Spears Against Shields: Are Defenders Winning the Phishing War?

Ayman El Aassal
University of Houston
Houston, TX
aelaassal@uh.edu

Rakesh Verma
University of Houston
Houston, TX
rverma@uh.edu

ABSTRACT

Phishing is the act of using deceptive methods to lure users into taking harmful and dangerous actions against themselves and/or the company they work for. Hackers have been using this method to bypass various security systems, steal personal information, make sensitive data public, and all of this while raking millions of dollars. **On the attack side**, hackers are using phishing kits and employ all possible phishing techniques to build a successful campaign. **On the defense side**, we find spam/phishing filters and malicious website detection in addition to anti-phishing simulation training. In this study, we assess the current phishing landscape by testing the tools used on both sides of the war. We generated 1,000 phishing emails containing phishing links semi-automatically using natural language generation technology and tested five popular tools with anti-phishing modules. The number of undetected emails ranged from 77 to 927. We also evaluate several anti-phishing training technologies and reveal their shortcomings. Our results suggest that both anti-phishing filters and current training tools have a long way to go and thus improving these defense mechanisms is still essential.

CCS CONCEPTS

• Security and privacy → Phishing;

KEYWORDS

Phishing, Phishing Tool Comparison, Anti-viruses

ACM Reference Format:

Ayman El Aassal and Rakesh Verma. 2019. Spears Against Shields: Are Defenders Winning the Phishing War?. In *Fifth ACM International Workshop on Security and Privacy Analytics (IWSPA '19)*, March 27, 2019, Richardson, TX, USA. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3309182.3309191>

1 INTRODUCTION

The defense landscape has been growing stronger over the years, yet phishing threats are still prevalent and succeeding. The reason for this phenomenon is that *phishing* is a social engineering

attack, which means it relies on human error. It is now quite common to receive emails, which appear to be from a well-known company, inviting recipients to click on a link in order to claim a reward or retype credentials to an online account. These types of emails are obviously scams, yet quite a few people still take the bait, as reported by Anti-Phishing Working Group, FBI Infragard, and others.¹ Although bad as it seems, it is still not the worst form of phishing attack. The worst, or most challenging kind, is called spear-phishing. In spear-phishing, attackers target an organization and carefully craft customized phishing emails that have a high success probability against its employees. The serious concern is that if one of the employees falls for the trick, the whole organization or at least a major part of it is adversely affected. Emails sent in spear-phishing campaigns are well crafted and hard to detect. According to a survey conducted by Cloudmark, the average cost of a spear-phishing campaign is \$1.6 million.²

A lot of research exists addressing multiple aspects of this problem. For example, solutions that focus on analyzing phishing emails [6, 9, 10, 14, 18, 23, 28, 30], and use machine learning algorithms and feature selection to raise detection rates [1, 2, 5, 16, 22, 32, 43]. Others work on detecting phishing websites, e.g., [31, 33], and analyzing URLs, e.g., [29, 37, 39]. We can also find studies that test user awareness and reaction to receiving or viewing phishing emails and/or websites [3, 13, 26, 27], in an attempt to get more insights and come up with new ideas on how to protect the end user. Another stream of research focuses on collecting and analyzing the tools used by phishers to create the campaigns [8, 17], thus explaining the methods used by attackers.

All this research raises the important question of why phishing still works when there is so much being done to combat the problem with filters and user awareness/training? It is hard to find a reasonable-size organization that does not use phishing campaigns to train employees and filters to stop phishing attacks. Yet attacks are still succeeding.

To find the reasons for the above paradox, we test anti-phishing tools and the technologies used for training employees. Testing anti-phishing tools with old phishing campaigns is not the optimal approach since the tools keep hashes of old attacks for comparison and since the attackers keep coming up with new attacks. Hence, we show how to generate attacks semi-automatically in this paper using NLG technology. We also evaluate the features and effectiveness of the anti-phishing training tools. To summarize, we make the following contributions in this paper:

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IWSPA '19, March 27, 2019, Richardson, TX, USA

© 2019 Copyright held by the owner/author(s). Publication rights licensed to Association for Computing Machinery.

ACM ISBN 978-1-4503-6178-1/19/03...\$15.00

<https://doi.org/10.1145/3309182.3309191>

¹<https://www.getcybersafe.gc.ca/cnt/rsrscs/nfgrphcs/nfgrphcs-2012-10-11-en.aspx>

²<https://blog.cloudmark.com/2016/01/13/survey-spear-phishing-a-top-security-concern-to-enterprises/>

- (1) We show how new phishing attacks can be generated using natural language generation (NLG) technology. Using these attacks, we study the effectiveness of anti-phishing tools available within five anti-malware tools. We also test these tools with phishing emails created using the templates available in anti-phishing training tools. We show that these tools fail, some of them quite badly, to catch many of our attacks. This happened even though we intentionally did not try to create complex grammars to generate the phishing emails (Section 2).
- (2) We collect anti-phishing simulation and training tools, install and review them based on a list of criteria that we find are common to these tools and also in light of the deceptive methods used by phishers. Then we suggest some improvements based on our review (Section 3).

In Section 4 we discuss the related work and Section 5 concludes the paper.

2 GOING AGAINST ANTI-SPAM, ANTI-PHISHING SHIELDS

In this section, we test five popular anti-virus tools (AVs) that offer anti-spam and anti-phishing modules. We test these tools in two scenarios: (i) using email templates from the anti-phishing training tools that we study in a subsequent section, and (ii) using semi-automatically generated phishing emails based on NLG technology.

Note that this experiment is done to test machine based detection not human based detection, that is why not much attention will be paid to the presentation of the email. Rather the focus will be strictly on the content, which means: URLs and IP addresses in the body, malicious attachments, fake or spoofed sender addresses, and common sentences that entice threats or urgency. We will use the detection rate of the AVs as a measure for the experiments.

To conduct these experiments, we created our own virtual domain and mail server to receive the phishing emails. The mail server was created with Hmailserver³ on a virtual machine hosting Windows 8. We used Office 365's Outlook to receive the emails since it could be configured to host accounts using POP3, and is compatible with most AVs' add-ins. Also, since we wanted to send a large number of emails, we could not use Gmail's SMTP server because it has a quota on the number of emails allowed to be sent.

The rest of the section is organized as follows: First, we describe how we used the email templates from anti-phishing training tools to test the AVs with live phishing websites. Second, we describe how to create phishing emails automatically using NLG technology, and use them to test the AVs with live phishing websites and a number of malwares.

2.1 Testing AVs Using Training Templates

Through this first experiment, we can gather some insights on how the emails created by the anti-phishing training tools are considered by popular anti-viruses.

Since not all the training tools offer a large number of templates, we ended up using mostly templates from SecurityIQ-PhishSim.

This tool offers a variety of templates with different themes including Banking, File Sharing, Personal Communication, etc., and served the purpose of this experiment.

We also wanted to verify how the AVs will treat these emails if we changed the nature of the URLs in these emails from redirecting to training websites, to linking to real live phishing websites. We will use the Pyphisher tool described in more detail in subsection 3.2.11 below, because it is easy to use in a script and has the function of replacing all the links in an email by a link specified as an argument. We downloaded 100 templates from PhishSim, and wrote a python script that uses Pyphisher to be able to not only automate the procedure of sending the emails to our virtual server, but also to be able to replace all the links in the emails with live phishing links from Phishtank. We made sure to activate only one anti-phishing tool at a time and to check if the anti-spam and anti-phishing add-ins of the tools are installed and active on Outlook.

Table 1 describes the results of the experiment. We show the number of original email templates that were added to the inbox, then the number of emails with live phishing links that were marked as spam or phish and thus added to the junk folder. For the second type of emails that were not marked as phish and added to the inbox, we also checked if the phishing websites would be detected at the browser level.

As an example, for Avast, all of the hundred email templates were added to the inbox. Whereas, when we replaced the original "fake" phishing link with a live one from Phishtank, 77 out of a 100 were added to the inbox and the other 23 were marked as spam. Out of the 23 emails that were marked as spam, 21 had phishing links that were detected at the browser level. As shown in Table 1, most of the anti-viruses do not consider the emails a threat as long as the URLs redirect to a training website from Phishsim. This is in fact a positive aspect, since it shows that employees will be able to learn without being hindered by AVs. However, this also hints to the fact that current detection methods rely too much on phishing website reputation to detect phishing emails, instead of also applying text analysis techniques on the body of the emails to help increase the detection rate.

This can be considered a huge flaw in phishing detection, especially in spear-phishing attacks where the attack is so targeted that the phishing website used will usually not be reported so as to be blacklisted until it is too late, and therefore the attack will succeed, since it will remain undetected by AVs. We will show an example of this in the next experiment (Subsection 2.5).

2.2 Building the Phish - Natural Language Generation

Now we show how to create phishing attacks on scale and study the effectiveness of the same five AVs at detecting them. For this experiment to be meaningful, we need a large number of phishing emails and phishing links. We decided to settle for a set of 1,000 generated phishing emails paired with 1,000 valid online phishing URLs reported on PhishTank⁴. To automate the generation of this number of emails, we used the Dada Engine, which is a tool based on NLG technology. Dada has been used before to generate books on postmodernism and also email masquerade attacks [4]. We used

³<https://www.hmailserver.com>

⁴www.phishtank.com

Anti-Virus Comparison on Training Templates					
Anti-Viruses	Avast	Kaspersky	AVG	Norton	ESET
Normal Emails added to Inbox	100/100	100/100	98/100	100/100	100/100
Emails + live Phishing link Marked as Junk	77/100	20/100	80/100	37/100	6/100
Phishing Links Detected in Browser	21/23	76/80	18/20	62/63	94/94

Table 1: Results of the first experiment, which used emails created using the templates of training tools

the phishing emails of the well-known Nazario dataset as a base for the structural properties to generate the emails.

Dada engine⁵ is based on the concept of a Recursive Transition Network, which is a graph that specifies the various ways a sentence can be constructed e.g Figure (1).

Figure 1: Example of sentence structure

If we take the example of an email subject field, then the sentence can be built following this structure: First the word "Subject" followed by a colon, then a sentence fragment that will be the subject of the email. The sentence can optionally feature the spoofed company's name. If we were to code this structure in Dada Engine, it would look something like Figure 2. With 'subject' the name of the rule or the structure, and 'company' another rule that randomly returns a company name from a list specified by the user. Note that 'cmp' is a variable that will get the result of the rule 'company,' and can be used later in the text, so as to not keep generating random company names and keep the text coherent.

```
subject: "Subject: " ["Information about your account" | cmp=company "Account Alert!"];
```

Figure 2: Dada engine code example

Using Dada Engine, we were able to create coherent emails with a fake sender email address like *it-team@ddns.company.com* or *it-team@bank.com*, and a subject that goes like this: *Confirm your*

⁵<http://dev.null.org/dadaengine/>

email account now! or *Notice from Company!* where 'Company' is replaced by the name of the spoofed company in the email. The body of the emails was composed of four parts: The first one is the greeting, e.g., 'Dear Customer,' 'Dear User' or 'Greetings from Company,' where again company is replaced by the spoofed company's name. The second part is the hook i.e a couple of sentences that explain the situation like: 'You have been chosen by Amazon Online Department to take part in our quick and easy reward survey. In return, we will credit \$100 to your account.' The third part was the bait. A couple of sentences to invite the recipient to click on the phishing link: 'We kindly ask you to spare two minutes of your time to take part in this unique offer! To continue please click the link below: ...'.

The phishing link was added later in the experiment using the phishing tool Pyphisher. The fourth and last part was the footer of the email, which was also coherent with the rest of the email: 'Company support' and/or 'Copyright 2017 Company, all rights reserved.'

```
<html>
Sender: IT.team@dns.amazon.com
Subject: Your Amazon store card is here!

<br><br> Dear Valued Customer <br>Thank you
so much for being such a loyal
customer in 2017. <br> Access the link with
following code to get your gift: <a
href="63.174.15.26">www.Amazon.com</a><br>
Code: 98473.<br><br>Regards,
<br><br>Amazon<br> Copyright 2017 Amazon.
All Rights Reserved<br><br></html>
```

Figure 3: Phishing email example generated with Dada engine

2.3 Adding Malware to the Mix

In this step, we describe two different methods of adding malware to an email. The first one is to create an attachment with a malicious payload. There are several ways to go about it. However, famous methods can't be used since security companies obviously know about those methods and are proud to offer protection in the shape of anti-viruses. For instance, we tried using SEToolkit or Metasploit framework to embed a malicious payload into PDF files, but this did not work. Not only AVs protect end-users from receiving those types of files, Gmail doesn't even allow users to send them. So we used one of the methods described in a personal security book [12] to generate a listener and a PowerShell script using PowerShell Empire⁶ and embed it in an Excel spreadsheet's macro.

PowerShell Empire is an intuitive and easy to install PowerShell post-exploitation framework that enables users to create listeners that handle multiple connections and create PowerShell agents that can be executed on the target's machine and offers various modules like KeyLogger, Mimikatz and privilege escalation, all of this over secure communication. For our experiment, we create a listener on the machine that will act as our server from which we will send phishing emails. Then we generate a stager, which is a PowerShell script encoded using base64 encoding, meant to be executed on the target machine and opens a reverse shell to our server. Then we used VBA to incorporate the stager into a random Excel sheet, and make it so as to be executed automatically when the sheet is opened.

The second way to add malware is to download available malware. We downloaded 100 malware from the DasMalwerk database⁷ and the malc0de database.⁸ There are other archives to download all kinds of malware from like VX Heaven,⁹ but since the discovery date of each malware was not reported, we preferred not to use them. We attached these malicious files to 100 emails from the dataset we generated. However, the original code of the open-source tool PyPhisher did not have the option to send an attachment with the phishing emails, so we had to alter the source code to add this function. All the malware we used were discovered at most one month before the experiment.

2.4 Building the Spear

To differentiate from the previous experiment, where we used random phishing links, we decided to create our own phishing website and use it in a spear-phishing scenario. We would like to show the flaw mentioned before related to the fact that most of the phishing website detection is based on reputation.

Detection based on reputation means that the more a website is reported, the easier it gets to be detected. But how about newly created phishing websites, meant to be used only for a specific attack like a spear-phishing attack. In this case, the malicious website will only be accessed by a few people who most likely won't report it even if they manage to detect it.

To simulate this situation, we created our own phishing website using SEToolkit, spoofing a random famous company: Bank Of

America. Obviously, we did not use it with 1000 emails, rather considering the scenario of a spear-phishing campaign, we only send one email with this link to the one recipient that we created.

The spear-phishing email can be spoofed using most of the tools surveyed in the next section. If the tool doesn't offer an HTML editor, then it is easy to find one online and then import the email using the import function of the tool.

Figure 4 is an example of a template of a phishing email created using Gophish. It spoofs an incoming email from Google, warning the user about an unknown sign-in.

It is worth pointing out that many of the people who may get these emails, if no sense of urgency or danger is induced in the recipients, then they will probably close the email and access the website directly via the browser later on.

Figure 4: Spoofed email from Google

2.5 Results of the second experiment

After generating the emails with Dada engine, we wrote a second python script that also uses Pyphisher, detailed below, to add the phishing links downloaded from Phishtank to the emails and send them to the recipient in our local mail server. Again, we made sure to activate only one anti-phishing tool at a time and check if the anti-spam and anti-phish add-ins of the tools are installed and active on Outlook. The results are as shown in table 2. Applying Fisher's Exact Test we find that at a significance level .05, the difference between Avast and AVG is not statistically significant, but all other pairs are statistically significantly different for the case of phishing emails with links. So for this case, {Avast, AVG} > Kaspersky > Norton > Eset, where > is "statistically significantly better."

A significant number of emails were not marked as spam/phishing. However it is still possible to catch phishing at the browser level.

⁶<https://www.powershellempire.com>

⁷www.dasmalwerk.eu

⁸www.malc0de.com

⁹www.vxheaven.org

Anti-Virus Comparison on Automatically Generated Phishing Emails					
Anti-Viruses	Avast	Kaspersky	AVG	Norton	ESET
Phishing Emails Detected	923	877	918	374	73
Phishing Emails Not Detected	77	123	82	626	927
Detected Malware	84(16)	47(11)	85(17)	40(16)	42(4)
Undetected Malware	16(6)	54(11)	15(5)	60(6)	58(18)
Spear-Phishing email	Undetected	Undetected	Undetected	Undetected	Undetected
Links Detected in Browser	40/50	30/37	30/50	49/50	49/50

Table 2: Results of NLG phishing experiment

These software tools have browser plugins and web protection that can be enabled to protect end users from malicious websites. So we decided to verify if the phishing links on the emails are flagged as malicious or not at the browser level. To do that, we selected randomly 50 active phishing websites not detected by the AV at the email level and verified if they are detected when we tried to access them.

Norton and SET performed the best in this experiment. Although the tools did not mark over 600 out of a 1,000 of phishing emails as malicious, most of the phishing links were properly detected at the browser level. For Avast out of 50 phishing links tested, 10 of them were not detected. For Kaspersky, Only 7 were not detected. Although we could not test 50 phishing links since most of the links that were not detected at the email level were not active. It is interesting to mention that only Kaspersky has an option for scanning emails for a list of sentences and keywords considered suspicious like: “product”, “customer” or “click here.”

Also, a decent number of malwares passed the barriers of these anti-viruses. These were not 0-day attacks, as previously mentioned, they were discovered a few days to at most one month prior to the experiment. We did not reverse engineer the malware nor the software to be able to explain these results as it is outside the scope of this study. However, we computed the md5 hash of each malware, and then we found out that the database downloaded from DasMalwerk contained a lot of duplicates. Only 22 out of a 100 were not duplicates. Then we analyzed the malware detected by the AVs, only to find out that some copies of the same malware were not detected, except for ESET that detected all copies. This raises the question: What was different in those files except the name since the duplicates have the exact md5 hash. If we only take the original 22 malwares into account, the number detected by AVs are put between parentheses.

Another interesting result is the fact that none of these tools detected the phishing website that we created. This strongly indicates that these AVs are not effective against 0-day attacks or a targeted spear-phishing campaign as suspected in the previous experiment. As a last conclusion to the experiment, all tools tested were able to detect the excel sheet with the malicious VBA code, which was not detected by Gmail nor Outlook.

3 ANTI-PHISHING TRAINING

Let’s consider the scenario where the top management of a company wants to reduce the company’s risk against phishing threats. One of the common solutions, other than installing anti-phishing filters, is to raise employees’ awareness and properly train them to detect and not fall victim to phishing emails. To do that, IT managers in charge of the project will look for anti-phishing training and awareness tools available on the market. This training can also be considered a social engineering based penetration testing. So what are the functions offered by these tools? What are the common ones and what differentiates each tool from the other?

To answer these questions, first, we will review the features that are available in phishing simulation tools. Then, we will try and give suggestions based on what facets we think can be added or improved in order to give employees and Internet users better training. This analysis is informed by the deception techniques used by phishers.¹⁰

3.1 Feature Review

After analyzing 10 anti-phishing training tools, which we will go over briefly in the next subsection, we created a list of criteria based on the common functionalities offered by these tools. The list is as follows.

¹⁰<https://pdfs.semanticscholar.org/5c7e/9ac9158b345591d8946ad0047d90da9531de.pdf>

License. This indicates whether a tool is free to use or not. Some licensed tools are free to set up but have limited functions enabled for non-premium users, such as a restricted number of targets, or being able to simulate only some types of phishing attacks. Other tools are open-source but can be lacking training modules.

Year Of Release gives an idea of how old is the tool.

Documentation refers to the availability of proper documentation. As first time users of these tools, we ran into various problems while installing and using some of them. As such, we believe that a well updated and available documentation is required to make the use of these tools easier and practical.

Phishing Campaign Types describes what kind of phishing campaigns are possible to build using the tool. *Drive-By attacks*, also called *hyperlink*, happen when the link in the phishing emails points to a website that triggers the immediate download of a malicious file. *Attachment type*, also called *File-Based*, is when various kinds of files (PDF, Excel sheets...) containing malicious payloads are attached to the phishing emails. *Data-Entry* or *Web-Based* is when victims are redirected to a website where they are led to fill in and submit personal information. *Awareness-only* means a simulation with no phishing (Victims are redirected to a harmless website where they are made aware of phishing). *Not Specified* means that there is no categorization of phishing campaigns in the tool. It gives the users freedom to create any type of campaign through an HTML editor. In the case of Social-Engineer Toolkit (SET), the tool offers two types of campaigns: *Spear-phishing* and *mass mailer attack*. As their name implies: The first is targeted to one victim and the second is for targeting multiple victims.

Available Templates and **Available Landing Pages** refers to whether a tool offers pre-built emails, and web pages (especially login pages) respectively, which spoof famous companies.

HTML Editor shows whether a tool has an available editor to build and edit emails or web pages using HTML. The tools that offer this feature make it possible to customize the phishing campaign to the need of the user. However, before doing that, the user has to change the encoding format since the email encoding system uses "quoted-printable," which allows non-ASCII characters to be represented as ASCII for email transportation.¹¹

Template creation is a feature that all of the tools in Table 3 allow, but not all of them allow **Template modification** or editing. In PhishSim for instance, it is limited since the tool doesn't allow to remove footer from email templates that indicates that it is a phishing simulation. This acts as a prevention mechanism in case an employee reports the email as a real phishing attempt, it also acts as a tell-tale sign meant to be detected by employees.

Different languages templates are provided by some tools. In PhishSim some templates are not in English to better spoof the targeted companies. Whereas for LUCY, every template is written in multiple languages including English. Wombat Security's ThreatSim also offers templates in a variety of languages.

Predefined Sender Domain is a feature that specifies whether users can use a random email address as the sender of the phishing email or not. Some tools restrict users to choose only from a limited list of email addresses or domains.

SMTP server specifies if users can use an SMTP server of their

choice to send emails or not. This feature can be useful for penetration testing local domains or running experiments on a local network, and it could also be interesting for attackers as demonstrated in the book [12] that teaches how to use Gophish (which has this feature enabled) to build a real spear-phishing campaign, in a well-crafted scenario of attacking a bank.

Website Cloning, as the name implies, refers to the function that enables to automatically copy a web page just by submitting the URL of the original page.

URL Redirection means the ability to redirect victims to the original (targeted) website after they submit their credentials into the spoofed web page. This is important so as to not raise user's suspicions.

Tracking Functions are essential if the user wants to be informed about the progress of the phishing campaign and collect information about the victims. Collected information can be the number of employees who participated in the simulation and who fell victim to the phishing email. For instance, how many clicked on the link in the email, downloaded the attachment, submitted information, their Geo-location, etc. This criterion is really important in the sense that it gives a detailed description of how users respond to phishing emails and what clues they can and can't detect.

Education Modules feature indicates if a tool offers a learning experience to users. The tools with this feature enabled are made not only for penetration testing but also to educate and raise employees' awareness of phishing threats.

Campaign Stats and Graphs takes collected information from tracking functions and displays it in graphs and numbers to make it easier to be read and understood by the user and easily explained to employees who participated in the simulation.

Finally, we add the following two rows: **Easy to Install**, which is self-explanatory, and **Rich on Features**, which summarizes how rich is the tool in terms of availability of features.

3.2 Multipurpose Tool Summaries

We discovered a list of anti-phishing tools on Google search¹² and added to that three more tools for different reasons: Famous and used by multiple companies (Wombat's ThreatSim), or resembles the tools in the list (FiercePhish), or used later in our study due to ease of use (PyPhisher).

We now give brief descriptions of the anti-phishing training tools.

3.2.1 SecurityIQ-PhishSim.¹³ is a web-based platform developed by Infosec Institute for the purpose of anti-phishing training and raising employees' security awareness. By login into this platform, the user has access to multiple functionalities including creating customized phishing emails or use already available templates, campaign scheduling, tracking functions and enabling different types of emails to be sent to different users in one campaign.

3.2.2 Gophish.¹⁴ is an open-source platform developed by Jordan Wright and launched in November 2013. It comes as an easy to build binary written in Go language. This tool is designed to enable

¹¹<http://stackoverflow.com/questions/4016067/whats-a-3d-doing-in-this-html>

¹²<http://resources.infosecinstitute.com/top-9-free-phishing-simulators>

¹³<https://www.infosecinstitute.com/security-iq>

¹⁴<https://getgophish.com>

penetration testers to launch phishing campaigns quickly and efficiently. It offers multiple features including importing email format and web page cloning, enabling users to easily spoof the target. The documentation is available with step-by-step instructions on how to install the tools and launch phishing campaigns.

3.2.3 LUCY Phishing Software.¹⁵ is a web-based tool that enables to raise phishing awareness of employees. It is free, but certain features are only available to premium users. Through the online platform, users have access to their personal dashboard where they can keep track of all their campaigns, or create new ones. Users can also set up lists of recipients to be used for phishing campaigns. LUCY offers multiple email templates, each of which can be used in multiple languages.

3.2.4 Simple Phishing Toolkit. (SPToolkit) is an open-source training tool to raise awareness and train employees to detect phishing emails. SPToolkit has not been actively developed for the past two years. The latest version called SPT (rebirth)¹⁶ was released in 2015. This tool offers features such as a site scraper that can clone any web page, and an HTML editor for editing or create new email templates. It also offers various options to track how victims respond to the phishing email, e.g., if the link was clicked, the date and time the link was followed, and the victims' IP address, browser, and operating system.

3.2.5 Social-Engineer Toolkit. (SET) is a famous penetration testing tool, written in python, developed by the founder of TrustedSec, and can be used to launch various social engineering attacks including spear-phishing. SEToolkit is available by default on Kali 2.0 and offers various attack vectors with different features including website cloning and payload generation.

3.2.6 KingPhisher. KingPhisher is an open source solution developed by SecureStare. This tool is a combination of a server-side for collecting harvested information, and a client side, which is a GUI made to help users create phishing campaigns. It offers various features that enable users to simulate advanced campaigns like harvesting credentials from web pages, cloning web page abilities, send emails with calendar invitations and track geo-location of victims.

3.2.7 SpeedPhish Framework. SpeedPhish Framework¹⁷ is a python tool created by Adam Compton. This tool is meant to be used for phishing training and exercise. The installation is very easy, and the commands to do it are available in the GitHub repository. This tool is only available on Linux systems. One of the useful features of this tool is the *recon* function that enables searching popular search engines or uses The Harvester to find potential targets. It also has a built-in web server based on Twisted python library, and offers cloning functions of login web pages.

3.2.8 Phishing Frenzy.¹⁸ is an open-source Ruby on Rails application for a penetration tester to send phishing emails. Developed in 2013 by Brandon "zeknox" McCann, it facilitates phishing campaign management. Features include available templates, website

cloning, credential harvesting, campaign statistics, and exporting results in XML or PDF formats. It has no HTML editor to change or create templates but it is possible to upload available templates in Github.

3.2.9 Wombat Security - ThreatSim.¹⁹ ThreatSim is a web platform for building phishing campaigns integrated with various training modules, acquired by Wombat Security on October 14, 2015.²⁰ There is no free version of ThreatSim. It offers more than 130 templates updated almost weekly, in more than 25 languages. The phishing email campaign types are varied between drive-by, data entry, and email with attachment types. It also offers multiple features including not only website cloning function and HTML editor to edit email and website templates, but also plugin detection that detects if certain plugins and add-ons like Java, Quicktime or Adobe Flash are up to date when the user clicks on the link inside the email.

3.2.10 SpearPhisher Beta. SpearPhisher Beta²¹ is a windows based program with a simple and easy to use GUI, developed by TrustedSec for non-technical users. No installation is required, just download and run the executable. It enables users to easily create and send phishing emails to one or multiple targets. It also offers an HTML editor to be used by advanced users for the possibility to add malicious payloads or exploits to emails.

3.2.11 PyPhisher.²² is python script created by sneakerhax in 2014. This tool is not displayed in Table 3 due to a lack of features, as compared to other tools, including not offering an Html Editor, nor templates, nor education modules. However, we cover this tool in this subsection because we used it for our experiments in the previous section. Pyphisher is a python script, that is run with a maximum of eight arguments, where the user can specify the SMTP server to be used to relay the emails, specify the path of the HTML file to use as email, choose the subject and replace all the links in the email with a URL of the user's choice, and, specify the sender and the recipient of the phishing email. The fact that this tool is minimalist and intuitive, makes it easy to incorporate in a script or used in a loop to send a high number of phishing emails.

3.3 Tool Analysis

In this subsection, we put our analysis of the tools based on the features presented in Section 3.1 in Table 3. To remind the readers, the list of features covers common functionalities offered by the anti-phishing training tools. A review of Table 3 shows that almost all of the tools we analyzed have available documentation and that they give the user certain the freedom in terms of template creation and editing. PhishSim, for example, has a limited editing option as it is not possible to remove the footer showing that the email is part of training campaign and not a real threat. Whereas, Gophish gives absolute freedom in email creation but does not offer pre-made templates.

Almost all of the tools let users choose a specific SMTP server to relay the emails. This can be a dangerous functionality to have

¹⁵<http://www.lucysecurity.com>

¹⁶<https://github.com/0x90shell/sptoolkit-rebirth>

¹⁷<https://github.com/tatanus/SPF>

¹⁸<https://www.phishingfrenzy.com>

¹⁹<https://www.wombatsecurity.com/about/our-company>

²⁰<http://www.csoonline.com/article/2993066/security-awareness/wombat-acquires-threatsim-to-extend-security-awareness-training-capabilities.html>

²¹<https://www.trustedsec.com/2013/09/introducing-spearphisher-simple-phishing-email-generation-tool/>

²²<https://github.com/sneakerhax/PyPhisher>

enabled since users can choose any open relays to send emails, making these anti-phishing training tools usable by hackers to send real phishing campaigns [12], especially if the user has the aforementioned freedom of template editing. PhishSim and Wombat Security's ThreatSim handle the sending of the emails on their side, thus relieving the user or the company from this task.

The tracking functions are important since they report back on how the campaign resolved after being sent. It's an important aspect of the training tools and can help managers know who are the employees who need more training and what are their sticking points.

After studying varied types of work in the literature on phishing email detection, we noticed that phishing clues can reside in every part of the email (Header, body and links). Based on that we would suggest that anti-phishing training tools, not only the ones mentioned in this paper, to focus on certain aspects of end user based detection and awareness like the following:

- (1) Ensuring that an employee knows how to view the full header of an email, whenever it is available and is able to find discrepancies in an attack email header when they exist.
- (2) The ability to detect whether an employee just skims a suspicious email or spends time and effort in analyzing it.
- (3) The ability to detect whether an employee hovers the mouse over links without clicking them to see if there are any discrepancies in the links of an attack email.
- (4) The ability to detect whether an employee searches the inbox or sent mailbox to find similar emails upon receiving an attack email.

One row deserves special mention. Even though most of the entries in the Tracking Functions row are positive, we note that the tracking provided is quite rudimentary.

4 RELATED WORK

Phishing has been intensively studied over the last decade, and researchers have tackled several different aspects of this threat including: (i) detection of phishing URLs, websites, emails, (ii) user studies and (iii) training.

Anatomy of phishing emails: In the literature, we can find various studies about the anatomy of phishing threats. An example would be Drake et al. [10], albeit more than a decade old now, who covered in their work the basic tricks and techniques used by hackers to create a phishing email and web page with high success probability.

Phishing Kits and dropboxes: Other studies focused on collecting phishing footprints, like phishing emails or toolkits, left by phishers, to analyze and try to come up with solutions. For instance, Maggi et al. [24] developed a system, consisting of bots, that collects data from all kind of communication channels namely emails, instant messages, phone calls and spam sent in social networking websites by suspicious accounts spotted by SpamDetector [45]. Moore et al. [25] developed a method to automatically detect "dropboxes," which are accounts owned by phisher used to collect credentials from their campaigns. They also cover phishing kits, which are PHP codes uploaded on vulnerable web servers that build a web page spoofing a targeted company and send submitted credentials to dropboxes. In a recent work, Han et al. [17] created

a sandbox which contains a honeypot server that hosts 18 vulnerable and exploitable PHP pages. After finding these pages using search engines, hackers upload their phishing kits that are then collected and analyzed. The system was configured so as to protect the privacy of the victims who submitted their credentials. Another work that also gave importance to participants' privacy during the experiment is [20]. A similar work to Han et al. was done by Cova et al. [8], where the authors analyzed a little over 500 phishing kits that were either available on distribution sites or collected from live phishing websites, reported in PhishTank, that allowed directory listing. The goal was to analyze the techniques used in these kits to gather and send information to phishers and also how the attackers who develop these phishing kits obfuscate backdoors. One of their major discoveries was that 37% of all phishing kits contain backdoors in the form of hidden code that exfiltrate phished information to other recipients.

Phishing Email Detection: Another prominent line of phishing research is on detection mechanisms for phishing emails or phishing web pages. There is a vast literature here and since it is not directly relevant to this paper, we just mention a few of them here. Chandrasekaran et al. [7] used what they called "structural features" as in most phishing emails are made using certain characteristics. By using them with a one-class support vector machine (SVM), they were able to detect 95% of phishing emails with very few false positives. Other studies have combined a blacklist of words that are usually contained in a phishing email with other features from the header. This kind of feature selection gives a satisfying result [15]. For analyzing the content of emails and extract features, Natural language Processing (NLP) techniques like parsing, stemming and stopword removal were used in various studies [7, 11, 23, 24, 35, 36, 38, 40, 42, 44]. Many studies relied not only on the body but also the header of the email and also the URLs inside the body to extract features and use them to build classifiers. Peng et al. [19] and Madhusudhanan et al. [7] both used SVMs and both had detection rates over 90%. Khonji et al. [21] tested various classifiers using 48 features, and before running their classifier, they used the "Wrapper subset evaluation criterion" and "Best-first: Forward" which selects the best features from a set of features. They ended up with 6 subsets of features. The classifier that ends up giving the best results was Random Forest: A precision higher than 99% for 5 of the 6 subsets.

The first anti-phishing shared task pilot was organized at the 2018 ACM IWSPA [41] with a separate proceedings [38] and two sub-tasks: phishing detection without header and phishing detection with header. The datasets for this shared task are publicly available.

User Studies and Training: There have been many studies of user behavior, e.g., see [3, 26, 34] and references cited therein; and many companies, e.g., PhishMe and Wombat Security, are involved in developing training regimes for employees.

5 CONCLUSION

In this paper, we showed how to semi-automatically generate attacks and tested five famous anti-viruses against templates from anti-phishing training tools and automatically generated phishing emails. We discovered that the protection they offer is still not

Tool Comparison									
Tool (Release Date)	PhishSim (2015)	Gophish (2013)	LUCY (2015)	Simple Phishing Toolkit (2012)	King Phisher (2014)	Social-Engineer Toolkit (2010)	FiercePhish (2017)	Phishing Frenzy (2013)	ThreatSim (2015 ²³)
License	Open-Source / Premium	Open-Source	Open-Source / Premium	Open-Source	Open-Source	Open-Source	Open-Source	Open-Source	Licensed
Documentation	Available	Available	Available	Not Available	Available	Available	Available	Available	Available
Phishing Campaign Types	Drive By/ Attachment/ Data-entry	Not Specified	Awareness Only/ Web Based/ File-Based/ Hyperlink	Not Specified	Not Specified	Spear-Phishing / Mass mailer attack	Not Specified	Not Specified	Drive By/ Data-entry/ Attachment/ classic Attachment
Education Modules	Yes	No	Yes	Yes	No	No	No	No	Yes
Available Templates	156	None	Available for Download	13	None	7	None	4	130
Template Editing	Limited	Yes	Yes	Yes	Yes	No	Yes	No	Yes
Different Languages Templates	Language different with template	n/a	Each template with different languages	No	n/a	No	n/a	No	+25
Predefined Sender @	Yes	No	No	No	No	No	No	No	No
SMTP server	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Available Landing Pages	36 Data Entry Pages	import web page is enabled	Yes	None	None	None	None	4	+50
HTML Editor	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes
Website Cloning	No	Use HTML editor	Use HTML editor	Yes	Yes	Yes	None	Yes	Yes
URL Redirection	No	Yes	Yes	No	No	Yes	No	No	Yes
Tracking Functions	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
Campaign Stats and Graphs	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
Easy to setup	Yes	Yes	Yes	No	Yes	Yes (Available on Kali 2.0)	Yes	Yes on Ubuntu 16.04 LTS	Yes
Rich on Features	Yes	Moderate	Yes	Moderate	Moderate	Yes	Moderate	Moderate	Yes

Table 3: Comparing Nine Tools on 19 criteria

perfect and relies too much on the reputation of phishing websites. This also proves that user training is essential and that companies should not feel safe just by installing or upgrading their anti-virus software. Instead, they should invest more in training and in keeping their employees aware of the current cyber threats.

We also analyzed several phishing simulation tools, and showed the different functions they offer for training Internet users and employees. We also suggested some aspects of improvement especially related to tracking the behavior of the users and their interaction with the emails. The list of tools covered is not intended to be exhaustive, but it is diverse and gives a good idea of what is the state of phishing frameworks in the current cybersecurity landscape.

ACKNOWLEDGMENTS

This research was supported by NSF grants CNS 1319212, DGE 1433817, DUE 1241772, and DUE 1356705.

REFERENCES

- [1] Andronicus A. Akinyelu and Aderemi O. Adewumi. 2014. Classification of phishing email using random forest machine learning technique. In *Journal of Applied Mathematics*.
- [2] Ammar Almomani, B. B. Gupta, Tat-Chee Wan, Altyeb Altaher, and Selvakumar Manickam. 2013. Phishing Dynamic Evolving Neural Fuzzy Framework for Online Detection Zero-day Phishing Email. *Indian Journal of Science and Technology* (2013).
- [3] Mohamed Alsharnouby, Furkan Alaca, and Sonia Chiasson. 2015. Why phishing still works: User strategies for combating phishing attacks. *International Journal of Human Computer Studies*, 69–82.
- [4] Shahryar Baki, Rakesh Verma, Arjun Mukherjee, and Omprakash Gnawali. 2017. Scaling and Effectiveness of Email Masquerade Attacks: Exploiting Natural Language Generation. In *Proc. 12th ACM ASIACCS, 2017, April, 2017, Abu Dhabi, UAE*.
- [5] André Bergholz, Jan De Beer, Sebastian Glahn, Marie-Francine Moens, Gerhard Paaß, and Siehyun Strobel. 2010. New filtering approaches for phishing email. *Journal of Computer Security* 18, 1 (2010), 7–35.
- [6] Manasi Bhattacharyya, Matthew G. Schultz, Shlomo HersHKop Eleazar Eskin, and Salvatore J. Stolfo. 2002. Met: An experimental system for malicious email tracking. In *NSPW*.
- [7] Madhusudhanan Chandrasekaran, Krishnan Narayanan, and Shambhu Upadhyaya. 2006. Phishing E-mail Detection Based on Structural Properties. In *NYS Cyber Security Conference*.
- [8] Marco Cova, Christopher Kruegel, and Giovanni Vigna. 2008. There is No Free Phish: An Analysis of “Free” and Live Phishing Kits. In *WOOT’08 Proceedings of the 2nd conference on USENIX Workshop on offensive technologies*.
- [9] Priyanka Deshmukh, Megha Shelar, and Nikhil Kulkarni. 2014. Detecting of targeted malicious email. In *GCWCN, IEEE*.
- [10] Christine E. Drake, Jonathan J. Oliver, and Eugene J. Koontz. 2004. Anatomy of a phishing email. In *CEAS*.
- [11] Ayman El Aassal, Luis Moraes, Shahryar Baki, Avisha Das, and Rakesh Verma. 2018. Anti-Phishing Pilot at ACM IWSPA 2018: Evaluating Performance with Metrics for Unbalanced Datasets. In *Proceedings of the 1st Anti-Phishing Shared Task Pilot at 4th ACM IWSPA, Vol. 2124*.
- [12] Spark FLOW. 2017. *How to Hack Like a Pornstar, a Step by Step Process for Breaking into a Bank*.
- [13] Eibe Frank and Ian H. Witten. 1998. Generating accurate rule sets without global optimization. University of Waikato, Department of Computer Science.
- [14] Hong Guo, Bo Jin, and Wei Qian. 2013. Analysis of Email Header for Forensics Purpose. In *Communication Systems and Network Technologies (CSNT), 2013 International Conference on, IEEE*, 340–344.
- [15] Isredza Rahmi A Hamid and Jemal Abawajy. 2011. Phishing email feature selection approach. In *International Conference on Trust, Security and Privacy in Computing and Communications, IEEE*.
- [16] Isredza Rahmi A Hamid and Jemal H. Abawajy. 2013. Profiling Phishing Email Based on Clustering Approach. In *International Conference on Trust, Security and Privacy in Computing and Communication, IEEE*.
- [17] Xiao Han, Nizar Kheir, and Davide Balzarotti. 2016. PhishEye: Live Monitoring of Sandboxed Phishing Kits. In *CCS ’16 Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*.
- [18] Amir Herzberg. 2009. DNS-based Email Sender Authentication Mechanisms: a Critical Review. *computers & security* 28, 8, 731–742.
- [19] Peng Hong, Wang Jun, Tiefeng Wu, and Dongna Zhang. 2004. A malicious email detection method based on support vector machine. In *International Conference on Control Automation Robotics and Vision*.
- [20] Markus Jakobsson and Jacob Ratkiewicz. 2006. Designing ethical phishing experiments: a study of (rot13) rnl query features. In *International conference on World Wide Web*.
- [21] Mahmoud Khonji, Youssef Iraqi, and Andrew Jones. 2011. Lexical url analysis for discriminating phishing and legitimate e-mail messages. In *ICITST*.
- [22] Mahmoud Khonji, Andrew Jones, and Youssef Iraqi. 2011. A study of feature subset evaluators and feature subset searching methods for phishing classification. In *The 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference, CEAS 2011, Perth, Australia, September 1-2, 2011, Proceedings*. 135–144.
- [23] Michael C. Kotson and Alexia Schulz. 2015. Characterizing Phishing Threats with Natural Language Processing. In *Communications and Network Security (CNS), 2015 IEEE Conference on, IEEE*, 308–316.
- [24] Federico Maggi, Alessandro Sisto, and Stefano Zanero. 2011. A social-engineering-centric data collection initiative to study phishing. In *BADGERS*.
- [25] Tyler Moore and Richard Clayton. 2012. Discovering phishing dropboxes using email metadata. In *eCrime*.
- [26] Ajaya Neupane, Md. Lutfor Rahman, Nitesh Saxena, and Leanne Hirshfield. 2015. A Multi-Modal Neuro-Physiological Study of Phishing Detection and Malware Warnings. In *ACM CCS*.
- [27] Kathryn Parsons, Marcus Butavicius, Malcolm Pattinson, Agata McCormac, Dragana Calic, and Cate Jerram. [n. d.]. Do Users Focus on the Correct Cues to Differentiate Between Phishing and Genuine Emails?
- [28] Satheshaan Pasupatheeswaran. [n. d.]. Email ‘Message-IDs’ helpful for forensic analysis?. In *Australian Digital Forensics Conference*.
- [29] Doyen Sahoo, Chenghao Liu, and Steven C. H. Hoi. 2017. Malicious URL Detection using Machine Learning: A Survey. *CoRR abs/1701.07179* (2017).
- [30] Fernando Sanchez and Zhenhai Duan. 2012. A Sender-Centric Approach to Detecting Phishing Emails. In *Cyber Security (CyberSecurity), 2012 International Conference on, IEEE*, 32–39.
- [31] Tanmay Thakur and Rakesh M. Verma. 2014. Catching Classical and Hijack-Based Phishing Attacks. In *Information Systems Security - 10th International Conference, ICISS 2014, Hyderabad, India, December 16-20, 2014, Proceedings*. 318–337.
- [32] Fergus Toolan and Joe Carthy. 2010. Feature selection for Spam and Phishing detection. In *2010 eCrime Researchers Summit, eCrime 2010, Dallas, TX, USA, October 18-20, 2010*. 1–12.
- [33] Gaurav Varshney, Manoj Misra, and Pradeep K. Atrey. 2016. A survey and classification of web phishing detection schemes. *Security and Communication Networks* 9, 18 (2016), 6266–6284.
- [34] Rakesh Verma, Devin Crane, and Omprakash Gnawali. 2018. Phishing During and After Disaster: Hurricane Harvey. In *2018 Resilience Week (RWS), IEEE*, 88–94.
- [35] Rakesh Verma, Narasimha Shashidhar, and Nabil Hossain. 2012. Detecting Phishing Emails the Natural Language Way. In *Computer Security - ESORICS 2012 - 17th European Symposium on Research in Computer Security, Pisa, Italy, September 10-12, 2012, Proceedings*. 824–841.
- [36] Rakesh M. Verma and Ayman El Aassal. 2017. Comprehensive Method for Detecting Phishing Emails Using Correlation-based Analysis and User Participation. In *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy, CODASPY 2017, Scottsdale, AZ, USA, March 22-24, 2017*. 155–157.
- [37] Rakesh M. Verma and Avisha Das. 2017. What’s in a URL: Fast Feature Extraction and Malicious URL Detection. In *Proceedings of the 3rd ACM on International Workshop on Security And Privacy Analytics, IWSPA@CODASPY 2017, Scottsdale, Arizona, USA, March 24, 2017*. 55–63.
- [38] Rakesh M. Verma and Avisha Das (Eds.). 2018. *Proceedings of the 1st Anti-Phishing Shared Task Pilot at 4th ACM IWSPA, Vol. 2124*. <http://ceur-ws.org/Vol-2124/>.
- [39] Rakesh M. Verma and Keith Dyer. 2015. On the Character of Phishing URLs-Accurate and Robust Statistical Learning Classifiers. In *ACM CODASPY*.
- [40] Rakesh M. Verma and Nabil Hossain. 2013. Semantic Feature Selection for Text with Application to Phishing Email Detection. In *Proc. 16th Int’l Conf. on Information Security and Cryptology, Springer LNCS vol 8565*.
- [41] Rakesh M. Verma and Murat Kantarcioglu (Eds.). 2018. *Proceedings of the Fourth ACM International Workshop on Security and Privacy Analytics, IWSPA@CODASPY 2018, Tempe, AZ, USA, March 19-21, 2018, ACM*. <http://dl.acm.org/citation.cfm?id=3180445>
- [42] Rakesh M. Verma and Nirmala Rai. 2015. Phish-IDetector: Message-Id Based Automatic Phishing Detection. In *SECRYPT, IEEE Xplore*.
- [43] Paul Watters, Simon Brown, Rosemary Torney, and Liping Ma. 2009. Automatically generating classifier for phishing email prediction. In *International Symposium on Pervasive Systems, Algorithms, and Network*. 779–783.
- [44] Masoumeh Zareapoor and Seeja K. R. 2015. Feature Extraction or Feature Selection for Text Classification: A Case Study on Phishing Email Detections. In *Engineering and Electronic Business*.
- [45] Xianghan Zheng, Zhipeng Zeng, Zheyi Chen, Yuanlong Yu, and Chunming Rong. 2015. Detecting spammers on social networks. *Neurocomputing* 159 (2015), 27–34. <https://doi.org/10.1016/j.neucom.2015.02.047>