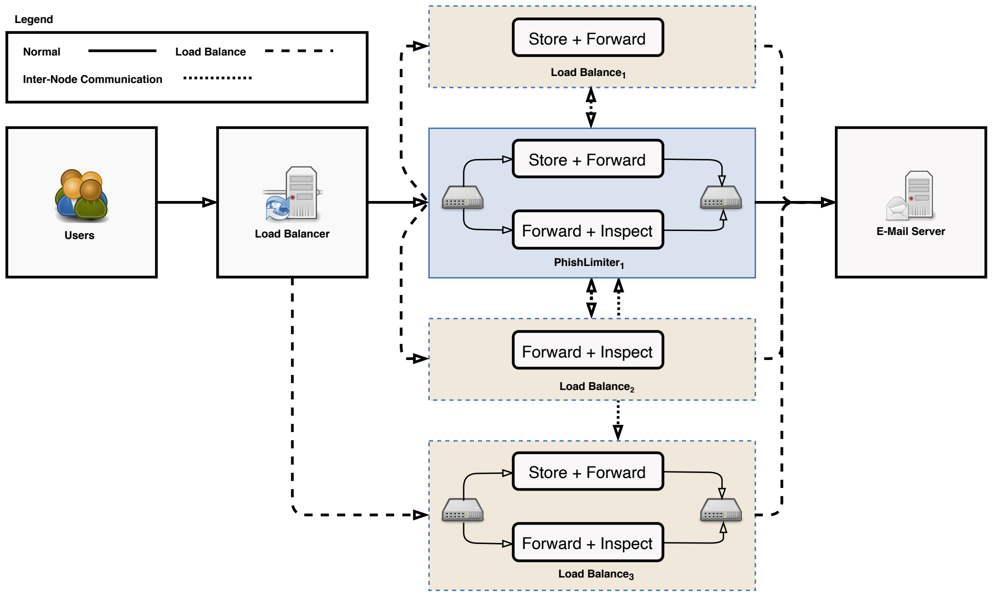


Figure 3.5: A load balancing distribution architecture indicating duplication of SF, FI, and PhishLimiter inspection

example type (`Load Balance`$_1$) presents an initial area of concern with the design of PhishLimiter, where the SF holds a message in memory (SFBuffer) for analysis. The increase or burst of network traffic fundamentally presents the scenario of a queue that increases over time. Adding or increasing the number of SF lanes allows for parallel processing of multiple e-mail messages, minimizing the overall queuing time of a message that would otherwise be held in the SFBuffer.

The second example type (`Load Balance`$_2$) presents the case of multiple FI systems where conditions of burst or a high volume of e-mail messages prolong the overall inspection time. Areas of concern focus on a malicious e-mail message reaching the intended recipient. Severe resource constraints of FI delay the inspection time by factors of several minutes, hours, or indefinitely.

The last example type (`Load Balance`$_3$) depicts the scenario of multiple PhishLimiter instances where the area of concern focuses on network latency and redundancy. A network topology that presents an environment with scalability concerns or high network latency constraints when the topology endures a situation where users send an e-mail message to a centralized mailing service. Users that are the most considerable distance (geographical or network hops) often endure significantly high network latency and potential network loss when communicating to a centralized service. Multiple PhishLimiter instances allow for the deployment of the inspection system in multiple locations of a network, thereby reducing the overall delay gain through network latency. Each three example cases of (`Load Balance`$_{1,2,3}$ presents the overall implementation challenges of PhishLimiter but did not address the concern of handling the result of each inspection lane and centralizing the overall reputation score of e-mail communication in a global scale.

An Inter-Node Communication (INC) pathway allows each instance of a load balancing appliance (PhishLimiter) to report modifications to the overall reputation score for e-mail traffic. The communication effort utilizes the *Control Plane* of the SDN environment to protect the overall reputation score changes of the network flow while leveraging a *client-server* model for multiple devices (e.g., inspection lanes). The selection of the network path and the deployment of a load balancing system differ between network topology/environment. The decision to implement more than one SF or FI depends on system resource constraints on PhishLimiter, network throughput of entering and exiting links, and Quality-of-Service (QoS) policies (e.g., prioritized traffic that has higher urgency or requirement over e-mail traffic).

Selection Process

SDN groups network traffic into “flows”, where one or more e-mail messages may belong to one flow. A load balancer [13] redirects traffic in an optimal method to reduce the overall concern of system resource and network constraints. Leveraging SDN, as the primary method for communication, presents significant advantages to an overall network. However, challenges are presented where multiple e-mail messages arrive under one flow (i.e., mailing list, Phishing, or spam). There are two load balancing (Round Robin and Resource Utilization Selection) for PhishLimiter, as described as follows:

Round-Robin (RR): SDN distinguishes network traffic by flows, and that multiple e-mail messages may originate from one source (e.g., router or e-mail server). The selection of using RR provides simplicity as a network flow naturally increases and decreases in the quantity of traffic over time. Alternating between entry points to PhishLimiter (e.g., SF, FI, or another PhishLimiter instance) simplifies the case with load balancing without the needed overhead to monitor resource utilization on each system frequently. Using RR presents one area of concern where a flow expires when an SDN switch does not observe any traffic after a defined period such that one system becomes overburden while another idles.

Resource Utilization Selection (RUS): The Control plane of SDN allows for the communication of SDN devices to construct flows and manage the overall ecosystem of the network. Load Balancing efforts while monitoring resource utilization of SDN switches and particularly PhishLimiter, allows for a more precise method of balancing network traffic. One area of concern in monitoring resource utilization pertains to the overall overhead gained by querying each networking device and the frequency to obtain the measurements. E-mail communication is often small and sort duration network streams (e.g., flows). The rapid succession of multiple e-mail messages presents the concern of excessive query time and tracking of resource utilization across the network. For example, a user sends an e-mail message where the controller inserts a network flow base upon a series of path selection methods and the decision of the load balancer using RUS. Multiple e-mail messages may require multiple network-flows, thereby requiring a query of each network device for resource utilization on a per-flow basis to define the optimal pathway to support load balancing effort.

3.3 Phishing Features and Extraction

The arrival of an e-mail undergoes a series of feature extraction methods by PhishLimiter to determine the maliciousness of a message. Figure 3.6 provides a brief overview of some of the techniques used in feature extraction while Appendix A outlines the comprehensive metrics.

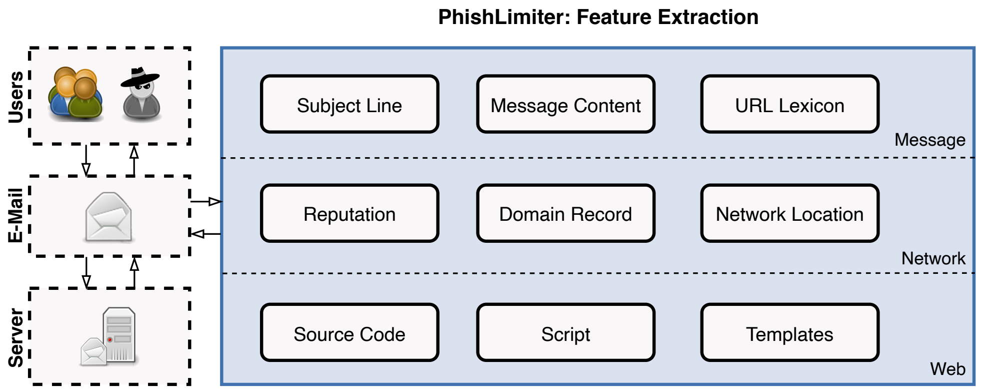


Figure 3.6: Categorization of features for PhishLimiter analysis

3.3.1 Feature Extraction Categories

There are three areas of analysis that PhishLimiter categorizes for feature extraction when determining the maliciousness of an e-mail message. The following provides a brief description of each category, while a comprehensive list of features is given in Appendix A.

Message

The specifics of an e-mail include subject line, message body (content), file attachments, e-mail headers, sender information, and the intended (recipient) location. Each section of an e-mail message comprises of areas where an attacker may leverage to compromise a user or gain valuable information ultimately. The *subject line* of an e-mail message contains a title or topic of the message. A subset of users often use the subject line as a quick method to convey information without a message body and often indicated with the phrase "end of message" or "eom". The *e-mail header* provides indications of metadata or information about the validity of an e-mail message, such as the IP address originating e-mail server to identify spoofing attempts. Lastly, *file attachments* is a ubiquitous method to send and deliver malware to a targeted individual or organization. Analysis of a file attachment determines the overall risk of an e-mail message, whether the attachment is an excel document with a macro, an executable file, or a normal word document.

Network

An e-mail message contains information about the sender (e.g., the author as denoted by the e-mail address in the "from" field), thereby indicating the domain name and the network of origin. The domain name by itself accompanies a set of registration data that provides vital information for analysis. The *whois* record provides data such as the creation and update date of the domain name (e.g., when the domain was first created and the date of last modification for the whois entry), the point-of-contact for the domain name, and registration authority. Many conventional domain reputation systems such as Bluecoat [38] categorizes domain names to groupings such as "Education" or "Shopping." Part of the categorization process involves manual user input through crowdsourcing efforts. A domain name with a history of malicious activities receives a harmful category such as "Phishing" by Bluecoat, thereby preventing organizations from accessing any computer system in that domain. One concern with such a system is the frequency of updating the category as domain names expire over time while the Bluecoat category persists. A popular website, expireddomains.net [18], allows adversaries to harvest and collect reputable domain names where the previous owner failed to renew their subscription or let expire. PhishLimiter aims to solve the concern of reputable domain names being transferred to the ownership of an adversary by analyzing a per e-mail transaction or periodic basis.

Web

Receiving an e-mail message often contains a hyperlink to a web server (either malicious or benign). An analysis of the website further improves the overall detection by validating that the intended page does not indicate potential malicious activities. Examples of areas of concern include source code analysis using signature matching, image analysis (e.g., matching the image of the website), and template matching (e.g., comparing source code to the existing website within an organization).

3.3.2 Phishing Dataset and Training

PhishLimiter's underlying technology leverages machine learning to quickly and decisively determine the maliciousness of an e-mail message. Using a set of sample e-mail messages for training purposes improves the effectiveness and accuracy of the model to optimize the system further. Table 3.3.2 outlines a series of datasets used for training PhishLimiter. Each data set presents a combination of known malicious, spam, or benign e-mail messages but lacks messages with file attachments. Mozilla provides a set of test files [30] commonly used to conduct software fuzzing for web browsers. In contrast, for the case of PhishLimiter–the files present a sample of benign e-mail messages with file attachments for analysis. Normal e-mail communication (non-phishing) is also necessary for analysis as false positives reflect weak indications for a detection system. Figure 3.7 provides an overview of the training process for the model. There are 26 features for analysis as defined by Appendix A. Benign or malicious traffic result in either a +1 (+) or -1 (-), respectively—the outcome of the training results in a set of numerical values that formulate the overall model. Previous work [10] compared detection accuracy amongst numerous machine learning models from Deep Convolution Neural Network to J48 while this study leverages an Artificial Neural Network as the method for purpose of identifying performances with load balancing and data encryption analysis.

Table 3.1: E-Mail dataset for machine learning training

#	Name	Description	Challenges
01	Enron [19]	A 1.7GB repository of e-mail messages between 150 employees	E-mail messages are formated in x.y.z format
02	Mail-Archive [27]	A collection of over two decades of e-mail communication and mailing list	A website host the dataset/-content and require web scraping and recursive download of messages
03	SpamAssassin [37]	A combination of multiple sets of e-mail messages containing spam and ham (non-spam) messages. Repository stores e-mail messages in individual files	Each e-mail message are stored in individual text files and marked with indicators of spam or ham making the only challenge is a method to replay messages to a destination e-mail server through Phish-Limiter
04	Lingspam [36]	A 62.9MB repository of text files containing a 2,893 e-mail messages	No indications of source or destination e-mail addresses as the text files are purely message body of an e-mail. Additionally, all URLs in the dataset have spaces added into each value such as `http : / / www . creditime . com` in `spmsga141.txt` thereby requiring a matching method to identify links
05	EmailIntent [32]	A repository of 4,649 single expressions or sentences accompanied by an indicator of a Yes or No value designating a benign or malicious message, respectively	No indications of source or destination e-mail addresses as dataset is two text files text files are purely message body of an e-mail. Ground truth provided with Yes/No indicator to support detection accuracy

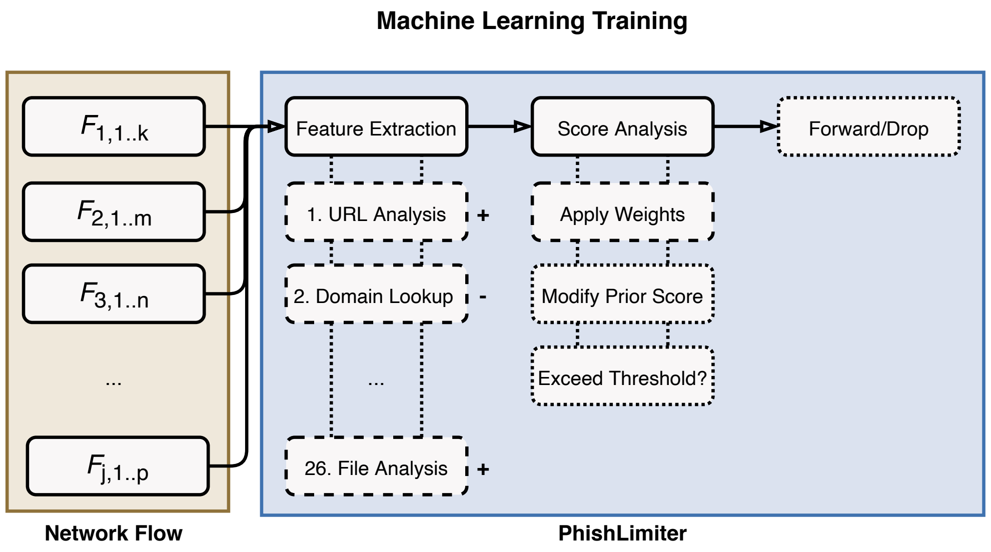


Figure 3.7: A high level depiction of training where the dashed-lined objects are used to define weights for the model and dotted-line represent the implementation of PhishLimiter

Lastly, the experiments used the value of increasing or decreasing the score (e.g., +1 or -1) to simplify the results while PhishLimiter allows for a more significant number to modify the score.

3.3.3 Encrypted Communication Analysis

Network communication that leverages encryption such as TLS presents a significant challenge as detection systems are unable to read the content of the message without a decryption key. E-mail messages and the exchange of e-mail communication often leverage encryption to protect messages from being tampered or modified by an adversary. Certificate Pinning [5] allows an inspection system to decrypt, analyze, and encrypt a message without user interaction. PhishLimiter presents two inspection lanes where the SF lane requires the message encrypted after analysis for delivery to the end-user. In FI, the user has already received a message before inspection, thereby not needing to encrypt the message after analysis. The implementation of certificate pinning requires the receiving e-mail server (destination) to trust a certificate shared with PhishLimiter. Specifically, both the e-mail server and inspection systems share a generated certificate that allows for the analysis of messages over an encrypted channel. The justification for this particular configuration mimics industry implementation of internal certificate authorities. Lastly, this work does not focus on any network devices beyond an e-mail server (e.g., end-user system) as the work primarily focuses on e-mail messages.

3.4 Deployment Strategies

The architectural design of PhishLimiter offers two methods for deployment and testing. One (hybrid) operates within a virtual networking switch, and the other (standalone) uses a separate system to process network traffic. The following describes the advantages of each strategy, while Figure 3.8 provides a high-level overview.

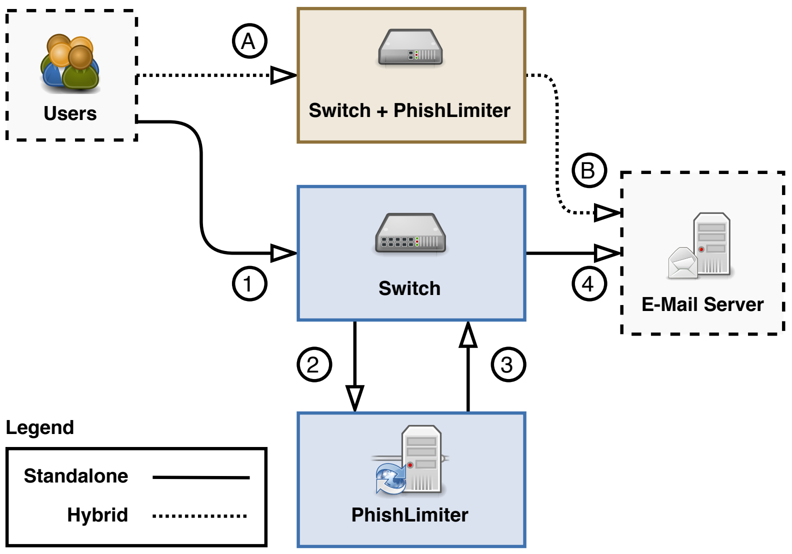


Figure 3.8: A high level overview of deployment strategies for PhishLimiter

Hybrid

PhishLimiter resides within a virtual networking switch where concerns are raised on resource requirements, the analysis of PhishLimiter, and the overall risk for the inspection system (switch). Figure 3.8 shows the example case where Step (A) indicates e-mail messages entering the hybrid model where the unified system endures the constraint of forwarding network traffic and inspecting e-mail messages. Step (B) denotes the hybrid model's output where an e-mail message arrives at an e-mail server. Concerns are raised under the hybrid model where a failure of PhishLimiter or the switching capability causes a total failure of the network path. Advantages to such a method address networks with significantly low throughput rates, high latency, or packet loss.

Standalone

Unlike the hybrid approach, PhishLimiter resides on a separate system from the virtual switching device, requiring additional forwarding of traffic from the switch. Multiple switches may forward traffic to PhishLimiter under this approach, reducing the overall resource requirement of networking switches. Step (2) and (3) denote the forwarding of traffic into and out of PhishLimiter, where additional overhead occurs from network latency. The advantages of such a method allow multiple switches to communicate to a centralized, high-performance system running PhishLimiter. In contrast, system failures within the network minimize the overall impact (e.g., one verse both systems).

3.5 Testing Environment

Testing PhishLimiter required a combination of network systems, virtualization technologies, and data. In the previous work [10], the Global Environment for Network Innovation (GENI) [3] was leveraged as a real-world testbed solution for analysis due to the ability to measure network latency across multiple, federated computer systems while conducting network transactions over the public Internet. In this work, a high-performance system was leveraged to purposely focus on the accuracy of data analysis and inspection of PhishLimiter. The following denotes the hardware specifications for the testing environment: 2x Intel E5-2660 @ 2.2 GHz (32-Cores), 192 GB Memory, 30 TB of Hard Drive Space in ZFS with L2Arc/SLOG configuration, and 40Gbps network interfaces (virtual).

3.5.1 Network Topology

PhishLimiter leverages several virtualization technologies to support the identification and deterrence of Phishing attacks. Figure 3.9 depicts the topology to support experimental evaluation efforts. *Sender-1*, *Sender-2*, and *Sender-3*

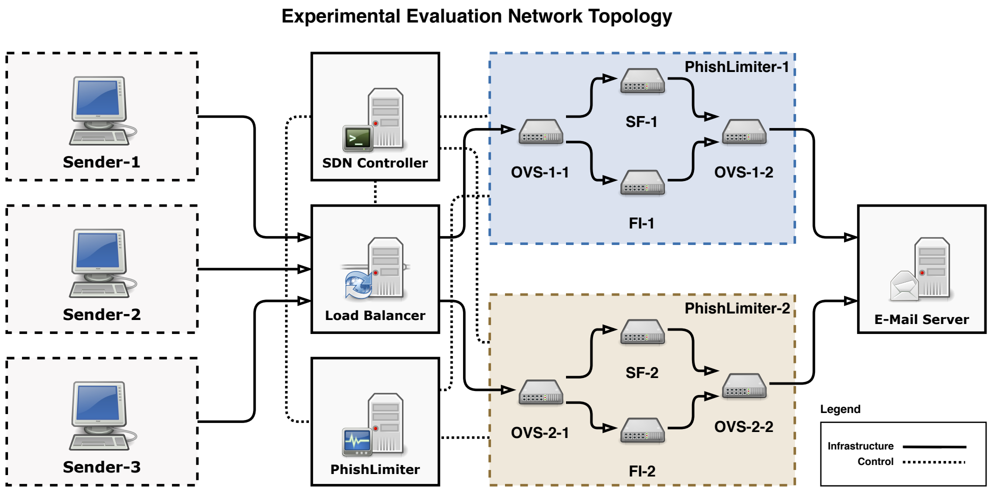


Figure 3.9: Network topology diagram for experimental evaluation

are exact duplicates of one another and serve the purpose of transmitting e-mail messages to the e-mail server. Network traffic enters the *Load Balancer* and traverses through either *PhishLimiter-1* or *PhishLimiter-2*. The outcome of the e-mail transmission arrives at the e-mail server if not deemed malicious. Additionally, all network links in the diagram operate at 1 Gbps throughput to model standard network configurations where the diagram indicates two layers of SDN (e.g., Infrastructure and Control). Network IP addresses are statically assigned to simplify testing purposes. Lastly, all devices within *PhishLimiter-1* and *PhishLimiter-2* connect to the Control plane as per the design of SDN. Following the network, topology is the description of each machine and its configuration for conducting experiments.

3.5.2 System Configuration and Specifications

The network topology diagram expressed several virtual systems in conducting experiments. Previous works [10, 11, 26] presented cases of data transmission and configuration while the following express essential design requirement for this study with references to Figure 3.9.

Sender-X and E-Mail Server

Three computer systems transmit e-mail traffic to the environment denoted as *Sender-1*, *Sender-2*, and *Sender-3*. Each system houses a local copy of the datasets listed for the analysis, as indicated from Table 3.3.2 in Subsection 3.3.2 above. Each VM operates 1 core and 2 GB of memory with Ubuntu 18.04.4 as the operating system. Graphical interfaces for the systems are not configured as each Sender system merely transmits e-mail messages to the receiving e-mail server. The e-mail server for this study merely receives messages, has a 1 core and 2 GB memory design to match the client, and uses Postfix as the software. No e-mail inboxes are configured on the server and will automatically delete messages upon arrival to conserve system resources.

SDN Controller

The Controller for the SDN environment operates Floodlight [20] as the software solution. The network topology presented nine devices that require management from the Controller (Load Balancer, OVS, SF, and FI). The VM operating Floodlight uses 1 core and 4 GB of memory with Ubuntu 18.04.4 as the operating system of choice. The configuration of Floodlight follows the default deployment from the code repository [20]. The VM exposes the REST API interface of Floodlight to allow PhishLimiter to interact with the SDN environment, thereby allowing flow manipulation and directing traffic from SF to FI and vice-versa.

PhishLimiter, Load Balancer, OVS, SF, and FI

The architecture of the experimental evaluation utilizes Open vSwitch [33] as the selected virtual switching platform for ease of deployment, development flexibility [11, 26], and popularity amongst research work [7, 8, 10]. The Load Balancer [13] utilizes a Round Robin approach between two instances of PhishLimiter to contrast the overall accuracy of PhishLimiter while identifying potential resource constraints (e.g., the overburden of an instance). All systems are connected to the Controller using the control plane of the network as per the SDN design. Lastly, SF and FI maintain a connection to the system running PhishLimiter per design for updates on score and reporting. System resource specifications for these systems are set at 1 core and 4GB of memory with Ubuntu 18.04.4.

Chapter 4

Results

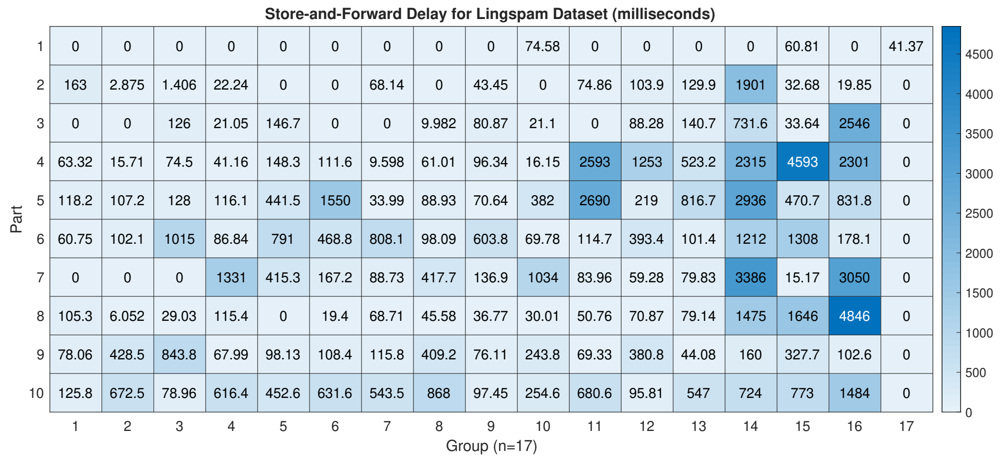
This chapter presents the evaluation of PhishLimiter to express the overall solution's effectiveness using a real-world testing environment. Further results are shown regarding measurements and samples of performance, timing, and accuracy of experiments that portray the overall ability to protect an environment from potential Phishing attacks.

4.1 Performance Analysis

This study presented the challenge that traditional e-mail inspection systems delay and prolong the delivery of an e-mail message due to the many steps necessary for data analysis. PhishLimiter presented a two-lane inspection approach to combat some performance concerns. The following describes the deep level timing analysis and a study on the overall communication behaviors and patterns for the SF and FI lanes.

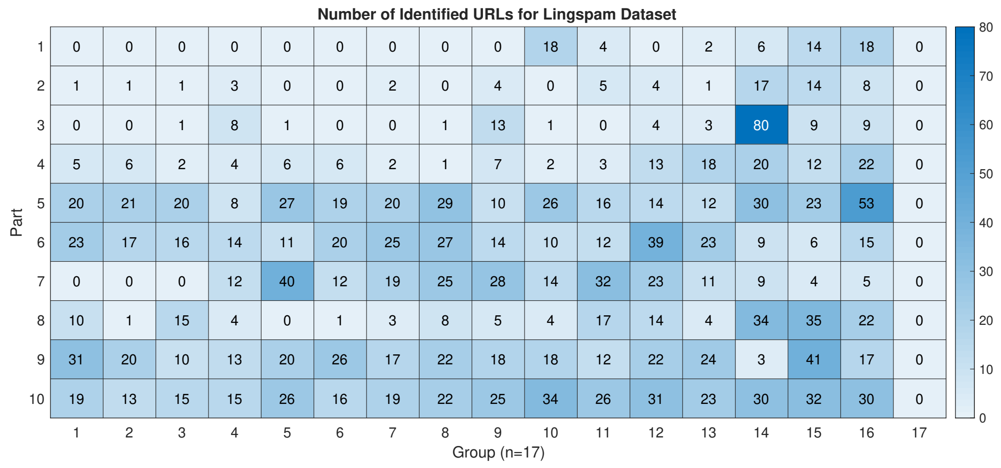
4.1.1 E-Mail Delivery Timing and Delay

The nature of the SF lane within PhishLimiter introduces a delay as messages are held for analysis before being forwarded to the destination (if not malicious). The Lingspam dataset [36] contains 10 folders with 289 text files (e.g. e-mail messages). Figure 4.1 shows the overall measurement of time the SF lane where the files are grouped in sets of 17. The Lingspam dataset's structure contains ten parts (e.g., folders) as to correspond to the y-axis in Figure 4.1. Testing transmitted a set of messages (files) across the network to only the SF lane to identify the induced delay when the analysis holds the



Part	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	0	0	0	0	0	0	0	0	0	74.58	0	0	0	0	60.81	0	41.37
2	163	2.875	1.406	22.24	0	0	68.14	0	43.45	0	74.86	103.9	129.9	1901	32.68	19.85	0
3	0	0	126	21.05	146.7	0	0	9.982	80.87	21.1	0	88.28	140.7	731.6	33.64	2546	0
4	63.32	15.71	74.5	41.16	148.3	111.6	9.598	61.01	96.34	16.15	2593	1253	523.2	2315	4593	2301	0
5	118.2	107.2	128	116.1	441.5	1550	33.99	88.93	70.64	382	2690	219	816.7	2936	470.7	831.8	0
6	60.75	102.1	1015	86.84	791	468.8	808.1	98.09	603.8	69.78	114.7	393.4	101.4	1212	1308	178.1	0
7	0	0	0	1331	415.3	167.2	88.73	417.7	136.9	1034	83.96	59.28	79.83	3386	15.17	3050	0
8	105.3	6.052	29.03	115.4	0	19.4	68.71	45.58	36.77	30.01	50.76	70.87	79.14	1475	1646	4846	0
9	78.06	428.5	843.8	67.99	98.13	108.4	115.8	409.2	76.11	243.8	69.33	380.8	44.08	160	327.7	102.6	0
10	125.8	672.5	78.96	616.4	452.6	631.6	543.5	868	97.45	254.6	680.6	95.81	547	724	773	1484	0

Figure 4.1: E-mail transmission delay on SF lane while the Lingspam Dataset is leveraged for analysis where the denoted values are averages of all identified e-mail messages within their respective group



Part	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	0	0	0	0	0	0	0	0	0	18	4	0	2	6	14	18	0
2	1	1	1	3	0	0	2	0	4	0	5	4	1	17	14	8	0
3	0	0	1	8	1	0	0	1	13	1	0	4	3	80	9	9	0
4	5	6	2	4	6	6	2	1	7	2	3	13	18	20	12	22	0
5	20	21	20	8	27	19	20	29	10	26	16	14	12	30	23	53	0
6	23	17	16	14	11	20	25	27	14	10	12	39	23	9	6	15	0
7	0	0	0	12	40	12	19	25	28	14	32	23	11	9	4	5	0
8	10	1	15	4	0	1	3	8	5	4	17	14	4	34	35	22	0
9	31	20	10	13	20	26	17	22	18	18	12	22	24	3	41	17	0
10	19	13	15	15	26	16	19	22	25	34	26	31	23	30	32	30	0

Figure 4.2: A count on the number of URLs identified in the Lingspam dataset

e-mail message. The calculation of the delay only focuses on when a message enters and exists in the SF system. Several cells denote the value of zero, indicating that the initial inspection of the message was unable to identify any segment that merit analysis using feature extraction. Part 8's Group 16 showed the highest delay with a 4.846-second delay. Deep analysis reveals the following URLs were identified as part of the inspection process. Appendix C shows the list of URLs identified as part of the analysis for Part 8's Group 16. Appendix C indicates a set of regex matching patterns for identifying a URL and Figure 4.2 shows the number of identified URLs within the Lingspam dataset. Notably, URLs' quantity does not appear to correlate to the overall delay of forwarding message to the destination but rather the complexity of the domain name, network latency to retrieve web data, and whether the website exists. Part 7's Group 4 showed a 1,331 ms delay of forwarding a message with the identification of 12 URLs, while Part 7's Group 14 experienced a 3,386 ms delay. The evaluation of FI presents additional comparisons against SF for inspection time and analysis.

Evaluating the time between FI forwarding the message to the destination and the time the message undergoes inspection presented an interesting comparison. PhishLimiter's overall risk pertains to FI potentially forwarding a malicious message to the destination before the inspection. Figure 4.3 shows a box plot of the approximate time discrepancies analyzing the Lingspam Part 10 data set over ten rounds. Notably, Group 17 indicated 0 delays as no URLs were identified as to match previous results from Figure 4.2. The highest delay was noted at almost 1,500 ms delay as to match similar results SF inspection time. Further analysis was conducted for the time necessary for inspection in FI after the delivery of the message. Figure 4.4 shows a depiction of an incremental delay for the analysis of 375 messages sent sequentially (e.g., one at a time). The graph indicates that the inspection time for FI increased slowly over time while the curve began to plateau between 235 and 275. Sampled data were fitted using a powerfit with R^2 and RMSE at 0.9547 and 0.2588, respectively. Lastly, system resources was monitored for utilization requirements

4.1.2 Inspection Under Denial-of-Service or Burst E-Mails

A major risk to any inspection system involves the case of Denial-of-Service (DoS). While FI has some safeguards in the sense that messages are delivered to the user under such constraint—SF presents the risk that messages are queued in the SFBuffer waiting for inspection. A deeper study was evaluated

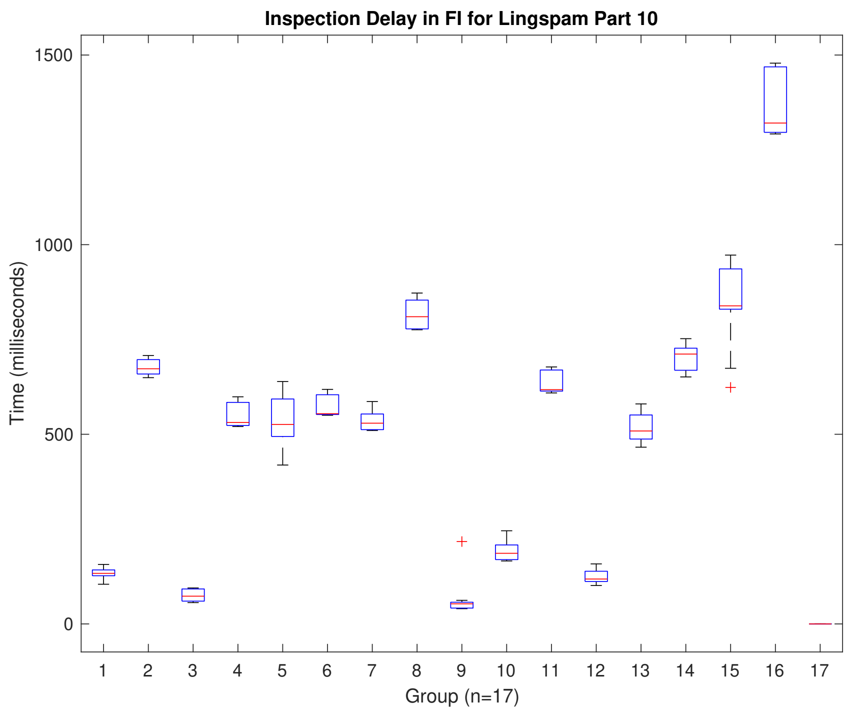


Figure 4.3: Inspection time for FI after delivery of an e-mail message

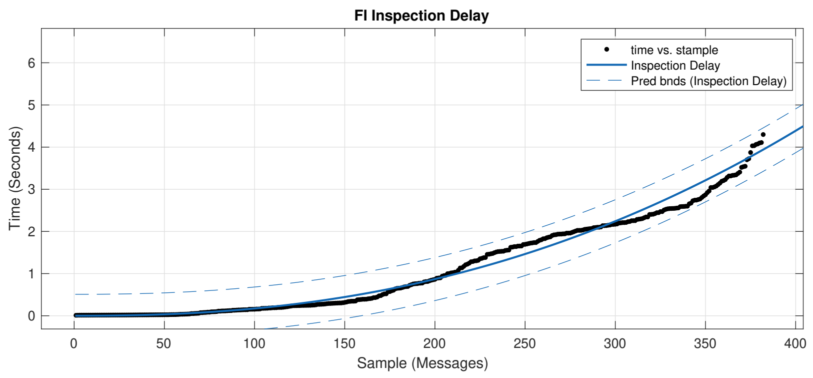


Figure 4.4: Inspection time for FI after delivery of an e-mail message

for such a scenario where all datasets were intermixed with one another and immediately forwarded through PhishLimter to emulate a sudden burst of network traffic. Figure 4.5 and 4.6 show CPU and Memory usage for the burst scenario, respectively. The CPU usage shown presents a compelling

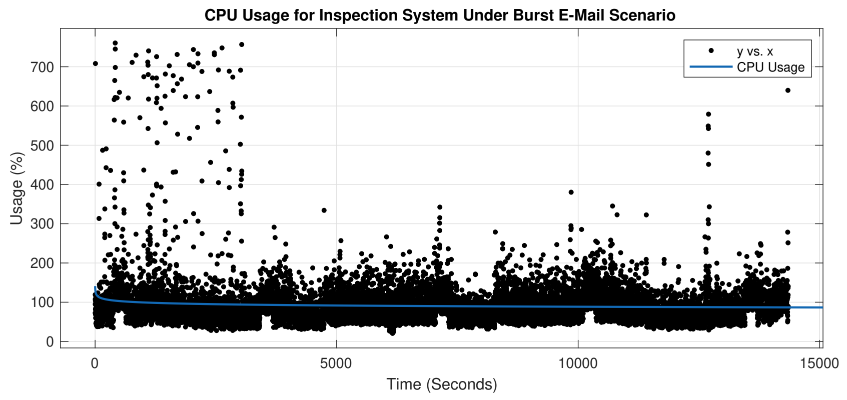


Figure 4.5: Inspection system CPU usage under resource constraint from the burst scenario

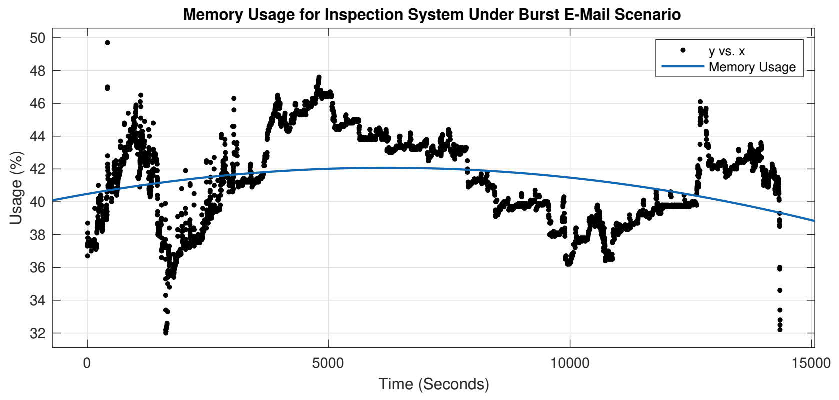


Figure 4.6: Inspection system memory usage under resource constraint from the burst scenario

case where the immediate burst of e-mail message caused an oversubscription of system resources per points greater than 100%. Memory usage increased briefly but subsided over time, indicating little impact on the overall bust of traffic. The lower and upper bound for CPU usage was 20.5% and 760.5%, respectively. Memory usage ranged from the lower and upper bound at 32% and 49.7%, respectively.

4.1.3 Lane Switching Dynamics

The switching and transitions between SF to FI and vice-versa presents an area of evaluation for performance and delay. AN SDN flow periodically refreshes itself for the presence of matching IP addresses and ports in a flow table. A network flow can never expire due to the continuous stream of e-mail traffic and data from a particular network (e.g., cloud provider). Switching between the two inspection lanes for new flows is trivial, as it is merely a network path configuration within SDN. Manipulation of the flow during the presence of traffic presence the concern of packet loss or a brief moment in time where messages are disrupted. Likewise, an e-mail message spans across multiple packets such that the transition between SF to FI and vice-versa may leave some portion in one lane and the rest in the other. Figure 4.7 shows the evaluation of the disruption of network traffic when switching between the two inspection lanes using Ping. Six measurements indicated a disruption of

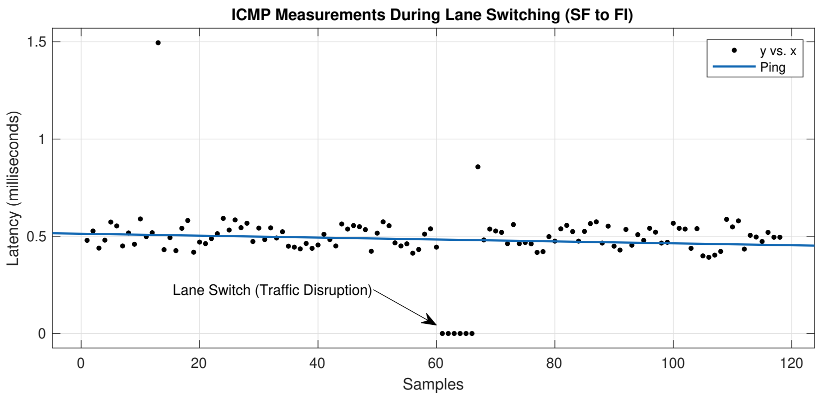


Figure 4.7: Network traffic disruption switching between SF and FI

the communication briefly as denoted with a latency of 0 for samples 61 to 66. The brief disruption is for cases where the lane switching process transitioned ICMP request and that the switch requires a bi-directional insertion of rules (e.g., sender to receiver and vice-versa). Traffic may route through FI, while the return path may traverse through SF during the switching.

Chapter 5

Discussion

The previous chapter identified several positive results demonstrating the effectiveness of PhishLimiter. This chapter presents some discussion on several assumptions of risk as part of this study with threat modeling.

5.1 Threat Model and Risk

The architecture and design of PhishLimiter present areas of concern when factoring the scenario of a malicious actor having access within the network (e.g., inside threat). External to the network, normally expected network attacks such as Denial-of-Service [8] or the Distributed-variant [7] are often combated by Content-Delivery-Network (CDN) and DDoS mitigation service such as Cloudflare [12]. Exploitation efforts such as attacking the e-mail server or a load balancer are also safeguarded by network firewalls, patch management, and vulnerability scanning efforts. Attacks that leverage a 0-day exploit are unavoidable. A secondary or compensating control such as defense-in-depth, Access-Control List (ACL), and Role-Base Access Controls (RBAC) is necessary to provide a layer of protection. PhishLimiter cannot safeguard such a style of attacks. A harmonious solution of several types of security systems and architectures is needed to mitigate and a wide-spread of varying attacks. The following describes a set of example threats and risk that presents challenges towards PhishLimiter's architecture.

5.1.1 Controller operates an insecure API

Controllers [20] offer the ability to interact with the SDN environment using a REST API to support the ability of the programmability of a network. Many API by default configurations remain insecure (e.g., plain text protocol) and do not require authentication (e.g., unauthenticated access).
Impact: An adversary may manipulate the environment using SDN flow insertion rules, thereby redirecting traffic to a malicious location in the network.
Justification: An adversary gaining access to the control plane of the SDN environment presents a significant challenge and achievement. Two methods to gain such access are through a potential pivot from the infrastructure plane of SDN or by compromising a node with access to the control plane.
Controls: Many controllers offer the ability to enable encryption for their REST API in addition to leveraging key exchanges and other authentication methods (e.g., LDAP). Lastly, enabling further protection such as Host-Intrusion Prevention System (HIPS) and other compensating controls (e.g., endpoint protection) minimizes the overall risk further from impacting PhishLimiter.

5.1.2 Time-based Denial of Service (DoS) Attack

A Round-Robin approach of load balancing allows for a potential risk where an adversary tricks the overall system to route high-volume and malicious traffic to a single inspection system, thereby causing a significant bottleneck in the environment.
Impact: The bottleneck delays non-malicious traffic, disrupting end-users receiving e-mail and prolonging message delivery indefinitely.
Justification: An adversary to gain such knowledge on the quantity of SF and FI systems requires significant inside knowledge of the network environment and architecture. Additionally, to create such an attack requires an adversary to control multiple IP addresses that are external to the network environment operating PhishLimiter as the controller creates a flow per matching source and destination IP address and port.
Controls: Monitoring of system resources a periodic checking and evaluation of message delivery time through a Service Level Agreement (SLA) offers a cause to monitor for then potential DoS attack.

5.1.3 Forgotten Messages in FI System Failure

The operation of PhishLimiter undergoes a situation where one of the inspection lanes fails due to hardware problems or the underlying operating system.
Impact: This scenario ultimately creates a situation where messages cannot be delivered to the final destination. In some cases–a user receives a message, and FI fails before the inspection.
Justification: Organizations naturally experience system failures as part of the life span of hardware devices, poor software behaviors, or unexpected events. Computer systems are often monitored for the health and status of the device, thereby presenting immediate indications for the environment's failure.
Controls: An organization may implement a monitoring solution to evaluate an inspection system's health with an SLA requirement defined by the organization's risk appetite. Messages arriving at the destination where FI fails to conduct inspections require a case of a compensating control such as endpoint security and HIPS.

5.1.4 Reputation Score Inflation

The awareness of PhishLimiter presents the case where an adversary purposely sends numerous benign messages to boost or inflate the reputation score to a positive value such that traffic inspection switches from SF to FI.
Impact: It allows for the potential risk of an end-user to receive a severely malicious e-mail message before inspection in the FI process.
Justification: The study results studied the delay process of inspecting a message under FI, indicating that the time an adversary potentially gains from any malicious activities results in under a 5-second window.
Controls: Standard safeguards include endpoint protection, RBACS, HIPS, and user training.

5.1.5 Score Update Replay Attack

An adversary gains access to the SDN environment's control plane and the ability to modify and replay reputation score messages.
Impact: The ability to inflate or deflate specific reputation scores of flows in the environment allows for potential directing of malicious and benign traffic to FI and SF, respectively.
Justification: To gain such access to the control plane of SDN requires a significant effort by an adversary, as previously mentioned in Sub Section 5.1.1.

Additionally, the ability to manipulate and replay messages as an approach of spoofing presence a challenge to interject between SF and FI connection to PhishLimiter.
<u>Controls:</u> The implementation of authentication or public key validates the communication channel, thereby preventing potential risk of replay attacks.

Chapter 6

Conclusion

This study presented an extension to PhishLimiter as an approach to improve the delivery of e-mail communication while combating potential phishing attacks. Traditional e-mail inspection systems hold messages for analysis before releasing the content to the recipient, thereby establishing a delivery delay. Users who are sending and receiving e-mail messages with high priority or time-sensitive urgency may not have the luxury to wait for an e-mail inspection system to complete the analysis. PhishLimiter presented a solution that dynamically speeds up and decreases the delivery of an e-mail message with a two-lane inspection approach of Store-and-Forward and Forward-and-Inspect. The results demonstrated that PhishLimiter presents a viable solution to combat phishing attacks while providing a potential approach to quickly receiving e-mail messages.

Bibliography

[1] APWG. *Phishing Activity Trends Report, 4th Quarter 2019.* APWG, Feb 2020.

[2] Nalin Asanka Gamagedara Arachchilage, Steve Love, and Konstantin Beznosov. Phishing threat avoidance behaviour: An empirical investigation. *Computers in Human Behavior*, 60:185–197, 2016.

[3] Mark Berman, Jeffrey S Chase, Lawrence Landweber, Akihiro Nakao, Max Ott, Dipankar Raychaudhuri, Robert Ricci, and Ivan Seskar. Geni: A federated testbed for innovative network experiments. *Computer Networks*, 61:5–23, 2014.

[4] Aaron Blum, Brad Wardman, Thamar Solorio, and Gary Warner. Lexical feature based phishing url detection using online learning. In *Proceedings of the 3rd ACM Workshop on Artificial Intelligence and Security*, pages 54–60, 2010.

[5] Broadcom. Certificate pinning. *URL: https://docs.broadcom.com/doc/certificate-pinning-en*, 2017.

[6] Niccolò Cascarano, Luigi Ciminiera, and Fulvio Risso. Improving cost and accuracy of dpi traffic classifiers. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 641–646, 2010.

[7] Tommy Chin, Xenia Mountrouidou, Xiangyang Li, and Kaiqi Xiong. An sdn-supported collaborative approach for ddos flooding detection and containment. In *MILCOM 2015-2015 IEEE Military Communications Conference*, pages 659–664. IEEE, 2015.

[8] Tommy Chin, Xenia Mountrouidou, Xiangyang Li, and Kaiqi Xiong. Selective packet inspection to detect dos flooding using software defined

networking (sdn). In *2015 IEEE 35th international conference on distributed computing systems workshops*, pages 95–99. IEEE, 2015.

[9] Tommy Chin, Mohamed Rahouti, and Kaiqi Xiong. End-to-end delay minimization approaches using software-defined networking. In *Proceedings of the International Conference on Research in Adaptive and Convergent Systems*, pages 184–189, 2017.

[10] Tommy Chin, Kaiqi Xiong, and Chengbin Hu. Phishlimiter: A phishing detection and mitigation approach using software-defined networking. *IEEE Access*, 6:42516–42531, 2018.

[11] Tommy Chin, Kaiqi Xiong, Chengbin Hu, and Yi Li. A machine learning framework for studying domain generation algorithm (dga)-based malware. In *International Conference on Security and Privacy in Communication Systems*, pages 433–448. Springer, 2018.

[12] Cloudflare. Cloudflare. *URL: https://www.cloudflare.com/*, 2020.

[13] Floodlight Controller. Load balancer. *URL: https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/1343617/*, 2016.

[14] Avisha Das, Shahryar Baki, Ayman El Aassal, Rakesh Verma, and Arthur Dunbar. Sok: A comprehensive reexamination of phishing research from the security perspective. *IEEE Communications Surveys & Tutorials*, 2019.

[15] Tamara Denning, Adam Lerner, Adam Shostack, and Tadayoshi Kohno. Control-alt-hack: the design and evaluation of a card game for computer security awareness and education. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 915–928, 2013.

[16] Yan Ding, Nurbol Luktarhan, Keqin Li, and Wushour Slamu. A keyword-based combination approach for detecting phishing webpages. *computers & security*, 84:256–275, 2019.

[17] Ronald C Dodge Jr, Curtis Carver, and Aaron J Ferguson. Phishing for user security awareness. *computers & security*, 26(1):73–80, 2007.

[18] Expired Domains. Expired domains. *URL: https://www.expireddomains.net/*, 2020.

[19] Enron. Enron email dataset. *Enron*, 2015.

[20] Floodlight. Floodlight. *URL: https://github.com/floodlight/floodlight*, 2020.

[21] Simson Garfinkel. *PGP: pretty good privacy.* " O'Reilly Media, Inc.", 1995.

[22] Uttam Ghosh, Pushpita Chatterjee, Deepak Tosh, Sachin Shetty, Kaiqi Xiong, and Charles Kamhoua. An sdn based framework for guaranteeing security and performance in information-centric cloud networks. In *2017 IEEE 10th International Conference on Cloud Computing (CLOUD)*, pages 749–752. IEEE, 2017.

[23] Mike Hall and Kevin Wiley. Capacity verification for high speed network intrusion detection systems. In *International Workshop on Recent Advances in Intrusion Detection*, pages 239–251. Springer, 2002.

[24] Michael Hart, Pratyusa Manadhata, and Rob Johnson. Text classification for data loss prevention. In *International Symposium on Privacy Enhancing Technologies Symposium*, pages 18–37. Springer, 2011.

[25] Amir Herzberg and Ahmad Gbara. Trustbar: Protecting (even naive) web users from spoofing and phishing attacks. Technical report, Cryptology ePrint Archive, Report 2004/155. http://eprint. iacr. org/2004/155, 2004.

[26] Yi Li, Kaiqi Xiong, Tommy Chin, and Chengbin Hu. A machine learning framework for domain generation algorithm-based malware detection. *IEEE Access*, 7:32765–32782, 2019.

[27] mail archive. mail-archive. *URL: https://www.mail-archive.com/*, 1998.

[28] METASPLOIT. Google chrome 80 - jscreate side-effect type confusion (metasploit). `https://www.exploit-db.com/exploits/48186`, 2020.

[29] MITRE. Mitre att&ck framework, Sep 2019.

[30] Mozilla. fuzzdata. *URL: https://github.com/MozillaSecurity/fuzzdata*, 2020.

[31] Yuya Jeremy Ong, Mu Qiao, Ramani Routray, and Roger Raphael. Context-aware data loss prevention for cloud storage services. In *2017 IEEE 10th International Conference on Cloud Computing (CLOUD)*, pages 399–406. IEEE, 2017.

[32] Inc Parakweet Labs. Emailintentdataset. *URL: https://github.com/ParakweetLabs/EmailIntentDataSet*, 2014.

[33] Ben Pfaff, Justin Pettit, Teemu Koponen, Ethan Jackson, Andy Zhou, Jarno Rajahalme, Jesse Gross, Alex Wang, Joe Stringer, Pravin Shelar, et al. The design and implementation of open vswitch. In *12th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 15)*, pages 117–130, 2015.

[34] Proof Point. Next generation email security, Oct 2019.

[35] Pawan Prakash, Manish Kumar, Ramana Rao Kompella, and Minaxi Gupta. Phishnet: predictive blacklisting to detect phishing attacks. In *2010 Proceedings IEEE INFOCOM*, pages 1–5. IEEE, 2010.

[36] Georgios Sakkis, Ion Androutsopoulos, Georgios Paliouras, Vangelis Karkaletsis, Constantine D Spyropoulos, and Panagiotis Stamatopoulos. A memory-based approach to anti-spam filtering for mailing lists. *Information retrieval*, 6(1):49–73, 2003.

[37] SpamAssassin. Spamassassin. *URL: https://spamassassin.apache.org/old/publiccorpus/*, 2017.

[38] Symantec. Symantec sitereview. *URL: https://sitereview.bluecoat.com/*, 2020.

[39] Ke Tian, Steve TK Jan, Hang Hu, Danfeng Yao, and Gang Wang. Needle in a haystack: Tracking down elite phishing domains in the wild. In *Proceedings of the Internet Measurement Conference 2018*, pages 429–442, 2018.

[40] Verizon. 2019 verizon data breach investigations report. *Verizon*, 05 2019.

[41] Harald Welte. The netfilter framework in linux 2.4. In *Proceedings of Linux Kongress*, 2000.

[42] Zikai Alex Wen, Zhiqiu Lin, Rowena Chen, and Erik Andersen. What. hack: engaging anti-phishing training through a role-playing phishing simulation game. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2019.

Appendices

Appendix A

Machine Learning Features

#	**Feature**	**Description**	**Implementation Challenges**	**Operational Risk**	+/-
01	IP Address	Determines whether a hyperlink uses an IP address	Utilizes string matching technique where pattern matching may generate a false positive due to IPv6 address shortening	N/A	-
02	URL Length	Determines the length of the URL	Lengthy URLs may indicate a potential phishing attempt through cross-site scripting, session-token usage, etc	N/A	-

Table A.1 continued from previous page

#	Feature	Description	Implementation Challenges	Operational Risk	+/-
03	URL Shortener	Determine whether link utilizes a shorting service such as `bit.ly`	Shorting service may mask malicious links while detection system need to query shortener for the full URL	Actively querying a third-party service may flag the detection system as malicious due to rate and interval of querying	-
04	@ Symbol Usage	Determine whether link utilizes the @ symbol as text prior to the notation are omitted	Distinguishing an e-mail address from a URL presence a challenge due to potential false positives	N/A	-
05	Sub domain Usage	Determine whether sub domains are being utlizes and quantity including matching for organizational names	N/A	N/A	-
06	Domain Expiration Date	Determine whether the domain expires greater than one year	Excessively querying of whois record may cause a temporary block by third-party provider due to rate limiting controls	Querying a third-party may block detection system permanently	+

Table A.1 continued from previous page

#	Feature	Description	Implementation Challenges	Operational Risk	+/-
07	Domain Creation Date	Determine whether the domain's lifespan (creation date) as newly created domain names may purposely host malicious content due to poor or unknown reputation	Excessive querying of whois record may block the detection system	Querying a third-party may block detection system permanently	+
08	Shared IP Address	Determine whether multiple domain names share the same IP address through reserve DNS queries	Excessive querying of whois record may block the detection system	Querying a third-party may block detection system permanently	-
09	Domain Name Pronounceability	Determine whether the domain name and sub domain name are easily pronounceable	International and foreign languages may present difficulties and bias to the feature depending on the implementation approach and end-users	Creates a false positive for alerting	+

Table A.1 continued from previous page

#	**Feature**	**Description**	**Implementation Challenges**	**Operational Risk**	**+/-**
10	Network Port Usage	Determine whether the link utilizes a non-standard web port for accesss	A subset of web applications use alternative ports (i.e., 8080 and 8443) while some administrative interfaces use any network port	N/A	-
11	Anchor Link to Display Text	Determine whether the link's targeted destination matches the display text in the e-mail message	N/A	N/A	-
12	JavaScript Execution	Determine whether the message has embedded JavaScript code for execution	Code may incorporate obfuscation to mislead detection system	N/A	-
13	HTML iFrame Usage	Determine whether the message utilizes an iFrame message has embedded JavaScript code for execution	Code may incorporate obfuscation to mislead detection system	N/A	-

Table A.1 continued from previous page

#	Feature	Description	Implementation Challenges	Operational Risk	+/-
14	Page Ranking	Determine whether the link directs to a reputable domain name using Alexa ranking	The ranking must be frequently updated to ensure accuracy of the detection	N/A	+
15	Google Search Index	Determine whether the domain name matches a top index rating using Google search crawlers	Query must be done in real-time against google service	Frequency of query to Google may accidentally block the detection due to rate-limiting and behavior analysis from Google's detection system	+
16	Bing Search Index	Determine whether the domain name matches a top index rating using Google search crawlers	Query must be done in real-time against google service	Frequency of query to Google may accidentally block the detection due to rate-limiting and behavior analysis from Google's detection system	+

Table A.1 continued from previous page

#	Feature	Description	Implementation Challenges	Operational Risk	+/-
17	Domain Client Update Permission	Reputable organizations limits the permission as prohibited while all other cases–are suspicious	Excessive querying for whois record may result in a temporary block by the DNS server	Provider for whois record may temporary or permanently block the detection system depending on rate and frequency of query	-
18	Domain Client Transfer Permission	Reputable organizations limits the permission as prohibited while all other cases–are suspicious	Excessive querying for whois record may result in a temporary block by the DNS server	Provider for whois record may temporary or permanently block the detection system depending on rate and frequency of query	-
19	Domain Client Delete Permission	Reputable organizations limits the permission as prohibited while all other cases–are suspicious	Excessive querying for whois record may result in a temporary block by the DNS server	Provider for whois record may temporary or permanently block the detection system depending on rate and frequency of query	-

Table A.1 continued from previous page

#	Feature	Description	Implementation Challenges	Operational Risk	+/-
20	Domain Server Update Permission	Reputable organizations limits the permission as prohibited while all other cases–are suspicious	Excessive querying for whois record may result in a temporary block by the DNS server	Provider for whois record may temporary or permanently block the detection system depending on rate and frequency of query	-
21	Domain Server Transfer Permission	Reputable organizations limits the permission as prohibited while all other cases–are suspicious	Excessive querying for whois record may result in a temporary block by the DNS server	Provider for whois record may temporary or permanently block the detection system depending on rate and frequency of query	-
22	Domain Client Delete Permission	Reputable organizations limits the permission as prohibited while all other cases–are suspicious	Excessive querying for whois record may result in a temporary block by the DNS server	Provider for whois record may temporary or permanently block the detection system depending on rate and frequency of query	-

Table A.1 continued from previous page

#	Feature	Description	Implementation Challenges	Operational Risk	+/-
23	Domain Client Hold Status	Determine whether the domain name has a hold status indicating potential legal issues	Excessive querying for whois record may result in a temporary block by the DNS server	Provider for whois record may temporary or permanently block the detection system depending on rate and frequency of query	-
24	Domain Whois Guard	Determines whether the domain implements whois guard to redact the information in the entry. Reputable organizations do not implement whois guard for operational purposes	Excessive querying for whois record may result in a temporary block by the DNS server	Provider for whois record may temporary or permanently block the detection system depending on rate and frequency of query	-
25	GeoIP Location	Determines whether the hosting location of the domain has any suspicious or present danger to the organization	Location position may be inaccurate for near-border locations and cloud service providers. GeoIP table requires frequent update to ensure accuracy of the detection system.	N/A	-

Table A.1 continued from previous page

#	Feature	Description	Implementation Challenges	Operational Risk	+/-
26	Domain Registrar	Determines the reputation of the domain registrar	Excessive querying for whois record may result in a temporary block by the DNS server	Provider for whois record may temporary or permanently block the detection system depending on rate and frequency of query	-
27	File Attachment	If the file contains an executable or file encryption	Encrypted files cannot be inspected without an appropriate password or key and that the size of a file may be significantly large for any analysis	Large files may overburden the inspection system for analysis due to complexity of the file or a variety of detection alerts for matching signatures	-

Appendix B

URL Identification Regex Statements

The following presents three methods of identifying and matching URLs within an e-mail message where B.1 and B.3 has the highest and lowest detection accuracy, respectively.

B.1 @diegoperini

```
_^(?:(?:https?|ftp)://)(?:\S+(?::\S*)?@)?(?:(?!10(?:\.\d{1,3}){3})
    (?!127(?:\.\d{1,3}){3})(?!169\.254(?:\.\d{1,3}){2})
    (?!192\.168(?:\.\d{1,3}){2})(?!172\.(?:1[6-9]|2\d|3[0-1])(?:\.\d
    {1,3}){2})(?:[1-9]\d?|1\d\d|2[01]\d|22[0-3])(?:\.(?:1?\d
    {1,2}|2[0-4]\d|25[0-5])){2}(?:\.(?:[1-9]\d?|1\d\d|2[0-4]\d
    |25[0-4]))|(?:(?:[a-z\x{00a1}-\x{ffff}0-9]+-?)*[a-z\x{00a1}-\x{
    ffff}0-9]+)(?:\.(?:[a-z\x{00a1}-\x{ffff}0-9]+-?)*[a-z\x{00a1}-\x{
    ffff}0-9]+)*(?:\.(?:[a-z\x{00a1}-\x{ffff}]{2,})))(?::\d{2,5})
    ?(?:/[^\s]*)?$_iuS
```

B.2 Spoon Library

```
/(((http|ftp|https):\/{2})+(([0-9a-z_-]+\.)+(aero|asia|biz|cat|com|
    coop|edu|gov|info|int|jobs|mil|mobi|museum|name|net|org|pro|tel|
    travel|ac|ad|ae|af|ag|ai|al|am|an|ao|aq|ar|as|at|au|aw|ax|az|ba|bb
    |bd|be|bf|bg|bh|bi|bj|bm|bn|bo|br|bs|bt|bv|bw|by|bz|ca|cc|cd|cf|cg
    |ch|ci|ck|cl|cm|cn|co|cr|cu|cv|cx|cy|cz|cz|de|dj|dk|dm|do|dz|ec|ee
    |eg|er|es|et|eu|fi|fj|fk|fm|fo|fr|ga|gb|gd|ge|gf|gg|gh|gi|gl|gm|gn
    |gp|gq|gr|gs|gt|gu|gw|gy|hk|hm|hn|hr|ht|hu|id|ie|il|im|in|io|iq|ir
    |is|it|je|jm|jo|jp|ke|kg|kh|ki|km|kn|kp|kr|kw|ky|kz|la|lb|lc|li|lk
    |lr|ls|lt|lu|lv|ly|ma|mc|md|me|mg|mh|mk|ml|mn|mn|mo|mp|mr|ms|mt|mu
    |mv|mw|mx|my|mz|na|nc|ne|nf|ng|ni|nl|no|np|nr|nu|nz|nom|pa|pe|pf|
    pg|ph|pk|pl|pm|pn|pr|ps|pt|pw|py|qa|re|ra|rs|ru|rw|sa|sb|sc|sd|se|
    sg|sh|si|sj|sj|sk|sl|sm|sn|so|sr|st|su|sv|sy|sz|tc|td|tf|tg|th|tj|
    tk|tl|tm|tn|to|tp|tr|tt|tv|tw|tz|ua|ug|uk|us|uy|uz|va|vc|ve|vg|vi|
    vn|vu|wf|ws|ye|yt|yu|za|zm|zw|arpa)(:[0-9]+)?((\/([~0-9a-zA-Z
    \#\+\%@\.\/_-]+))?(\?[0-9a-zA-Z\+\%@\/&\[\];=_-]+)?)?))\b/imuS
```

B.3 stephenhay

```
@^(https?|ftp)://[^\s/$.?#].[^\s]*$@iS
```

Appendix C

SF Delay on Lingspam Part 8 Group 16 URLs

A set of identified URLs as an initial cause to 4,846 ms delay in the SF lane for inspection. Several URLS are of standard format while a mismatching occured for `http://@` while numerical formats such as `http://3624108859` translated to the IP address `http://216.3.131.59/`. The following shows a sample list of identified URLs for analysis while the author omitted some results due to profanity or non-professional terminologies:

```
http://@
http://capitalfm.com
http://capitalfm.com
http://capitalfm.com
http://capitalfm.com
http://capitalfm.com
http://www.virtcasino.com
http://capitalfm.com
http://capitalfm.com
http://capitalfm.com
http://www.ixpres.com
http://www.onlinemlm.com
http://www.promlm.com
http://3624108859
http://3511663956
http://3488977290
http://3510834019
```

```
http://www.raquelscasino.com
http://www.tobet.com
http://www.nekonline.com.nek
http://www.mplayer.com
http://www.gamehub.net.thank
http://capitalfm.com
http://capitalfm.com
http://capitalfm.com
http://capitalfm.com
http://www.virtcasino.com
http://capitalfm.com
http://www.highway.bt.com
http://capitalfm.com
http://www.kaya-optics.com
http://www.truster.com
http://208.144.10.20
http://208.166.75.251
http://22
```