Phish Player
Record & Replay Phishing attacks

Karthika Subramani
Graduate Student
University of Georgia, Athens, Georgia

***Abstract*—Phish player is a Record & Replay tool that aims at recording web interactions of an user with phishing websites and replaying the steps. The purpose is for the tool to aid forensic analysts to understand how the attack happened in the first place.The attack is reconstructed by modifying chromium telemetry tool that was developed by Chrome basically intended to measure performance of websites. Since this reconstruction provides an exact simulation of how the user interacted with the phishin site, it provides a better insight in to the nuances of the phishing community. If sufficient analysis was done on the data collected , it may help detect the phishing sites in the initial stages and necessary actions could be taken on blocking those sites.**

***Index Terms*—Phishing, chromium Telemetry, Replay attacks**

I. Introduction

PHISHING is one of the stealthiest attacks in the security community. Most of its victims are unaware that they were attacked. In fact, a person could have all his personal information stolen by doing nothing but click a link or download a file that he believed was from a trusted source, whereas atually, he/she was actually tricked to reveal information to an untrustworthy source. The reason that these attacks are more prevalent is that they go unnoticed for a long time unless an expert in cyber forensics bother to look into the details of the attack. It is known that such forensics analysis is a difficult and time consuming task and involves a lot of manual work. Therefore, this paper focuses on developing a tool that would assist the forensic analysts to a great extent in understanding the attack.

A. Motivation

The main motivation of this project is to reduce the work of forensic analysts in understanding the attack that happened. In order to achieve this, the user interactions over web with the phishing sites are recorded including the details of the data that user provided. Once the data is recorded, it could be replayed by the analysts to observe how the attack occured and collect more information on the same. The collected information also prove to be useful in developing best defenses against these attacks in the future.

II. Design

The design of this tool included two phases: Record and Replay. The recording of data is done with the help of chrome extension that makes use of chrome development APIs to access the DOM of a web page in its HTMl state. The recorded data is then stored in JSON format. To reconstruct the attacks, the tool must be able to interact with the browser to reproduce the steps that were recorded. A tool that serves the purpose of interacting with the browser to execute certain commands is chromium telemetry. Certain modifications are made to this tool to deterministically replay the non-deterministic events the were recorded in the first phase.

A. Phase 1: Record

As mentioned before, the recording is done via chrome extension. The extension includes 1) A background file, 2) A content script that acts as an intermediate between the web page and background file. The content script has access to the page's DOM. Hence, it can be used to capture events and the data provided in the page. Further, the contentscript connects to the background page and posts the data to the background page. On the other hand, the background page listens to messages from the content script constantly and acts based on the message recieved. An overview of the working of extension is provided in figure 1

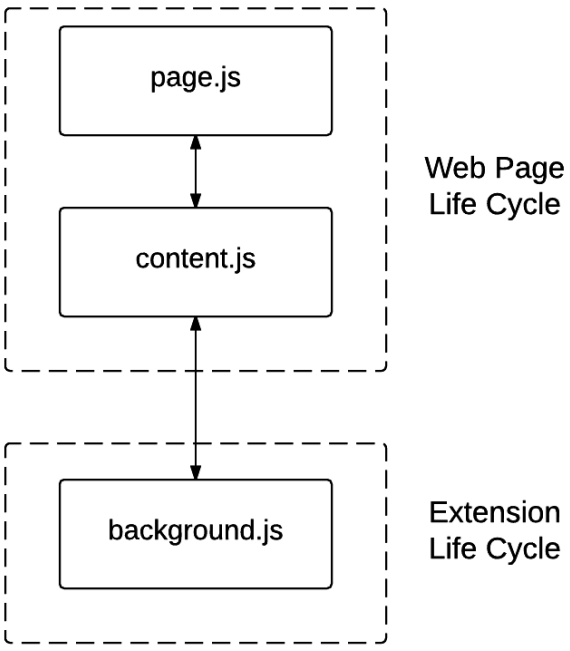

Fig. 1: An overview of chrome extension

B. Phase 2: Replay

The tool used for replay is telemetry, a performance testing framework used by chrome. It was designed in such a manner that the tests run on major platforms such as windows, Mac, Linux, Chrome OS, and Android for both Chrome and ContentShell. It runs on browser binaries, thus eliminating the need for building browsers to perform replay. A story in telemetry is an application scenario and a set of actions to run in that scenario such as clicking, scrolling etc. In order to accommodate our needs to replay the recorded data, a story is created that contains information on the reconstruction steps. Then the story is run as a page/story set using record command.

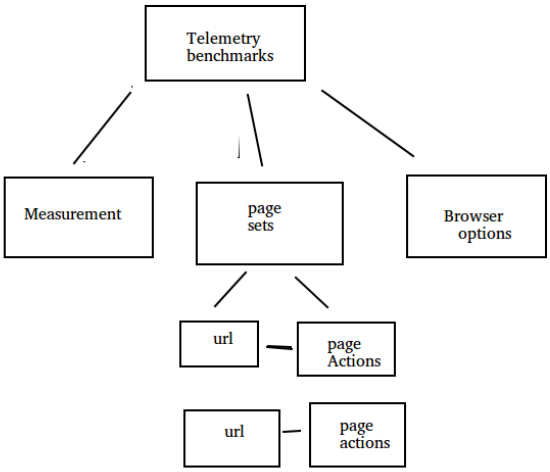

Fig. 2: The role of page set in chromium telemetry

III. Background

Verizon research has found that phishing is now the second most common cyber threat vector. According to a global phishing survey in trends and domain name use in second half of 2014, phishing attacks were not mitigated as quickly. The median uptime of phishing attacks increased to 10hours 6 minutes from 8hours and 42 minutes that waas observed in first half of 2016 and the average uptime was 29hours and 51 minutes. Also, a survey on the targets revealed that they were diverse and one of the reasons is to perform credit card theft, and hitting new targets may lull consumers into a false sense of security. Phishers aim to gather credentials, let it be a login information or credit card information. IT Governance also reports that once the emails have made it past filters, 8 million are opened, 800,000 recipients click on the links, and 80,000 of them unwittingly hand over their information to criminals. A google-led research paper also found that almost half(45%) of the visitors to a phishing web page completed the form and submitted their personal data.The figure 3 shows the statistics on the targeted fields.

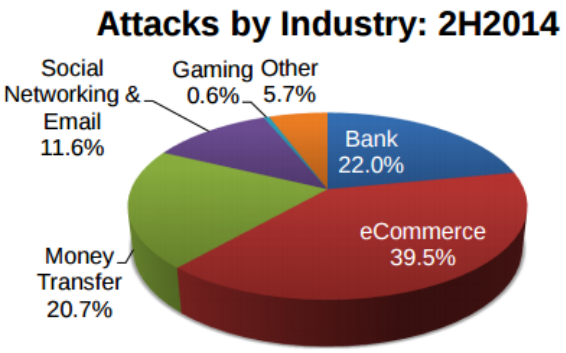

Fig. 3: A survey on the targeted fields of phishing sites

IV. Assumptions

Based on the above data, in order to efficiently capture the user interactions to a phishing site, certain assumptions were made.First assumption is that this tool may be used by majorly corporate companies or private networks which would be able to successfully enforce the addition of the recorder extension for the browser in the systems connected to network. Second, the users are made aware of their actions being recorded. Since most of the phishing websites aim at gathering credentials such as username, password, SSN number, credit card number, PIN etc, considering the design of the web page, these fields would be part of a form in a HTML page and they would have to be submitted for the attacker to actually obtain the information. Third, that all the esential information required in order to move to the next step is found inside a form of a web page.

V. Implementation

1) Phase 1: Record: Based on the assumptions, the goal of the record phase was not to successfully capture minute interactions of the user with the web page in a sequential manner such as mouse click, text change, selection change etc. The goal was to collect the data that was entered into the fields by the user and submit the form. In order to achieve this, the recorder needs access to the DOM of a page. So, developing a chrome extension proved to be a simple and efficient way to implement this requirement.

a) Recorder Extension: As already explained in the design section, the content script of the extension directly communicates with the web page and has access to its DOM content,though it would be run in an isolated environment different from that of the page's environment. On the other hand, baackground script has complete access to the chrome development APIs and can control and monitor actions such as tab creation, session tracking, etc. The recorder for the phish player tool was implemented such that once the extension is included in the page, the content script waits for the web page to load and tries to connect to the background page. Thus, the background page obtains information such as the tab from which the script connected and the port number using which it can communicated to the conten script at any point of time.

b) Content Script: The content script is unique to each web page / tabs in the browser. This scripts waits for the page to successfully load and an event listener is dynamically added for the submit event of the page. The listener gets triggered on submission of a page and the listener obtains the data of the form submitted using an inbuilt jQuery function and parses it to JSON format. This information basically includes the name of a field and its value. Note that in order for the field to be recogonized by the form on submission, the fields are required to have an attribute name. Hence, teh detail name and value are collected. Further, these details are used to obtain additional information about the fields such as tag name and specific type of the node as the tag name for most input fields in HTML is ĬNPUT. One other detail teh form submission data includes are the details of the hidden fields and their values. The hidden fields doesn't concern this tool and are hence ignored. On forming the required data, the content script connects to the background page and sends the data to the background page via postMessage API.

c) Background Page: After the initial connection to a content script, the background page constantly listens for messages from the content script of the tab. the data that is obtained is stored temporarily associated to the tab id. One scenario that needs to be understood is that a phishing site may have multiple pages and successful submission of one page leads to another where more data is requested in forms. So, a recorder needs to capture the data of sequential forms. Thus, the background page collects information over a session since the opening of a tab to its close and send the data to a server file which is required to save it to a file in the form of JSON data.

2) Phase 2: Replay: The data that is already stored by the extension is to be used to simulate the same experience. The Chromium telemetry has built-in page sets which are stories that define a sequence of actions. The data in the json file is read to generate the sequence of actions dynamically. Based on the assumptions, it is sufficient if the page is loaded, the data are filled in their necessary fields and a submit event is triggered. In order to load the page, chromium telemetry provides a wrapper called Navigate that takes in the url to be navigated to. In order to fill in the data in their respective fields, based on the type of the node, different wrapper methods are written to populate the data such as setting the value of a field, selecting the correct index or value from a dropdown control etc. The telemetry provides a wrapper class action_runner which consists of methods such as navigate, executeJavascript, waitForelement, WaitForDocumentReady etc that are used to achieve the reconstruction of events.

a) Page set: A page set is how the reconstruction is initiated. The telemetry tests are done via benchmarks, page sets or unit tests. Page sets are more suitable to the requirements and hence a pageset called Replay_Page was created. This page internally makes a call to a helper class that initiates the loading of the page and execution of events

b) Helper class: This helper class is nothing but a wrapper class that reads the JSON file, parses its contents and calls the appropriate methods to execute each event such as loading the page,filling the data, selecting data and submitting the page. The execution continues to the next page after the submit event as long as it reaches the end of the recorded JSON data. The JSON data itself is stored along with the other data in the data folder of the telemetry code.

VI. Experiments

Phishtank, a popular website provides information on the existing and newly discovered phishing sites. In order to evauate the phish player tool, certain phishing sites from phishtank website was selected and the tool was used to record and replay the interaction with these websites. Four specific test cases are explained in the following sections

3) Case Study 1: A facebook phishing site: This site had the users follow through multiple steps gathering their login information and credit card and personal information. During this process, a confirmation page was shown to gain user's trust and confidence. a second payment page was shown asking for the credentials mentioning that the details entered in the previous page was incorrect. These kind of tricks has been observed in othe rphishing sites as well. It could be a noteworthy trait that helps distinguish phishing sites from legitimate sites.Finally, once all the required information was gathered, the user was navigated to the legitimate facebook site. The screenshots of the multiple steps of this site is shown in the following figures

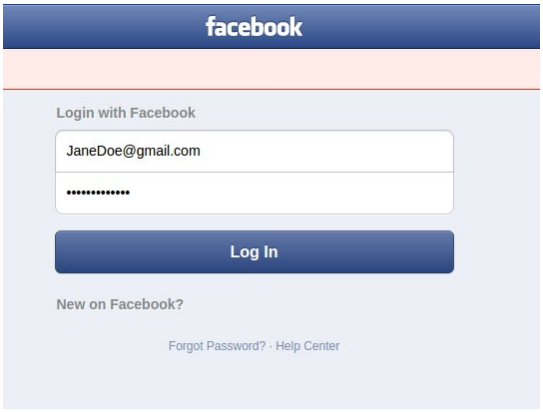

Fig. 4: Facebook fake login page

Fig. 5: Facebook fake confirmation page

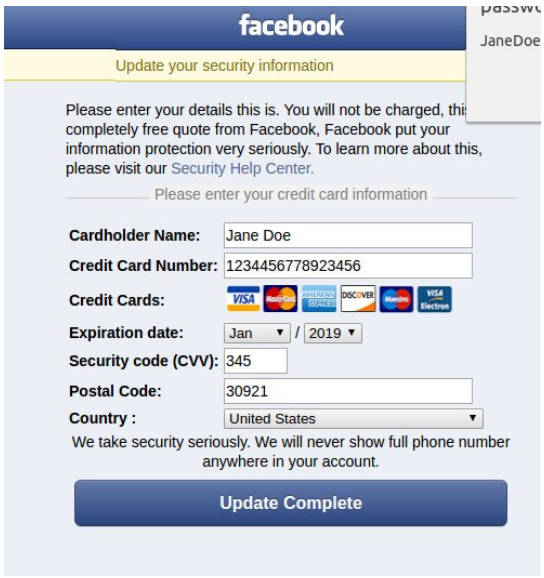

Fig. 6: Credit Card details page

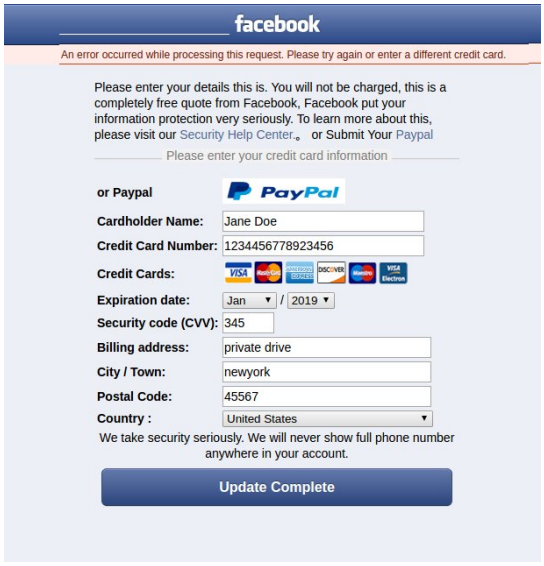

Fig. 7: Credit Card Confirmation details page

Fig. 8: Legitimate facebook page

4) Case Study 2 : A Microsoft Office phishing attack : This attack focussed on obtaining a user's microsoft office account credentials by opening an excel sheet and requesting for authentication in orderto proceed and download the sheet. Once the credentials were provided the user was navigated to a sample excel file.

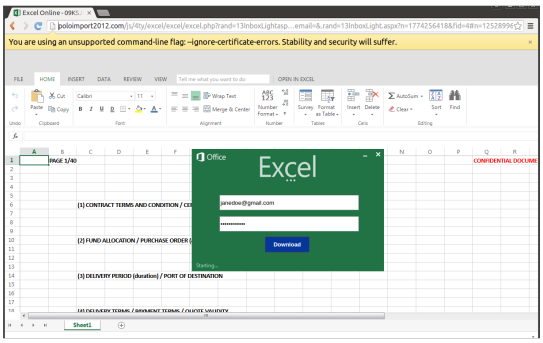

Fig. 9: Requesting MS office credentials

5) Case Study 3 : An attack on Apple Account Credentials: It has been observed that apple is one of the most targeted company by phishing attackers. This specific site included two pages, first for collecting the account credentials and the billing adrees of the user and second, to collect the credit card information. The unique thing about this would be the check for valid credit card number format. This site refused to take in any 16 digit number for credit card number , whereas other sites didn't bother to check for the corrcet format.

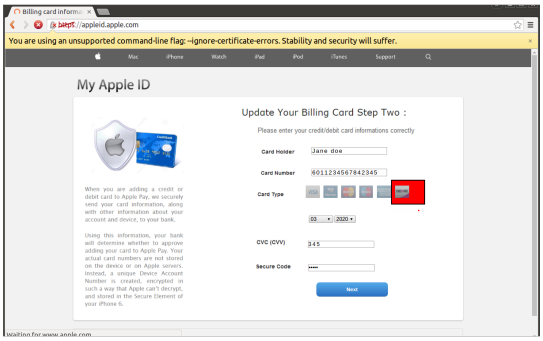

Fig. 10: Credit Card details update form

6) Case Study 4 : A Pay pal phishing site: This site was again well-built to be considered as a fake site. It starts by requesting the user to login to the pay pal fake web site, dislays a progressing window and redirects to a page that helps a user to update the credit/ debit card information. This page also specifically requested for the Social Security Number.

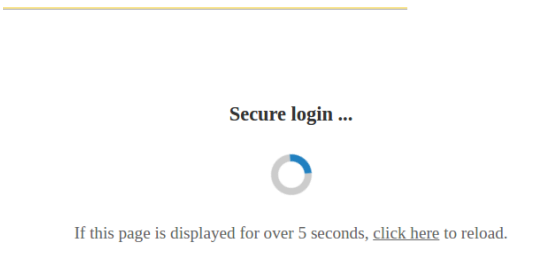

Fig. 11: Progress page

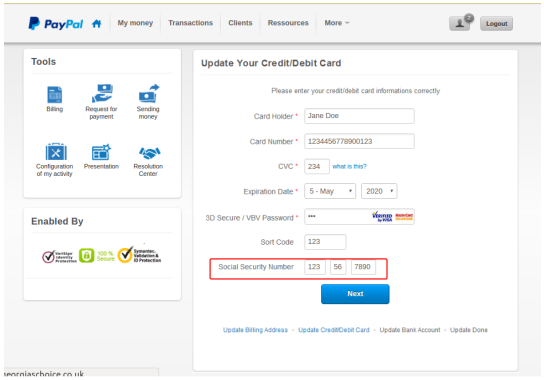

Fig. 12: A page requesting SSN information

VII. Observations

The phishing sites discussed in this paper had remarkably similar appearances to their targeted sites, that it would not make an user doubt its validness, especially, if the site is opened from mobile devices that hides the address bar. An observation that was made during this project is the lifetime availability of the phishing sites. Most of the phishing sites that were considered were not available in a short period of time. Hence, once the phishing sites are identified, they are blocked or the attacker changes the domain of the site. But, blocking the sites doesn't completely solve the problem. They are moved to different domains and still many users become victims of these attacks. According to a survey made by an industry APWG in 2014, the following figure shows the data about the average and median uptime of phishing websites

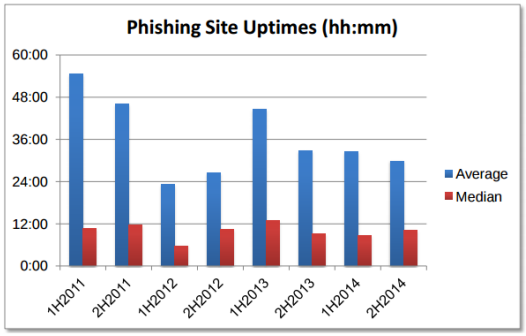

Fig. 13: Average and Median uptimes of phishing sites

VIII. Conclusion

The ultimate goal of this paper is to help identify the sites as phishing sites in a faster manner comapared to existing methods. The automated reconstruction of the interactions would prove to be much easier than manual analysis of the log files and manual reconstruction of the events. As far as the results of the experiments, almost all the interactions that involved a form and submission was successfully recorded and reconstructed.

IX. Future Work

Considering the vast availability of phishing sites and their different nature, the recording and reconstruction could be expanded to actually capture user's interactions such as page scroll, mouse click in order to cover all types of phishing sites which includes not just the ones that focus on gathering credentials. Some of the phishing sites included tricking user to fill out a survey in order to recieve some gifts posing they are from a trusted company.

Acknowledgement

I would like to thank Dr.Roberto Perdissci, Associate Professor in University of Georgia for his guidance and support on this project.

References

[1] Google Chrome, *Chrome Extensions*, VerfL̈ink: https://developer.chrome.com/extensions/getstarted

[2] Google Chrome, *Chromium Telemetry*, VerfL̈ink: https://www.chromium.org/developers/telemetry/run_locally

[3] Christopher Neasbitt, Bo Li,Roberto Perdisci, Long Lu, Kapil Singh and Kang Li, *WebCapsule: towards a Lightweight forensic Engine for Web Browsers*, Verfügbar unter: http://www.longlu.org/downloads/ccs2015.pdf

[4] Agari, *Cisco's IronPort solutions*, Verfügbar unter: https://www.agari.com/phishing-statistics/

[5] APWG, *global Phishing Report*, Verfügbar unter: http://internetidentity.com/wp-content/uploads/2015/05/APWG_Global_Phishing_Report_2H_2014.pdf