PHISH ALERT

Katie Doyle

C00240662

Christopher Staff

Institute of Technology, Carlow

25 April 2022

Abstract

This paper details in depth the processes that ultimately led to the Phish Alert end product. Phish Alert is a Phishing Detection Tool that extends the functionalities of the Gmail Application. Although there are many commercial Phishing Detection Tools available, the fact still remains, that Phishing is the cause for 90% of all data breaches (ReTruster, 2019). With this in mind, the decision was made to develop a tool that is conventionally programmed to detect and identify a more generic email, as opposed to training a Machine Learning system with a specific format of email.

The developmental cycle of Phish Alert was not unique, in that both general and technical issues arose continuously. The general issues stemmed back to the young age of Google Workspace add-ons, more specifically the restrictions imposed by Google on the development of the add-ons and the lack of documentation to support development.

The technical issues stemmed back to the initial research phase of Phish Alert, and the problems encountered as the project continued to develop. Some of these issues include a complete overhaul of the chosen technologies at the beginning of the research phase, issues relating to displaying a notification in the Gmail application, trying to implement Firebase Cloud Messaging, creating the actual add-on with the use of cards and widgets and identifying links in an email, just to name a few.

The project excelled in all areas but one, that is the unfulfillment of the evaluation approach selected during the research phase. The proposed evaluation approach was to obtain a sizeable dataset of Phishing and Non Phishing emails and deduce the system accuracy based on the correct identification of these emails. This is a typical approach for a Machine Learning system, but it became apparent late in the development phase that this approach would not work for my proposed system, and time constraints limited a workaround.

Table of Contents

Introduction

It's safe to assume that nearly everyone has received a Phishing email before, whether or not they have fallen for them. Despite the number of pre-existing tools available on the market to combat Phishing, it continues to grow and evolve as the leading cyber threat. In 2020, the FBI Internet Crime Complaint Centre received 241,342 Phishing complaints, which is more than double the number in 2019. This number is merely just the few that get reported, but it does give a good indication of the general growth trend, that the number of reported incidents doubled year-on-year (Mimecast, 2021).

The impact this threat has on businesses is severe, with a report by Ponemon Institute suggesting that the average cost of a data breach to a business sits at about $3.86 million US dollars.

One of the well-known preventives is to install security technologies such as Phishing filters on email applications to prevent successful attacks, yet the number of attacks is rising (Mimecast, 2021). On foot of this, Phish Alert was developed with a generic email in mind, to identify more potentially harmful emails, and not limit the abilities of the system by training it with a certain template of email. Phish Alert is integrated within the Gmail application and runs on Google's servers, meaning the user does not have to run anything locally to use the system.

As with any project development cycle, it is guaranteed that issues will arise. This paper documents the successes and troubles experienced during the development cycle of Phish Alert.

General Issues

Throughout the life cycle of Phish Alert many issues of magnitude arose which were not foreshadowed. Many of these are predominately technical issues, which are documented below, but two can be categorized as general issues:

Google Apps Script Restrictions

Google imposes the following restrictions on add-on development, and what the add-ons can be designed to do.

- Change Features in Google Workspace

  The add-on framework is designed to enhance Google Workspace applications – not to add limits. This means that existing features cannot be altered or locked down during the Google Workspace document sharing model.

- Charge Users to Install

  Google doesn't provide a way to charge users to install add-ons, and they cannot include ads. However, companies can roll their own payment system or call into an existing billing database.

- Detect Many Events

  Except for certain triggers, add-ons can't tell what a user does outside the add-on itself. For example, it cannot be detected when a user clicks on the host application toolbar.

- Extend all Google Workspace apps

  Google Workspace add-ons can only extend Gmail, Calendar, Drive, Docs, Sheets and Slides. Google does plan to roll these add-ons out to other Google applications in the future.

- Use HTML/CSS or Client-Side Scripting

  Google Workspace add-ons must use card-based interfaces. The HTML/CSS interfaces supported by editor add-ons can't be used. Google Workspace add-ons use a widget-based approach to building user interfaces. This lets the add-on work well on Desktop and Mobile platforms.

- Full Mobile Support

  For the most part, Google Workspace add-ons function on desktop web clients. Non-contextual homepages are not yet available from the Gmail, Calendar, or

Drive mobile apps. Google Workspace add-ons are not available from mobile web browsers, only the Gmail application.

- Use SVG Images

  Currently, SVG images cannot be used with card service cards and widgets.

- Have More Than 100 Widgets

  For performance reasons, you cannot add more than 1000 widgets to a card section, or more than 100 card sections in a card (Google Apps Script, 2022).

Lack of Documentation from Google

Google does provide documentation on how to write an add-on, however, when an issue arose, I had no place to turn, because the development platform is so new that most of the issues encountered were not documented. This increased time pressure as some issues, for example the issue I have documented above about card service brought the development stage to a halt until it was resolved. That issue was my lengthiest, spanning over a period of nearly 14 days. Again, due to the lack of detailed documentation, I had to resolve this particular issue through trial and error, as opposed to being able to reference a similar issue online, and resolve it in a timely manner.

I will begin by documenting the successions of Phish Alert before detailing the difficulties that arose and ultimately contributed to the learning process and end result of the project.

Accomplishments

I have achieved the required specifications for Phish Alert; that is to extend the functionality of Gmail, a free email service provided by Google, by developing an integrated Phishing Detection Tool. Below is a table which details the functionality of the Gmail Add-On, under the headings of User Experience (UX) and User Interface (UI).

User Experience (UX)

Functionality	Description
1. Compare the ‘Mail From’ and ‘Return-Path’ addresses	Comparison of two email addresses, the ‘Mail From’ address, which is visible to the mail recipient, and the ‘Return-Path’ address, which is contained in the email header information, to check for inconsistencies.
2. Check if Domain-based Message Authentication, Reporting and Conformance (DMARC), DomainKeys Identified Message (DKIM), and	A check is undertaken on the emails header information, to identify if DMARC, DKIM, and SPF, a set of email authentication protocols, are all passing.

Sender Policy Framework (SPF), are all passing	
3. Identify if the message body of an email contains a Uniform Resource Locator (URL)	A check is done on the emails message content to identify and store any URLs present. The URL could feature an IP Address or a domain name.
4. Perform a WHOIS lookup to obtain a domain registration date	An external WHOIS lookup is performed to retrieve the registration date associated with the domain name contained in the URL.
5. Perform an analysis of the domain registration date	Analyse the domain registration date to identify if the domain age is less than or greater than thirty days old.
6. Perform a Google Custom Search	Perform a Google Custom Search, a programmable Google Search Engine, to return the number of web search totals for the URL / domain name.
7. Perform a Google Blacklist Search	Perform a Google Blacklist Search, by referencing Googles Safe Browsing Blacklist, to identify if the URL is listed.
8. Analyse the Email Header information to identify the 'X-Google-Original-From' entry	Analyse the email header information to identify if the 'X-Google-Original-From' entry is present, as this indicates the 'Mail From' address the sender attempted to use is an alias.
9. Analyse the URL to determine if a full URL was received including the protocol type	Analyse the URL to determine the protocol type (if a full URL is contained in the email body) of the URL. This could be HTTP or HTTPS.
10. Analyse the URL to identify and isolate the Top Level Domain (TLD)	Analyse the URL to determine the TLD and compare it against a Blacklist of the top 10 most dangerous TLDs.
11. Identify if the email contains an attachment	Identify and document if the email contains an attachment. This result will determine if an external scan needs to take place and if implicit consent is required from the user before this is done.
12. Perform a scan of the attachment to determine if it is malicious or benign	Perform an external scan of the attachment using VirusTotal to detect
13. Provide a classification based on the results of the various functions	Provide a classification of Phishing or Non-Phishing based on the results of the functions. As each function is iterated the result is documented and each result then combines to deduce a classification.
14. Provide recommendation(s) based on the email classification and the results of each function	Provide unique recommendation(s) based on the results of the email classification and metadata.
15. Provide a lasting record of each email and its	Provide a lasting record of each email and its classification and detailed metadata to the user by

classification and metadata to the user	inserting this information into a Google Sheets spreadsheet simultaneously as the analysis is taking place. The user is given access to the spreadsheet through the User Interface and the spreadsheet also contains a chart for the user to visually see the breakdown of their inbox in terms of Phishing and Non-Phishing.

User Interface (UI)

Functionality	Description
Display to the user, in the form of an integrated User Interface in the Gmail Application, a consent message if it is found that an attachment is present	An interactive, unique and customized User Interface is presented to the user to extend the functionalities of the Gmail Application. The user can act on the message in different ways, by giving active consent to allow an external attachment scan to take place, which has the potential to change the outcome of the findings.
Display to the user, in the form of an integrated User Interface in the Gmail Application, a summary of the metadata findings if it is found that a link is contained in the message body	An interactive, unique and customized User Interface is presented to the user to extend the functionalities of the Gmail Application. The user can choose to display a summary of the metadata findings within the Add-On, and / or have the option to view the full details of the findings in the Google Sheets spreadsheet via a link.
Display to the user, in the form of an integrated User Interface in the Gmail Application, a consent message and a summary of the metadata findings if it is found that an attachment and a link is present	An interactive, unique and customized User Interface is presented to the user to extend the functionalities of the Gmail Application. The user can act on the message in different ways, by giving active consent to allow an external attachment scan to take place, which has the potential to change the outcome of the findings. In this instance, the user can also choose to display a summary of the metadata findings within the Add-On, and / or have the option to view the full details of the findings in the Google Sheets spreadsheet via a link.
Display to the user, in the form of an integrated User Interface in the Gmail Application, a summary of the metadata findings if it is found that no attachment or link is present	An interactive, unique and customized User Interface is presented to the user to extend the functionalities of the Gmail Application. The user can choose to display a summary of the metadata findings within the Add-On, and / or have the option to view the full details of the findings in the Google Sheets spreadsheet via a link.
Display to the user, in the form of an integrated User Interface in the Gmail Application, a message stating that there is no unread	A unique and customized User Interface is presented to the user to extend the functionalities of the Gmail Application. The user is informed that there is currently no unread emails to be analysed after

emails to be analysed if the Add-On is initiated for a previously read and / or analysed email	initiating the Add-On to analyse a previously read and / or analysed email.

User Experience Functionality Description

1. The reason a check is done on the FROM address with the RETURN-PATH address is to identify any inconsistencies between these two. It can be common for a bad actor to spoof or falsify the visible FROM address on an email and leave the real RETURN-PATH address inside the header. For this reason, any inconsistencies between these two addresses can be indicative of spoofing (btel, 2022).

2. DMARC, or Domain-based Message Authentication Reporting and Conformance, is an authentication method, that enlists the help of two protocols, DKIM and SPF, to help domain owners fight Business Email Compromise (BEC), Phishing and Spoofing (dmarcian, 2022). If incoming mail is passing DMARC alignment is provides an extra layer of reassurance. If DMARC alignment does not pass, depending on the domain owners policy settings, the recipient may never even be aware of the incoming mail.

3. A regular expression helps to identify any URLs contained in the content body of an email. The regex can identify URLs that contain either a domain name or IP Address.

4. A WHOIS lookup is performed on the URL located in the body of the email, to identify a domain registration date. Newly registered domains (NRDs) are often used for malicious intent (Palo Alto Networks, 2022).

5. A check is carried out on the domain registration date, to calculate if it is less than 32 days old. A domain is considered newly registered if it has been registered or had a change in ownership in the last 32 days (Palo Alto Networks, 2022). Again, this could indicate malicious intent.

6. A Google Custom Search is performed to retrieve the total number of Google search results for the URL. A lower number here could potentially indicate an illegitimate website as popular sites such as Amazon have 3,940,000,000 search results, and eBay with 2,560,000,000 search results.

7. A Google Blacklist Search is performed to identify and retrieve information about a URL being present in the database of unsafe sites maintained by Google. Websites may feature in this database for a number of different reasons, some being; Malware, Phishing attacks, and Spyware (Astra, 2022).

8. A search is performed on an emails headers to identify if the entry “X-Google-Original-From” is present. If present, this indicates that the sender attempted to use an alias address, different from the actual sending address. Instead of fulfilling this, Google displays the actual from address to the recipient, and places the proposed alias address in the header.

9. A check is carried out to identify the protocol included in the full URL. HTTPS indicates a secure site, so if HTTP is identified, it shows that the web client will be accessing the resource using an unencrypted / insecure channel.

10. A further check is carried out on the URL to identify the Top Level Domain present. A blacklist array currently contains what is believed to be the top 10 most dangerous TLDs as of April 2022. Although this list does not contain the biggest TLDs (such as the .com TLD), it does contain a list of the top 10 TLDs which have been identified as being abusive, which are also not labelled as the most popular / largest TLDs, which clearly signals criminal preference (Palo Alto Networks, 2021). The hardcoded blacklist array contains the follow TLDs; .cn, .surf, .ga, .gq, .cf, .tk, .ml, .work, .top, .cam (Spamhaus, 2022).

11. As the email is analysed, the system will identify if an attachment is present. If an attachment is present, the Card Service UI will then display a warning message for the user, detailing the risks of sharing potentially sensitive attachments with a third party service, followed by requesting active consent from the user for this scan to commence.

12. Once consent is granted from the previous function, the scan takes place, and if the attachment if found to be malicious, the number of security vendors who have positively identified the attachment as malicious is returned to the Card Service UI. Also returned from the VirusTotal API is a unique link which will bring the user to the landing page associated with their attachment to view the security vendor results as a UI. Alongside this, the VirusTotal results are also added to the spreadsheet.

13. A results array, initialized at the beginning of analysis, contains the results of each of the functions mentioned above. Depending on the results of each function, for example, if the Google Blacklist search returns a result of Malware for that specified URL, the results array is given a value of 1 – indicative of Phishing for the index position which represents this function. If the results of this function were returned as ‘Not Detected’, the results array would be given a result of 0 – which is indicative of no Phishing for this particular function. This results array proves vital for the classification of the email at the end of analysis. There are two options of classifications; Phishing and Non-Phishing.

14. Aside from providing a classification, Phish Alert also provides the user with a recommendation (feedback) based on the results of each individual email, with the hopes of creating a teachable moment with the user. Again, this feedback is based upon the individual results of each function.

15. To really create a teachable moment with the user, the use of adding all the analysis metadata to the spreadsheet provides a lasting solution for the user to visit and revisit the entries to understand what the program picked up on, how the program determined the classification, and show the user what they missed. The aim of this is to have the user more Phishing aware, and hopefully allow them to identify Phishing emails themselves.

Security

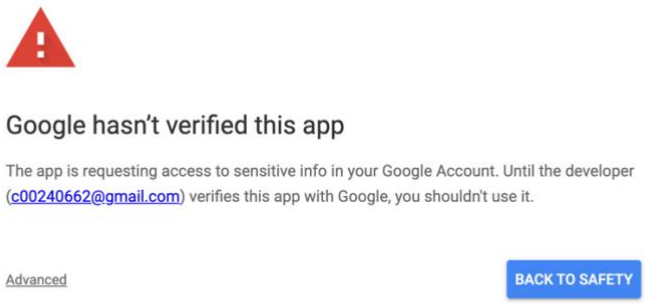
Security is an important aspect in any development, but in this case, Phish Alert has full access to a user's mailbox, which is a massive security and privacy risk. In 2018, Google announced a change in the Gmail Add-On sphere, to further elevate user trust within this ecosystem. This announcement introduced an application review for all applications accessing the Gmail mailbox. This review takes place before the proposed application can be added to the Google Workplace Marketplace to allow for user downloads. Before an application is even submitted for review, Google emphasises the importance of secure data handling, requiring developers to demonstrate secure data handling with the following assessments before review:

- Penetration Testing
- External Network Penetration Testing
- Account Deletion Verification
- Reviews of Incident Response Plans
- Vulnerability Disclosure Programs
- Information Security Policies

The security assessment reviews the proposed application for compliance with policies governing appropriate access, limited use, and minimum scope. Following this, a third party assessor will begin the security assessment. The assessment incurs a fee of between $15,000 to $75,000 US dollars depending on size and complexity. A fee is paid as the security assessments takes place by a 3rd party to ensure the confidentiality of the application (Google Cloud, 2018).

This security assessment would ensure the confidentially, integrity and availability of the system.

For the purposes of this project, Phish Alert is currently not available for public download on the Google Workspace Marketplace, and is instead installed as a developer add-on, on a test account.



The screenshot above shows the warning banner that is displayed to the user (test account) when installing the developer add-on for the first time.

As scripts created with Google Apps Script run on Google's servers, Phish Alert benefits from the security designed into Google's technical infrastructure. Google offers protections such as:

- DoS Protection – this is achieved by the scale of Google infrastructure, which allows it to absorb many DoS attacks. On top of this, there is multi-tier, multi0layer DoS protections.

- User Authentication – the next layer of defence comes from the central identify service. This service requires a username, password and it also challenges users for additional information based on certain risk factors.

- Safe Software Development – Google uses libraries that prevent developers from introducing security bugs such as Cross Site Scripting. Google also use automated tools such as fuzzers, static analysis tools, and web security scanners to detect security bugs. In addition, Google offers a vulnerability rewards program for members of the public to discover and inform them of bugs in their infrastructure and applications.

- Source Code Protections – Google's source code is stored in repositories which feature built-in source integrity and governance, where current and past versions of services can be audited.

- Keeping Employee Devices and Credentials Safe – Safeguards are implemented to protect employee devices and credentials from compromise. To prevent against phishing attempts, Google have replaced One-Time Password (OTP) second-factor authentication with the mandatory use of Universal 2nd Factor Authentication (UTF) compatible security keys.

- Reducing Insider Risk – Employees granted administrative access to Google's infrastructure have their activity limited and are closely monitored. The need for privileged access for certain tasks is eliminated with the use of automation to complete the same tasks in a controlled way.

- Threat Monitoring – There is a threat analysis group at Google that monitors threat actors and the evolution of associated tactics and techniques. The goal of this group is to improve the safety and security of Google's products.

- Intrusion Detection – Sophisticated data processing pipelines are used to integrated host-based signals on individual devices, network-based signals from various monitoring points in the infrastructure, and signals from infrastructure services. Rules and machine intelligence built on top of these pipelines gives security engineers warnings of possible incidents. The investigation and incidence response teams triage, investigate and respond to these potential incidents 24x7x365.

- Deletion of Data – Deletion of data starts with marking specific data as scheduled for deletion rather than actually deleting the data. This allows Google to recover from accidental deletion – whether customer or bug initiated.

- Encryption at Rest – Google provides several layers of encryption to project data at rest. By default, the storage infrastructure encrypts all user data before the user data is written to physical storage.

- Service identity, integrity, and isolation – To enable inter-service communication, applications use cryptographic authentication and authorization. Authentication and authorization provide strong access control at an abstraction level and granularity that administrators and services can understand.

- Secure Service Deployment – Google services are the application binaries that their developers write and run on Google's infrastructure. The infrastructure does not assume any trust between the services that are running on the infrastructure. This trust model is referred to as a zero- trust security model. A zero-trust security model means that no devices or users are trusted by default, whether inside or outside the network.

- Secure Boot Stack and Machine Identity – Google servers use various technologies to ensure that they boot the correct software stack. Cryptographic signatures are used for low-level components like the baseboard management controller (BMC), BIOS, bootloader, kernel, and base operating system image. These signatures can be validated during each boot or update cycle. The first integrity check uses a hardware root of trust. Each server in the data centre has its own unique identity. This identify can be tied to the hardware root of trust and the software with which the machine boots.

- Security of Physical Premises – Each data centre is built to incorporate multiple layers of physical security. Access to these centres is tightly controlled. Multiple physical security layers such as biometric identification, metal detection, cameras, barriers and laser based intrusion detection systems (Google Cloud, 2022).

Non-Fulfilment

In the wake of the major successions of Phish Alert, the goal of evaluating the system in a typical approach became unrealistic. This led to the unfulfillment of the evaluation approach selected at the beginning of the research phase.

Evaluation Approach

The initial evaluation approach was selected after researching systems of similarities, with the main technology used for these developments being Machine Learning techniques. In a typical Machine Learning approach to developing a Phishing Detection Tool, the system is evaluated with the use of a substantial dataset, a collection of Phishing and Non-Phishing

mails to gauge the success of the system. After an overhaul of the original technologies selected to develop Phish Alert the Technical Issues section, it became apparent that this atypical approach to developing a Phishing Detection Tool could not be evaluated using the same approach for a typical system.

It became apparent that there was no way to detach the developed Add-On and its functionalities as standalone software to test a sizeable dataset, nor was it feasible to set up the individual test mails as incoming mail for analysis, as they would have to be altered to send from a predetermined SMTP configuration. A solution to this problem, to set up a SMTP server to be able to send the test mails in their original state, would enable evaluation of the system and its accuracy. Further discussion of configuring an SMTP server is discussed in the Reflection section below.

Ultimately I prioritized a usable solution over an evaluable solution.

Experience

Throughout the life cycle of the Phish Alert, I have had many opportunities to learn new technologies I would not have had previous exposure to. These technologies include:

- Google Apps Script Development
- JavaScript Development
- Google Safe Browsing API
- Google Custom Search API
- GoPhish
- VirusTotal API
- DNSTwister API
- Gmail API
- Sheets API

Aside from the exposure to new technologies, I've also learned some valuable lessons with respect to project management. Some of these lessons include knowing:

- To break down the main functionalities of the project scope into smaller, manageable segments
- That the original blueprint will be revisited and recalibrated
- That a usable end product is better than an incomplete, evaluated system
- That project scope may sometimes have to be altered for a better outcome
- To expect design changes as issues arise
- To expect schedule changes as issues arise
- To expect implementing workarounds as opposed to resolving every issue
- That time management is critical as it directly affects the quality and scope of the end product
- That research and planning is crucial in identifying the goals, focus and methods involved

- That it is critical to identify and reduce risks at the beginning of a project – where possible
- That perseverance goes a long way towards making the end product successful
- How to problem-solve more effectively – especially with time constraints

Reflection

With the knowledge I now possess with respect to Phish Alert and its processes, there are several things I would do differently to alter the outcome of the project.

The main difference a revised model would feature would be in respect to the evaluation approach. I would locate a sizeable collection of Phishing and Non-Phishing emails and configure an SMTP server to send the emails in an unaltered state, to be received as incoming mail within the Gmail Application.

After researching several methods of configuring an SMTP server, I would opt to switch development from Mac OS to Microsoft Windows to allow access to hMailServer, which is a free, open source email server for Microsoft Windows (hMailServer, 2022). The reason I would select this particular software would be due to the fact it is free, open source and a switch to the Windows operating system would not have any repercussions for Google Apps Script as it runs right in the browser. With an SMTP server configured, a chosen collection of emails, including the original header information, could be relayed to the inbox of a testing account for Phish Alert to analyse. As the chosen collection would contain both Phishing and Non-Phishing, the level of system accuracy could be deduced accurately using this method.

In terms of the algorithmic workings Phish Alert currently possesses, I would not choose to alter these, but rather add more features to expand the abilities of the system. Further research online suggests several methods of examining URLs for precise Phishing Detection, such as looking at the number of dots contained in a URL, the length of the URL, and the number of forward slashes contained in a URL, to name a few (Hindawi, 2019).

As discussed in the Technical Issues section below, I would bypass the use of the Java Programming language and its related Java Mail API to receive email metadata outside of the Gmail Application, and take the timing used on this aspect and apply it to the configuration of an SMTP server for system evaluation.

In an ideal situation, the schedule of the project would be longer to ensure a better outcome.

Technical Issues

Throughout the life cycle of Phish Alert, the system design began to change as issues were encountered. Each significant problem encountered is documented below.

The use of the Java programming language and the Java Mail API.

As per the Google Workspace for Developers website, a Gmail Add-On can be built in any language (Google, 2022). With this in mind, the programming language selected during the research phase was Java, due to my proficiency and the extensive libraries the language offers. As the development process began, it became apparent that the use of the Java programming language could replace the Google Apps Script development platform by allowing for the development of the scripts in a Java Application, but still required these scripts to be JavaScript based. This discovery was made after a period of time had been dedicated to developing the Java Application-environment. The use of the Java Application then proved no benefit, so the decision was made to revert back to the Google Apps Script development platform, which is a browser based editor, running on Google's servers.

Notifying the user of a mail classification within the Gmail Application

The initial blueprint for Phish Alert concluded that the mail recipient would be notified of an email classification (Phishing, Non-Phishing) from within the Gmail Application, in the form of a notification. Extensive research revealed that Google restricts the use of JavaScript properties such as alert boxes, confirm boxes and prompt boxes in the Gmail Application. It became apparent that it would be impossible to allow the mail recipient to receive a notification naturally from within Gmail, so further research was conducted to find a suitable alternative. This research returned a plausible solution, to implement Firebase Cloud Messaging, a server based solution to send messages and devices across multiple devices such as iOS, Android and the Web (Firebase, 2022). The result of this methodology is discussed below in further detail.

Firebase Messaging

As discussed above, a solution to allow for a notification to be displayed to inform the user of an emails classification, is Firebase Cloud Messaging. Although this solution would of restricted the use of the Phish Alert Add-On across multiple devices due to the need for a device to be uniquely identified through a generated device token, it would of allowed for the desired outcome. During a lengthy implementation of Firebase, the process of integrating it into Google Apps Script became increasingly difficult due to the lack of documentation available on such a task. In the best interests of the outstanding tasks and reaching task deadlines, the Firebase Cloud Messaging integration had to be abandoned, and further research began to find a plausible solution. This time, I made the decision to enlist the help of Google Sheets – a web based spreadsheet program. The use of Sheets comes into effect once Phish Alert begins to analyse the metadata surrounding an email, adding the findings to a lasting spreadsheet accessible to the user. This solution also provides the added benefit of a lasting solution for the user to go back and reference, not only to protect the user from malice in the short term but to also provide a teachable moment, for future reference.

Card Service User Interface

Card Service is a service that allows scripts created within Google Apps Script to configure and build card and widget components and behaviours for a User Interface (UI) within the Gmail Application (Google Apps Script, 2022). For the purposes of Phish Alert, I decided to create multiple cards to display multiple User Interfaces conditionally, based on the analysed email metadata, for example, if an email contained a URL, an attachment, contained both, or contained neither. The process of ensuring each Interface only displayed when certain conditions were met was lengthy and challenging. This process involved having to move the code to a sandbox and dissect each individual function to fix the logic.

Phish Tank

At the beginning of the research phase, Phish Tank, an online repository containing the data and information of phishing URLs (Phish Tank, 2022), was chosen to be used to test the accuracy of the system. Alongside this data being accessible in the browser, a developer API is offered for free, requiring just an account registration. Upon attempting registration, I discovered that the site had suspended new user registrations, putting an end to the use of Phish Tank.

Evaluation Approach

As mentioned above, the original evaluation approach selected through research was similar to Machine Learning technique evaluations, that is to determine the system accuracy with the use of a sizable dataset containing both Phishing and Non-Phishing emails. As I chose to develop my Phishing Detection Tool in an atypical fashion, it became apparent that using a typical evaluation approach was not going to work with Phish Alert. The main reason a dataset evaluation would not be successful is the lack of ability to detach the Add-On from the Gmail Application as a standalone program to read in a dataset. This would mean that the contents of the dataset (Phishing and Non-Phishing emails including headers) would have to be received as incoming mail. A standard, open source Phishing framework like GoPhish, requires a valid SMTP configuration to send mail from, for example, a Gmail account. To send these mails in an unaltered state, an SMTP server would have to be configured, an additional task that had not been allocated time.

Google Sheets Spreadsheet

When the decision was made to create a lasting repository for analysed email metadata, issues arose when trying to get the initial data into the spreadsheet. Further research revealed that the script had to be applied to the spreadsheet. The original code then had to be copied and added to a new project which was then applied to the spreadsheet. This resulted in the data carrying over to the spreadsheet as it should.

Identification of Links in an Email

The initial Regular Expression used to identify URLs in the body of the email worked in identifying URLs which contained a domain name, for example www.google.com. When I ran a campaign with the body of an email containing a URL with an IP Address as opposed to a domain name, for example www.192.168.1.10, the Regular Expression was unable to identify that this was in fact a URL. I had to implement a new Regular Expression which catches all kinds of URLs, whether they feature a domain name or IP Address.

Google Custom Search API

The Google Custom Search JSON API imposes a restriction of 100 search queries per day for free. This proved challenging during the development phase, as it was particularly easy to max out this quota.

Record Layout Diagram

Below is a record layout diagram which depicts the layout and / or the display of the data in the Google Sheets spreadsheet.

Date	Sender	Subject	Content	Link
Attachment	Total Security Vendors	Security Vendor Positives	Address Result	Domain Registration Date
Search Results Total	Threat Type	Registration Classification	Address Result	DMARC, DKIM, SPF
TLD	Protocol	Protocol Result	TLD Result	Classification
Recommendation				

*Line breaks shown are for the purposes of this document only.

Whereby:

- Date – contains the date and time the email was sent
- Sender – contains the sender address (that is visible to the user)
- Subject – contains the subject of the message
- Content – contains the entirety of the message body
- Link – contains the link contained in the body of the email
- Attachment – identifies whether the email contains an attachment or not
- Total Security Vendors – relative to the VirusTotal API results
- Security Vendor Positives - relative to the VirusTotal API results
- Address Result – identifies whether the FROM and RETURN-PATH addresses match
- Domain Registration Date – contains the domain registration date
- Search Results Total – contains the total number of Google search results for the link
- Threat Type – contains the threat type returned by the Google Blacklist search

- Registration Classification – identifies if the domain registration date is recent or not
- Address Result – identifies the presence of a spoofed address
- DMARC, DKIM, SPF – identifies is these authentication methods are passing
- TLD – contains the TLD, taken from the link
- Protocol – contains the protocol, taken from the link
- Protocol Result – identifies whether the link is establishing an insecure (HTTP) or secure (HTTPS) connection
- TLD Result – identifies whether the TLD is blacklisted or not
- Classification – Identifies whether the email is Phishing or Non-Phishing
- Recommendation – establishes the feedback for the user, depending on all the results above

System Testing

Although an evaluation approach to determine the accuracy of the system could not be achieved, the reliability of Phish Alert has been tested using several test cases as documented below.

Test Case Type	Description	Test Step	Expected Result	Status
Usability	Program should detect a URL in the body of an email	Set up a campaign to send an email with a URL	Card Service UI for URLs should appear	PASS
Usability	Program should detect that an email contains an attachment	Set up a campaign to send an email with an attachment	Card Service UI for attachments should appear	PASS
Usability	Program should detect that an email has an attachment and a URL	Set up a campaign to send an email with an attachment and a URL	Card Service UI for attachments and URL should appear	PASS
Usability	Program should detect that an email does not have an attachment or URL	Set up a campaign to send an email with no attachment or URL	Card Service UI for no attachment or URL should appear	PASS
Usability	Program should detect that there is no unread emails	Mark any unread emails as read	Card Service UI for no unread emails should appear	PASS
Functionality	Program should detect a URL in	Set up a campaign to	Correct URL should be inserted	PASS

	the body of an email	send an email with a URL	into the spreadsheet	
Functionality	Program should detect whether the mail FROM and RETURN-PATH addresses match	Set up a campaign to send an email with matching the mail FROM and RETURN-PATH addresses	Correct identification (MATCH) should be inserted into the spreadsheet	PASS
Functionality	Program should detect whether DMARC, DKIM & SPF are passing	Set up a campaign to send an email with DMARC, DKIM & SPF passing	Correct identification (PASS) should be inserted into the spreadsheet	PASS
Functionality	Program should identify the domain registration date for a URL contained in the message body	Set up a campaign to send an email with a URL	Correct registration date should be inserted into the spreadsheet	PASS
Functionality	Program should determine if the domain registration date is older than 30 days	Set up a campaign to send an email with a URL	Correct determination (<30 days or >30 days) should be inserted into the spreadsheet	PASS
Functionality	Program should determine the total number of Google search results for a URL	Set up a campaign to send an email with a URL	Correct number of search results should be inserted into the spreadsheet	PASS
Functionality	Program should determine if the URL is present in the Google Blacklist database	Set up a campaign to send an email with a URL with a threat type of Malware	Correct threat type of Malware should be inserted into the spreadsheet	PASS
Functionality	Program should determine if the 'X-Google-Original-From' entry is present in	Set up a campaign to send an email with a spoofed FROM address	Correct determination (SPOOFED) should be inserted into the spreadsheet	PASS

	the header information			
Functionality	Program should determine the protocol type if a full URL is present	Set up a campaign to send an email with a full URL including protocol	Correct protocol type should be inserted into the spreadsheet	PASS
Functionality	Program should determine the TLD of the URL	Set up a campaign to send an email with a URL	Correct TLD should be inserted into the spreadsheet	PASS
Functionality	Program should determine if the email contains an attachment	Set up a campaign to send an email with an attachment	Correct determination (YES) should be inserted into the spreadsheet	PASS
Functionality	Program should conduct an external scan of the attachment	Set up a campaign to send an email with an attachment that is known to be malicious	Correct number of security vendors and security vendor positives should be inserted into the spreadsheet	PASS
Functionality	Program should deliver a classification based on the previous functions results	Set up a campaign to send an email with a URL (Can be any test step)	Classification should be inserted into the spreadsheet	PASS
Functionality	Program should deliver recommendations (feedback) based on the previous functions results and classification	Set up a campaign to send an email with a URL (Can be any test step)	Recommendations should be inserted into the spreadsheet	PASS
Functionality	Program should provide the user with a timeless repository of metadata analysis results	Set up a campaign to send an email with a URL (Can be any test step)	Metadata analysis results should be inserted into the spreadsheet and should not expire	PASS

Glossary

Glossary of Terms, Abbreviations and Acronyms

API:	Application Programming Interface
BEC:	Business Email Compromise
BIOS:	Basic Input/Output System
BMC:	Baseboard Management Controller
CSS:	Cascading Style Sheets
DKIM:	DomainKeys Identified Mail
DMARC:	Domain-based Message Authentication, Reporting and Conformance
DoS:	Denial of Service
Email:	Electronic Mail
FBI:	Federal Bureau of Investigation
HTML:	Hypertext Markup Language
HTTP:	Hypertext Transfer Protocol
HTTPS:	Hypertext Transfer Protocol Secure
IP:	Internet Protocol
NRD:	Newly Registered Domain
OS:	Operating System
OTP:	One-time Password
SMTP:	Simple Mail Transfer Protocol
SPF:	Sender Policy Framework
SVG:	Scalable Vector Graphics
TLD:	Top Level Domain
UI:	User Interface
U2F:	Universal 2nd Factor
US:	United States
UX:	User Experience
URL:	Uniform Resource Locator

Bibliography

Astra, 2022. *Google Blacklist – How to Remove Your Website from Google Blacklist (WordPress, Magento, PrestaShop, OpenCart, Drupal & PHP)*. Available at: https://www.getastra.com/blog/911/google-blacklist/#:~:text=this%20post%20helpful%3F-,What%20is%20Google%20Blacklist%3F,protects%20the%20users%20from%20these. [Accessed 21 April 2022].

BTEL, 2022. *Email Spoofing Explained*. Available at: https://www.btel.com/email-spoofing-explained/ [Accessed 21 April 2022].

Dmarcian, 2022. *Explained: What is DMARC?* Available at: https://dmarcian.com/why-dmarc/ [Accessed 21 April 2022].

Firebase, 2022. *Send notifications across platforms at no-cost*. Available at: https://firebase.google.com/products/cloud-messaging?gclid=CjwKCAjwo8-SBhAlEiwAopc9W8nptJ66U2MqnOxcmcsWJi6NObuveSNr4WHS0H0oJ2HQjwS_5XJr-BoCc10QAvD_BwE&gclsrc=aw.ds [Accessed 21 April 2022].

Google, 2022. *Build an add-on in any coding language*. Available at: https://developers.google.com/workspace/add-ons/guides/alternate-runtimes [Accessed 21 April 2022].

Google, 2022. *Card Service*. Available at: https://developers.google.com/apps-script/reference/card-service [Accessed 21 April 2022].

Google Apps Script, 2022. *Restrictions*. Available at: https://developers.google.com/apps-script/add-ons/guides/workspace-restrictions [Accessed 21 April 2022].

Google Cloud, 2018. *Elevating user trust in our API ecosystem*. Available at: https://cloud.google.com/blog/products/g-suite/elevating-user-trust-in-our-api-ecosystems [Accessed 21 April 2022].

Google Cloud, 2022. *Google infrastructure security design overview*. Available at: https://cloud.google.com/docs/security/infrastructure/design[Accessed 21 April 2022].

Hindawi, 2019. *PDRCNN: Precise Phishing Detection with Recurrent Convolutional Neural Networks*. Available at: https://www.hindawi.com/journals/scn/2019/2595794/tab2/ [Accessed 21 April 2022].

hMailServer, 2022. *Core features*. Available at: https://www.hmailserver.com/
[Accessed 21 April 2022].

Mimecast, 2021. *Phishing Facts and Statistics You Need to Know*. Available at:
https://www.mimecast.com/blog/phishing-statistics-facts/
[Accessed 23 April 2022].

Palo Alto Networks, 2022. *What Are Malicious Newly Registered Domains?* Available at:
https://www.paloaltonetworks.com/cyberpedia/what-are-malicious-newly-registered-domains[Accessed 21 April 2022].

Palo Alto Networks, 2021. *A Peek into Top-Level Domains and Cybercrime*. Available at:
https://unit42.paloaltonetworks.com/top-level-domains-cybercrime/
[Accessed 21 April 2022].

Phish Tank, 2022. *Join the fight against phishing*. Available at:
https://phishtank.org/
[Accessed 21 April 2022].

ReTruster, 2019. 2019 *Phishing Statistics and Email Fraud Statistics*. Available at:
https://retruster.com/blog/2019-phishing-and-email-fraud-statistics.html
[Accessed 23 April 2022].

Spamhaus, 2022. *The World's Most Abused TLDs*. Available at:
https://www.spamhaus.org/statistics/tlds/[Accessed 21 April 2022].

Acknowledgements

I would like to thank my supervisor, Chris Staff, for providing guidance and feedback throughout this project.