Phishy

by

Travis Waters, Derrek Mayse, and Jacob Thomas

Submitted to
the Faculty of the School of Information Technology
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Technology

Travis Waters	25 - April - 2021
Travis Waters	Date
Derrek Mayse	25 - April - 2021
Derrek Mayse	Date
Jacob Thomas	25 - April - 2021
Jacob Thomas	Date
Tony Iacobelli	25 – April – 2021
Tony Iacobelli, Faculty Advisor	Date

University of Cincinnati
College of
Education, Criminal Justice, and Human Services

April 25, 2021

Table of Contents

List of Tables and Figures:

Abstract

Phishy is a service intended for security administrators in order to allow them to launch several simultaneous, simulated phishing campaigns to gauge and improve the phishing awareness of employees within their organization. With our service, we want to take security awareness training to the next level. We designed *Phishy* to help combat the major phishing problem companies face today and to help better educate employees on the dangers of phishing. Currently, there is not an open-source platform that acts as a "one stop shop" for managing phishing campaigns and users' phishing awareness training. *Phishy* fills this need by helping its users launch realistic phishing campaigns, utilizing WGET to clone websites in order to create realistic phishing situations. Along with giving users the ability to create practical phishing training for their organization, live visual and data analytics will be provided with tools such as Elasticsearch and Kibana. Since we know how busy security professionals are, it would be unlikely that each user who falls for a phishing simulation would be able to receive security training catered to what phishing tactic they fell for. *Phishy* has the ability to track which users are falling for which phishing scams and enroll them into specific phishing awareness training based on their interactions with the phishing simulation. Our hope is that *Phishy* will provide security professionals with a comprehensive tool that will allow them to better train and prepare their organization against the dangers of phishing.

Introduction

There is a new phishing site launched every 20 seconds, and 74% of phishing websites use HTTPS protocol (Crane, 2020). With the rate at which technologies change and different scamming methods emerge, it is not enough for a company to hold a couple of sessions of security awareness training every year. Cyber security training needs to be conducted regularly in order to ensure employees are up to date on the threats facing their company and even their personal devices. Based on the research that we have done up to this point, there is not currently a tool that fulfills both the needs of security administrators and is beneficial to employees. A solution that allows security administrators to plan and carry out simulated phishing campaigns, and organize a response based on the results would help organizations stay safe from phishing attempts.

. Problem:

According to the 2019 Data Breach Investigations Report (DBIR), the leading cause of data breaches are phishing attacks. It is estimated that 32% or more of breaches involved phishing in some way (2019 DBIR Summary of Findings). This is no surprise, especially in a time where organizations heavily rely on email to conduct their day-to-day operation. To properly protect an organization from phishing attacks, employees need to be able to identify well-orchestrated phishing attempts, and security administrators need to be able to gauge the susceptibility of the organization's employees when they experience an attack. By having well educated and trained employees, a crucial extra layer of security will be added for any company. In our

opinion, the best way to accomplish this is to give users safe and practical real world experience dealing with phishing emails in addition to regular security awareness training.

. **Solution:**

Phishy is a solution that allows security administrators to carry out realistic, simulated, phishing campaigns against their end-users. Administrators are given the ability to clone websites to create realistic phishing campaigns and are able to access real-time analytics around their campaign. When users fall for one of Phishy's campaigns, not only will administrators be able to see who fell for it, but the users will also be automatically enrolled in phishing awareness training courses based on their interactions with the simulated phishing campaign.

. **Project Goals:**

- Allow security administrators to clone websites for realistic phishing situations
- Make it easy to send out and manage simulated phishing emails to users
- Provide a catalog/database for holding data pertaining to user interactions with phishing emails
- Real-time visualizations and data trends depicting results from active phishing simulations
- Categorize types of phishing emails sent out and track what users are struggling with

- Automatically distribute phishing awareness training to users depending on their interactions with phishing email

. Overview:

Immediately following in this report will be a detailed discussion on the development process of Phishy. Reviewing topics such as the goals of the project, user profiles, technical architecture, budgeting, and a project timeline. In addition to these topics will also be an analysis of problems that arose along the way and solutions to those problems, as well as future recommendations and ways that Phishy could be improved with further development.

Discussion

. Project Concept:

Phishy was brought into fruition as our team realized that we all shared a passion for cybersecurity and the desire to protect people from cyber threats. Through research and real-world experience, we have found that different types of phishing attacks are some of the most common and successful ways that malicious actors deceive people. We looked at several other tools that allow administrators to test their users' phishing awareness with simulated phishing campaigns, and found that there are other solutions, but they do not offer phishing awareness training as a part of the same solution. We also plan on giving administrators a deeper look into what types of phishing tactics people are struggling with, providing specific training based on those struggles. *Phishy* offers phishing simulations, as well as more useful statistics to assist administrators. When a user falls for a campaign carried out with *Phishy*, they are automatically notified and enrolled in phishing awareness training based on their interactions with the phishing email. Our solution will help reduce the planning and complexity required to ensure that employees are properly trained in phishing awareness.

. Design objectives:

Our main project objective is to create a phishing email simulation service that allows administrators to create phishing campaigns that target their employees. One of our goals is to make this process as easy as possible for the admin. They will be able to clone websites to use in phishing simulations and format the phishing emails to suit their

needs. We will also be including features that allow for bulk uploading of employee information that can be used to create many different targets at once. Another of our goals is to be able to allow the admin to actually see and use the results of the campaigns that were sent out. That is why we have designed Phishy to visualize the important data that is coming in from the interactions between employees and the simulated phishing attacks. After sending out the simulated phishing emails to any number of employees, the administrator will then be able to see real-time, detailed information on how the campaign is going for each user, as well as any trends that are occurring with the users. Since each type of phishing email that was sent out will fall under a certain phishing scam category, the security administrator can use *Phishy* to automatically assign each employee specific training based on if they were deceived by the simulation. Phishy was designed to provide all of these features in one tool that security administrators can utilize to more efficiently train their employees on phishing safety.

. Methodology:

There were several different design requirements that were considered when figuring out how we could meet our goals for this project. As mentioned above, some of the most important goals for *Phishy* were:

- Providing system administrators a simple way to give their employees real time training against phishing
- The administrator could visualize and use relevant data from the tool
- Training information could be easily sent out and managed

In order to meet our first primary goal, *Phishy* had to be well documented and give administrators access to a simple interface focused on only the necessary core functions of the tool. For this reason, a command line-based interface was chosen with options and commands clearly presented to the administrator. The interface lists out clear instructions for the administrator to complete all the tasks necessary to send out realistic phishing campaigns. Another key part of *Phishy's* design is that fake web pages and phishing templates are easily created for use by the tool. This was done through automatic web page cloning and pre-built templates to pick and choose from. In order to really make this an easy experience for admins, these were major functions we included while brainstorming design requirements. The second of our goals was met through another design requirement that dealt with utilizing information gathered from simulations and visualizing it. In order to achieve this, *Phishy* was required to integrate some sort of visualization and monitoring interface. It was decided that ELK would be used as part of the project design to accomplish this goal. Through a series of scripts, *Phishy* utilizes logs (information) gathered from when users fall for a phishing attack. These logs are formatted and sent along to Kibana where several pre-formatted visualizations have been created for this project. These visuals show which users click on what phishing simulations, how many users and when are falling for the simulation, etc. Filters can be applied to narrow down the visualizations even further, giving administrators a wide range of flexibility. This design is what allows the admins to use data collected from their simulations to create reports or gauge employee awareness. To accomplish the third of our main goals for *Phishy*, there needed to be a means for training to be sent out relatively quickly and with little hassle. This tool would be wasting

a lot of potential if phishing campaigns were sent out but there was no follow up training or useful information to help improve user awareness in areas they are struggling. For this reason, another design requirement was that administrators would be able to send out training to all users who fell for a simulation with the click of a button. This is accomplished through our simple interface and can be done after a campaign has ended by automatically acquiring who fell for the attack.

Through these design requirements that were carefully planned and discussed throughout this project, *Phishy* was able to accomplish the desired goals it was created to meet. Simplicity, usefulness, and employee improvement-oriented functions were key design requirements that were needed to make this project a success.

. User Profiles:

User Profile Form 1

Application:

Phishy: Apache, ELK, and Linux command line.

Potential Users:

Security administrators

Software and Interface Experience:

The user should be comfortable operating Linux from the command line. They should also have some familiarity with ELK, as well as the knowledge to properly configure services.

Experience with Similar Applications:

Familiarity with full text search engines and data visualization tools is ideal. Experience launching phishing campaigns is not required, but must be familiar with phishing as a concept. Must be comfortable with Linux.

Task Experience:

The system administrator will need to know how to maintain ELK and Apache servers in order to operate Phishy. It is also necessary for this user to know how to generate the simulated phishing emails that are sent to other users.

Frequency of Use:

How often the user interacts with the solution is entirely dependent on their needs. The user will have to run the script to stage a new Apache instance any time they wish to start a new phishing simulation. Outside of that, they

will be using it to view current/past phishing campaigns and maintain their Elastic Stack however frequently they deem necessary.
Key Interface Design Requirements that the Profile Suggests: Security administrators will need to be able to maintain and use an Elastic Stack, and run scripts from the Linux command line.

Table 1: The table above outlines expected usage information for the administrator/system owner user profile

User Profile Form 2

Application:

Users would only interact with Phishy through receiving simulated phishing emails through their Email client. They would also receive training sent from Phishy, which would be completed in a web browser.

Potential Users:

Simulation Targets/Employees

Software and Interface Experience:

Email inboxes, Web browsers, Media platforms

Experience with Similar Applications:

Google Chrome, Safari, FireFox, Internet Explorer, Outlook, Youtube

Task Experience:

The users will need to know the basics of how to interact with emails and inboxes. This includes navigating to an email and opening it, as well as clicking on links included in the email (if they are to fall for the phishing attempt). While going through the training they will need to operate in a web browser.

Frequency of Use:

Whenever a user is targeted with a simulated phishing campaign and/or go through the training that is provided. The frequency would be up to the system admin of Phishy.
Key Interface Design Requirements that the Profile Suggests: The way that the training is sent out needs to be clear with the users who have interacted with the simulated phishing emails. We might send the user an email first stating that they have been enrolled in specific training with clear instructions on how to proceed.

Table 2: The table above outlines expected usage information for the simulation targets user profile

. Use Case Diagram:

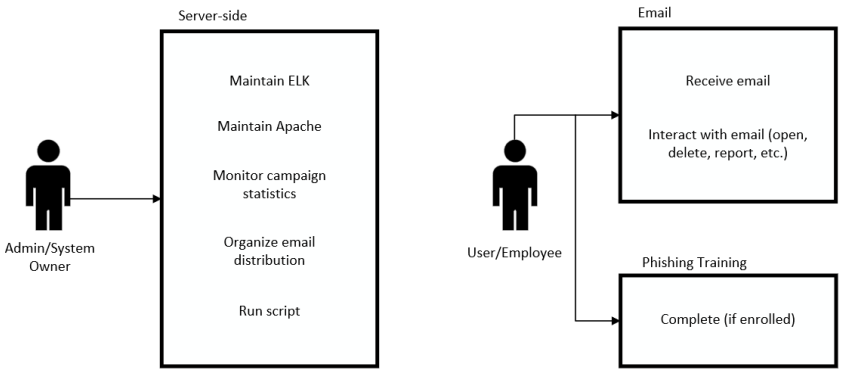

Figure 1: The use case diagram above outlines the various interactions that each type of user would have with Phishy.

. Technical Architecture:

Our team chose the technologies we decided to use based on ease of use, scalability, and integration with one another. Many of these tools are not very beginner friendly, but they have some excellent documentation and great community support, which makes them very easy to learn. The nature of these tools require that they are maintained, and are not “set it and forget it” tools. In order to keep the solution working properly, a technical understanding is essential.

Infrastructure:

We decided to host our solution on Amazon Web Services (AWS). Using AWS gives us an extreme amount of flexibility in terms of scaling our architecture up/down as needed, easy access from anywhere for our entire team, and excellent integration between tools. IAM and EC2 allows anyone on our team to access all of our resources

from anywhere. EC2 also has great integration with Amazon's Simple Email Service (SES) which we are using to send emails containing links to our phishing campaigns, and the resulting phishing awareness training.

Database:

Our database is being built on the Elastic Stack (ELK). The use of ELK (Elasticsearch, Logstash, and Kibana) gives us an easy to use, manage, and extremely customizable database. This allows us to have multiple active phishing campaigns, and keep them separate, as well as provide easy to read visualizations providing live statistics on active phishing campaigns. ELK is accessible through a web browser, making interacting with it very easy.

Websites:

Hosting realistic websites is the backbone of our solution. We use WGET to copy web pages, which are then hosted internally with Apache to conduct our simulated phishing campaigns. Apache's logging makes tracking which users are interacting with phishing emails easy, and its integration with ELK is well documented, as the two are very commonly used together.

. Testing:

. Methodology:

For testing, we will be utilizing previous information that we have compiled as well as anything that is given from the results of using our project. The previous

information includes User Profiles, Use Cases, User Scenarios and our design objectives for the project. This information along with what we have learned through the creation of our project gives us the idea of how best to formulate a testing methodology. We know that there will be two main types of users who interact with Phishy, an administrative user and an end user (someone who is receiving the phishing simulation). With this information we can use Phishy from each user's perspectives by giving ourselves different levels of access based on real scenarios. This will allow us to test features that each type of user should have, and ensure they are working as they should. We will conduct these tests by using our pre-made Use Cases and User Scenarios, which cover all the ways users will interact with Phishy. With careful documentation during this testing method, we believe this approach will allow us to test Phishy from every angle and help us realize if anything needs to be updated.

. Plan:

The Test Plan for Phishy is to run the Testing through the Administrator's view point and have users supply input by either falling for the phishing attempt or not falling for it to supply Phishy and to verify that every aspect of the program is working. The plan can be shown in Table 3.

. Results:

The results from the plan can be found in Table 4 below.

Step ID	Steps	Test Date	Expected Results	Actual Results	Pass / Fail	Comments
1	User, with Administrator Access, runs the “siteDownlaod.sh” file.	2021-03-15	The “siteDownlaod.sh” file runs without throwing any errors.	The “siteDownlaod.sh” file runs without throwing any errors.	Passed	Tester verifies that the actual results are what was expected.
2	Verify that phishy.py is executed and select the appropriate option to start the campaign that you need.	2021-03-15	The “phishy.py” file runs without throwing any errors.	The “phishy.py” file runs without throwing any errors.	Passed	Tester verifies that the actual results are what was expected.
3	Fill out the appropriate variables to properly set up your phishing campaign.	2021-03-15	The phishing campaign is set up and sent to the selected users from the set setup.	The phishing campaign is set up and sent to the selected users from the set setup.	Passed	Tester verifies that the actual results are what was expected.
4	Verify that 01-test-apache.conf correctly parses the logs	2021-03-15	The logs are filled appropriately.	The logs are filled appropriately.	Passed	Tester verifies that the actual results are what was expected.
5	Wait for a predetermined amount of time for data to accumulate after running	2021-03-15	The predetermined time has passed.	The predetermined time has passed.	Passed	Tester verifies that the actual results are what was expected.

	the files in the previous steps.					
6	Verify that the Dashboards in Kibana are showing the correct information based off of Steps 1-3.	2021-03-15	The Dashboards are displaying the correct information based off of the files and logs that have been run and saved.	The Dashboards are displaying the correct information based off of the files and logs that have been run and saved.	Passed	Tester verifies that the actual results are what was expected.
7	End the Phishy Campaign by running the phishy.py file again and selecting the appropriate option.	2021-03-15	The results are displayed in phishing still and the campaign has ended.	The results are displayed in phishing still and the campaign has ended.	Passed	Tester verifies that the actual results are what was expected.

Table 3: The able above contains details on tests that were run on Phishy, as well as the outcome of each test

. Budget:

The Budget before the project was finished and the Budget after completion ended up being the same costs and totals so both budgets are represented in Figure 2 below.

Phishy: Tech Budget				
No.	ITEM	Unit, Hours	Unit Price	Total
Software				
1	Amazon AWS (EC2)	1	$689.41 (yearly)	$689.41
2	Amazon AWS (SES)	1	$12.45 (monthly)	$149.40
3	Kibana	1	$0.00	$0.00
4	Logstash	1	$0.00	$0.00
5	Elastisearch	1	$0.00	$0.00
	Subtotal			$838.81
Labor				
6	Logo Design	1	$200 (flat fee)	$200.00
7	Security Modifications	1	$400 (flat fee)	$400.00
8	Website & Social Media Build	1	$350 (flat fee)	$350.00
9	Database Build	1	$450 (flat fee)	$450.00
10	Development Hours	3	$28.50 (per hour)	$109,218.20
	Subtotal			$110,618
	Total			**$111,457.01**

*Figure 2: The table above outlines the budget for all software items and all of the labor items for the project**

** Note that item 10's total is the cost for three software developers working 40 hour weeks for 32 weeks.*

. **Project Timeline:**

Task #/Name	Duration	Start Date(MM/DD/YYYY)	End Date(MM/DD/YYYY)
1.0 Project Management and Deliverables	232 days	8/24/2020	4/13/2021
1.1 Team Building	1 day	8/24/2020	8/24/2020
1.2 Ideas and Brainstorming	1 day	8/24/2020	8/24/2020
1.3 Fall Semester Assignment 0: Team Members/ProjectName	1 day	8/24/2020	8/24/2020
1.3.1 Project Name	7 days	8/24/2020	8/31/2020
1.3.2 ProjectLogo/Branding	7 days	8/24/2020	8/31/2020
1.4 Fall Semester Assignment 1: Team Contract	8 days	8/24/2020	09 / 01 / 2020
1.4.1 Project Approval	1 day	09 / 01 / 2020	09 / 01 / 2020
1.4.2 Work Breakdown Structure	2 days	8/31/2020	09 / 01 / 2020
1.5 Fall Semester Assignment 2	42 days	09 / 01 / 2020	10 / 12 / 2020
1.6 Fall Semester Assignment 3	42 days	09 / 01 / 2020	10 / 12 / 2020
1.7 Fall Semester Assignment 4	7 days	10 / 12 / 2020	10 / 19 / 2020
1.8 Fall Semester Assignment 5	7 days	10 / 12 / 2020	10 / 19 / 2020
1.9 Fall Semester Assignment 6	21 days	10 / 19 / 2020	11 / 09 / 2020

1.10 Fall Semester Assignment 7	21 days	11 / 09 / 2020	11 / 30 / 2020
1.11 Fall Semester Oral Presentation	3 days	12/01/2020	12/03/2020
1.11.1 Presentation Practice	2 days	12/01/2020	12/02/2020
1.12 Spring Semester Assignment 1: Testing Plan/Report	2 weeks	01/04/2020	01/18/2021
1.13 Spring Semester Assignment 2: Abstract	2 weeks	01/10/2021	01/24/2021
1.14 Spring Semester Assignment 3: Draft Tech Expo Poster	2 weeks	03/01/2021	03/14/2021
1.15 Spring Semester Assignment 4: Final Poster	1 week	03/15/2021	03/22/2021
1.16 Spring Semester Oral Presentation (virtual)	3 days	04/01/2021	04/03/2021
1.16.1 Presentation Practice	2 day	04/01/2021	04/02/2021
1.17 Spring Semester Assignment 5: Final Report	1 week	04/04/2021	04/11/2021
1.18 Spring Semester Assignment 6: Safe Assign Final Report	1 week	04/04/2021	04/11/2021
1.19 IT Expo (virtual/In person?)	1 day	04/13/2020	04/13/2020
1.19.1 IT Expo Exhibit and Preparation	1 day	04/13/2020	04/13/2020

1.20 Spring Semester Assignment 7: Final Library Copy	3 day	04/14/2021	04/17/2021
2.0 Research	61 days	09/01/2020	10/31/2020
2.1 Equipment Requirements	3 weeks	09/01/2020	09/22/2020
2.2 Software Requirements	3 weeks	09/01/2020	09/22/2020
2.3 Safety Requirements	3 weeks	10/12/2020	11/02/2020
2.4 Miscellaneous Research	3 weeks	10/10/2020	10/31/2020
3.0 Design	2 months	11/01/2020	12/31/2020
3.1 Create Prototype	2 months	11/01/2020	12/31/2020
3.2 Create Software/App Legal Documentation	2 months	11/01/2020	12/31/2020
4.0 Environment Set-Up	2 months	11/01/2020	12/31/2020
4.1 Import Libraries for Development	3 days	11/01/2020	12/31/2020
4.2 Setup GitHub	2 days	11/01/2020	12/31/2020
5.0 Development (Back End and Front End)	5 months	11/01/2020	03/01/2020
5.1 Create Layout	1 week	11/01/2020	12/31/2020
5.2 Choose Color Scheme	3 days	11/01/2020	12/31/2020
6.0 Testing	2 months	02/01/2021	3/31/2021

6.1 Functionality Test	2 months	02/01/2021	3/31/2021
6.2 User Pilot Test	2 months	02/01/2021	3/31/2021
6.3 Defect correction	3 month	02/01/2021	04/11/2021

Table 4: The table above shows the projected timeline for our progress as we worked on Phishy

. Problems Encountered:

These are some of the Problems that we encountered while working on this project this semester:

- Sending emails through AWS, which we solved by doing research and ended up working through Amazon SES to allow emails to go through to our personal emails without being marked as spam.
- AWS was problematic at first seeing as none of us had really used it, we solved this issue by looking around and finding guides online and by getting advice and pointers from our Advisor to help get back on track.

. Future Recommendations:

While there are a few features that would make Phishy more user friendly, the most important of these is in the initial setup. In the continued development of Phishy,

we would be sure to automate the installation of the tools needed for Phishy to operate. One script, to be run on the ELK system, would automate the installation and basic configuration of Elasticsearch, Logstash, and Kibana, having an understanding of how these platforms work together is important for troubleshooting the system should any errors occur, however to effectively use Phishy all that is needed is knowledge on accessing/interfacing with Kibana through a web browser. A second script would be written to run on the system that Phishy actually runs on. This would handle the initial setup and configuration of Apache and Filebeat to ensure that logging is set up properly and that logs are being received by Elasticsearch.

While we developed Phishy to work with an ELK stack, we also see the value in it being able to work with other, similar systems. In continued development, we would create a second version of Phishy written to integrate with Splunk given the widespread use and growing popularity of that platform. With this we would also script the automated installation and basic configuration of each required component.

Conclusion

Throughout the Fall Semester, we learned a lot about working in AWS and the security parameters that they have in place to prevent people from using their service for malicious intent. We also learned how to use Python to create scripts and other backend work that allow the actions within our project to run simultaneously. In the Spring Semester, we were able to start fine tuning how our tool was able to be used. Now that the foundation of multiple technologies was put in place, we started to connect and configure them. There were many small tweaks that had to be made to allow the necessary information to pass between our web servers, hosted sites, logs, and ELK. Finally, we started to focus on making the tool easier for the user by creating simple command line menus to follow each step of the way. Many different templates and dashboards were also created, which gave users a starting point for the tool. Overall, this was a very enjoyable project, and the team learned new tools and technologies along the way.

References:

Crane, C. (2020, January 27). Phishing Statistics 2020: 15 Phishing Stats to Help You Avoid Getting Reeled In. Retrieved September 21, 2020, from https://sectigostore.com/blog/phishing-statistics-phishing-stats-to-help-avoid-getting-reeled-in/

Crane, C. (2020, April 30). Phishing Statistics: The 29 Latest Phishing Stats to Know in 2020. Retrieved October 05, 2020, from https://www.thesslstore.com/blog/phishing-statistics-latest-phishing-stats-to-know/

Egan, G. (2020, January 23). 2020 'State of the Phish': Security Awareness Training, Email Reporting More Critical as Targeted Attacks Spike: Proofpoint US. Retrieved October 01, 2020, from https://www.proofpoint.com/us/security-awareness/post/2020-state-phish-security-awareness-training-email-reporting-more-critical

2019 DBIR Summary of Findings. (n.d.). Retrieved November 10, 2020, from https://enterprise.verizon.com/resources/reports/dbir/2019/summary-of-findings/