Phishing for Users

Sean Stahly, Keathan Fertig, and Dr. Matt Miller
Dept. of Computer Science and Information Technology
University of Nebraska-Kearney
Kearney, NE 68849
stahlysr@lopers.unk.edu, fertigkc@lopers.unk.edu, millermj@unk.edu

Abstract

Phishing is the method of sending fraudulent emails to users that leverage the trustworthiness of computer systems. Phishing has become one of the most prevalent and effective methods of attacking Internet users. A user whom succumbs to clicking a link or opening a malicious document can lose information, reputation or pay a ransom. Our project had the goal to create a framework that would help test the effectiveness of phishing. We designed the framework such that it could be easily configured used by non-technical users, such that they could use it to study human behavior. The framework is designed such that it can be re-used for different experiments and extended to add additional capabilities.

1 Introduction

Phishing has been used on a regular basis in the last several years and continues to grow in popularity as a social engineering tool. According to the Anti Phishing Working group, phishing attacks have grown to more than 289,000 websites in Q1 of 2016 (a 250% increase from 2015) [1]. Phishing is used by attackers to obtain information or to gain unauthorized access to computer systems. Phishing emails are one of the top methods attackers use to steal information, compromise a computer, or hold ransom a computers files. Security experts have attempted to prevent phishing attempts by providing education about the detection and avoidance of phishing. Despite this education, it is still clear that some people are falling for phishing attempts. Recent high profile examples include the hack of the Democratic National Committee [2] and John Podesta email's [4]. As Phishing continues to be a problem, we decided to set about developing a set of tools that would allow for a researcher to generate, send, and track phishing emails in order to test the effectiveness of phishing awareness education.

2 Goal of Project

The goal of this project was to create a framework that would be able to test the effectiveness of phishing education. The framework simulates a real phishing attack by generating and sending phishing emails. To provide feedback to the researchers, the framework would track interactions of the email recipients. The recipients that believed that the email was legitimate and fell victim to unsafe behavior could receive additional training to help thwart future attacks. The project was also created to be streamlined and operated in such a way that a person with a limited technical background would be able to configure and run the project without further technical education.

3 Project Overview

To divide the work of development we chose to develop multiple different tools that each would implement a different portions of the system. The first tool would be in charge of generating a unique email for each recipient and it would send the emails to the intended recipients. The second tool would track the user's interaction with the email after it is received by the user. Both tools would record the interactions by the email recipients, for use by the researchers. These tools were developed using the python programming language, which is a user friendly programming language. A high level overview of the system is shown in Figure 1. Section 3.1 describes how a researcher can configure the program, create a template email and enumerate the phishing recipients. Section 3.2 describes how emails are configured and sent to the recipients. Section 3.3 describes how behavior is tracked and Section 4 describes future work.

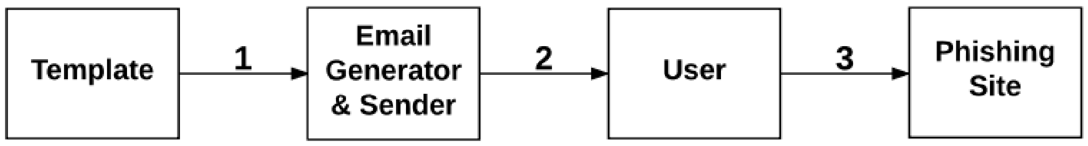

Figure 1: Model of Program Flow

3.1 Generating Emails

The email generator was responsible for creating the emails that would be sent as part of the phishing campaign. In order to make the process of generating emails simple and configurable, we chose to store configuration settings in plain text files. The main configuration file lists the program settings. In addition to the main configuration file, we allowed the researcher to provide an email template. This template would emulate a legitimate email in a way the researcher desired. This may include personal information, such as the recipients name, interests or known affiliations. Finally, we listed the phishing email recipients in a Comma Separated Values (CSV) file. This would include the personal information that would make the phishing email appear to be legitimate.

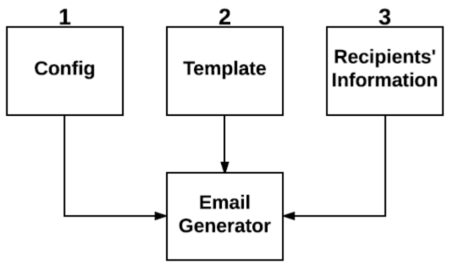

Figure 2: Model of Email Generation

3.1.1 Program Configuration

The program configuration file was designed to run the entire email generator and sender program. In order to maximize the organization and ease of use for the user of this program we decided to use a Yet Another Markup Language (YAML) file. A YAML file is easily separated into sections that make each configuration simple and clear.

1. location of the template
2. location of the CSV
3. website tracking information

These configurations will be discussed in Section 3.2.1. If the researcher was interested in running a single phishing campaign and did not want to use or modify a configuration file, they also have the option to pass all of the data that would be found in this file into the program via command line arguments.

3.1.2 Email Template

The email template is a file that is the basis of the phishing emails that can be launched through this program. The program allows the researcher to simply download an actual legitimate email. The researcher can then modify the file slightly and produce what is called a template email. Thy can the proceed to use the template in the phishing program, to send out emails.

Listing 1: Original Code

```
<title>Hello John Doe,</title>
```

Listing 2: Template Code

```
<title>Hello {{FirstName}} {{LastName}},</title>
```

An example of creating a template may include replacing a first name with an identifier that can be replaced later on in the process. Listing 1 shows the original email, which was sent to *John Doe*. Listing 2 shows that the researcher replaced *John* with {{*FirstName*}} and *Doe* with {{*LastName*}}. This markup allows the program to do substitutions when it generates the phishing emails.

3.1.3 Recipients' Information CSV

In Listing 3 shown an example CSV file. The CSV is configured to feature one recipient per row with a required email address for each recipient. In addition, the researcher can create additional columns containing more information on each recipient. The information found within these columns can then be inserted into the generated phishing emails.

Listing 3: Example CSV Code

```
LastName,FirstName,EmailAddress,FavColor
Stahly,Sean,stahlysr@lopers.unk.edu,red
Miller, Matt, millermj@unk.edu, blue and gold
. . .
```

Listing 4: Phishing Code

```
<title>Hello Sean Stahly,</title>
```

Listing 4 shows the generated spam email text, where {{*FirstName*}} has been replaced by *Sean* and {{*LastName*}} has been replaced by *Stahly* The generator was designed to accept a html formatted email, however it will accept documents of any extension was designated. To increase flexibility, the system is designed such that the researcher can define any set of substitutions (i.e. it will do more than just first and last name). The researcher can use any column that is specified in the CSV file. In Listing 3 we can see that the researcher added the *FavColor* substitution to the list of substituted values.

After the email generator has finished replacing any delimiters found within the template, it will set the email address to the address located in the CSV. If the CSV did not feature an **uuid** column, than a unique identifier (**uuid**) was created for each user. This unique identifier is use to identify and track which recipients interacted with the phishing email. The **uuid** was saved into the CSV to record the linkage between the user and the unique identifier.

3.2 Email Sender

The email sender was originally developed as a separate program. Figure 3 shows an overview of the process. The email sender will use the configuration and it will take a set of emails and send them to the desired recipients. To stream line the process, it was decided to merge the generation and sending programs.

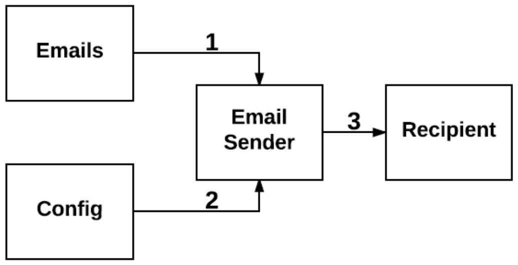

Figure 3: Model of Email Sending

3.2.1 Configuration File and Sending of Emails

The configuration file for the email sender is the same configuration yaml file that was mentioned in Section 3.1.1. The settings for sending the email are shown below. These settings allow the researcher to customize how the emails are sent and what the subject and spoofed email address will look like.

1. email server settings
   (a) username
   (b) password
   (c) SMTP hostname
   (d) SMTP port
2. sender's name
3. reply-to address
4. spoofed email subject

The sender's email address , sender's name, the reply-to email address, and the subject of the email are attached as headers to the emails before the email sender connects to a mail server to be sent. The email sender begins its final tasks by taking the generated emails with attached spoofed information and connecting to an smtp server. The email sender connects to the smtp server through a port and hostname provided in the configuration file and then begins sending emails.

3.3 Tracking Emails

When discussing how to track our phishing emails we decided that the most important part was the ability to track the links that were in the phishing emails. Figure 4 shows an overview of this process. The recipient will receive an email, click the link and visit a phishing site. The sit is hosted by our webapp and information about their visit is logged to a database. Phishing makes it quite easy for attackers to host malware on a website that can be visited from such links. The **UUID** generated in Section 3.1.3 is used to link email recipients with their behavior.

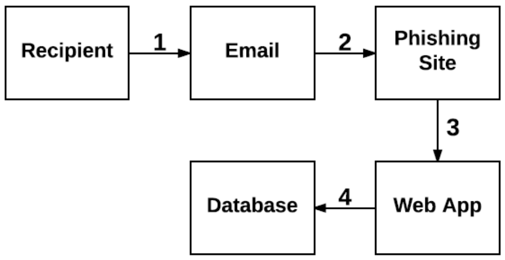

Figure 4: Model of Email Tracking

3.3.1 Receiving phishing Email

The recipient needs to receive the email in order to interact with it. After the recipient receives this email they are now presented with the option to click the link or ignore it. During our testing of the project, we did notice that while our phishing emails did go through, not all of them were usable. This was primarily due to the email client that the recipient used. We noticied that both Google's Inbox and Gmail client were able to detect our emails as phishing attempts and stipped the link from the email. Microsoft's Outlook, however, did no such action and did not even appear to identify our emails as a phishing attempt. This reinforces the fact that automated processes can in fact help detect and avoid phishing attempts.

3.3.2 Recipient Interacts with email

While hopefully all the recipient's received proper training to spot and ignore phishing attempts, we know there are still people who fall for these attempts. We developed project, to test the effectiveness of education that the recipients had received. If the recipient is not able to identify one of our emails as a phishing attempt, they may click on the link which will cause the webapp to track that behavior.

3.3.3 Web App Display's Phishing Site

Upon clicking the link found within the email, the user will now be taken to a webpage hosted by our webapp used to track the user. This webpage is a statically hosted html document that is designed by the researcher. The researcher can also choose to include a form on the webpage if they would like to conduct a more in depth the phishing attempt. Regardless of the webpage that the user decided to display, the webapp also records this interaction with the phishing email.

3.3.4 Web App Logging Recipient's Interaction

The webapp of this project was designed with purpose to track user's interaction with the phishing emails. We decided to create a python based webapp in order to keep it a consistent language with our email generator and sender. We chose the Twisted Framework [3], as it allows us to choose the port on which connections are made in the case we needed them for future improvements. When the user interacts with the link provided in the phishing email this web app will trigger an event logging the user. All of the links that were sent out featured a **UUID** parameter which what the web app uses in order to log who exactly clicked the link. The app will take this id and log it along with a time stamp and the url hit in both a log file and a mysql database. This webapp will remain active for the duration of the campaign, continuing to monitor the emails and log interactions with the phishing emails.

4 Future directions

While we were able to accomplish our goal of creating an easy to use framework that could generate, send, and track phishing emails, we can still improve upon this project. The first improvement we would like to make is to allow the webserver to host PDFs and other payloads that could be downloaded. In the same venue we would like to be able to allow our users the option to attach objects to the phishing emails, allowing them to test additional methods of phishing. We would also like to work on bypassing the spam and phishing alerts that certain email clients use to detect phishing emails. We know not every automated defense will catch every email so it would be useful to test the training that people have received about phishing and not rely upon automated systems. Finally, we would like to have the option to scan an email template for links, and then replicatie all the webpages linked to by that email on our webapp, making our phishing appear more legitimate.

5 Conclusion

This project provided an opportunity for the students involved to work in a team environment with technologies to which they were not previously exposed to such as python based web apps and programmatically sending emails. In addition, they were able to create a streamlined process that was able to focus on giving a non-technical researcher an easy opportunity to generate, send, and track phishing emails. As phishing continues to common attack from hackers, the ability to spot and properly respond to phishing emails becomes more crucial. This project sought to create, and was able to successfully implement a solution to test the education that users receive about phishing in a safe manner.

References

[1] BARTH, B. APWG report: Phishing surges by 250 percent in Q1 2016. `https://www.scmagazine.com/apwg-report-phishing-surges-by-250-percent-in-q1-2016/article/528186/`.

[2] LIPTON, E. How We Identified the D.N.C. Hack's 'Patient Zero'. `https://www.nytimes.com/2016/12/20/insider/how-we-identified-the-dnc-hacks-patient-zero.html?_r=0`.

[3] TWISTED MATRIX LABS. The Twisted Framework. `https://twistedmatrix.com/trac/`.

[4] WIKILEAKS. The Podesta Emails. `https://wikileaks.org/podesta-emails/`.