Detecting Malicious Webpages using Virustotal API

Benazeer
Dept. of CSE, P.D.A College of Engineering, Gulbarga, Karnataka, India

Abstract
Cell phone specific webpages differ significantly from their desktop equivalent in content, layout and functionality. Consequently, existing techniques to find malicious websites are not likely to work for such webpages. In this work we use a Virustotal API. Virustotal API is a mechanism that identifies malicious webpages. Virustotal makes this determination based on dynamic features. The Virustotal API lets the user scan the URL's and search through the dataset to determine malicious web pages. We tentatively exhibit the requirement for mobile specific procedures and after that recognize a scope of new successful highlights that exceptionally correspond with mobile malicious sites. Moreover, we discover, define, report quantity of webpages missed by Google safe browsing, and detected by our application.

Keywords
Websites, Web Browser, Virus Total & Mobile Security

I. Introduction
Malignant programming regularly known as Malware is basically programming that is expected to invade a PC framework without the assent of the system's owner. It is an example of malicious code with expectation to hurt a PC or system. It covers a scope of dangers like infection, Trojans, adware's, spywares, and so on. They replicate themselves and enter into the system in different ways; either multiple media or through the most popular way of getting downloaded into the system as the genuine application. Since various malware discovery framework has been acquainted till date with bypass the attacks caused by malwares.
There are two unique procedures static and dynamic to dissect malware infected records. Here we goes with Dynamic analysis which is also known as behavioral analysis, in dynamic analysis detection of malware relies on information that is collected from the working framework at run-time (i.e., during the execution of the program, for example, framework calls, arrange access and records. Previous technique fail to detect the mobile malicious Webpages, hence whenever a mobile malicious webpage is detected on desktop browser it sends a blank message or unable to detect specific mobile website. Therefore existing tools such as Google Safe Browsing are not enabled on the mobile versions of browsers[10]. It work for work area program and DNS based instruments don't give further comprehension of the particular action executed by a page or space. Hence to overcome the above problem we are using virus total API.

Virus Total is a free apparatus that can be downloaded and gotten to online to investigate suspicious documents, hashes or URLs. It utilizes various Antivirus motors to encourage the location of various malwares including infections, worms, and Trojans . In addition to Antivirus engines, Virustotal uses website scanners to analyze and detect any malicious content available in URLs or files. It utilizes 60 Antivirus items, for example, Kaspersky Lab, Specialist Web, AVG Advancements, Cyren, up to 62 Site/domain examining motors and datasets, for example, AutoShun,CRDF, Sucuri SiteCheck, Quttera, and up to 18 file characterization tools and datasets such as Exit Tool, Snort and Wire shark.

Virustotal gives two sorts of administrations,[9] a free open interface for clients to submit documents for examination and a paid administration that exposes significantly more information through a search interface about the files that were submitted. Aside from the information about the record submitted, Virustotal additionally furnishes metadata related with the individual who submitted the document. Research has reasoned that it's conceivable to predict and distinguish casualties, on-screen characters and different kinds of records in light of their transfers to Virustotal over a period of time using the submission data associated with the document they upload. In this project we will outline and discuss the process associated with the Virustotal API coming to this conclusion we experimentally demonstrate the need for mobile particular methods and after that recognize a scope of new unique highlights that highly connect with mobile malicious website pages.

II. Related Work
The majority of malware infections begin with a remote malicious executable download. Two common infection vectors are drive-by downloads and social engineering [1]. Drive-by downloads exploit an unmatched vulnerability in the browser. Social engineering is also a common infection vector. However, the target of the attack is the user, not software. The attacker uses persuasion [2] and deception [3] techniques to convince the user to download and install malware.
One benefit for the attacker in using social engineering is that his attack can be successful even on systems that have a limited attack surface and no known exploitable vulnerabilities. Furthermore, social engineering attacks are less likely to be identified by modern detection systems since their focus is on exploitation [4, 5].Blacklisting is a popular technique that is used to detect and block malicious domains that are associated with exploitation, social engineering and malware downloads [6]. The primary issue with blacklists is that they are always out-of-date. For example, in the case of drive-by downloads, the domains that are typically blacklisted are the ones associated with the exploit and malware executable. However, these domains are typically only in use for a day or less. So by the time they are blacklisted, the attackers are already using new domains to host their exploit kit and malware [1] or plug-in. The exploit gains control of the application by pointing its execution at code (i.e., shell code) controlled by the attacker. The shell code then downloads and executes the malware. Often, a drive-by download attack will go unnoticed by the victim and they will be unaware of the compromise.
Sandboxes provide a controlled environment to run executables. The outputs of a sandbox are system and network traces. This information is used by analysts and automated systems to identify malware. However, malware often uses anti-sandbox technique such as Virtual Machine (VM) detection and timing measurements to detect the sandbox and proceed with a benign code path that has no suspicious behaviour [7]. The domain name service (DNS) plays an important role in the operation of the Internet, providing a two-way mapping between domain names and their numerical identifiers. Given its fundamental role, it is not surprising that

a wide variety of malicious activities involve the domain name service in one way or another [9]. Thus, it seems beneficial to monitor the use of the DNS system for signs that indicate that a certain name is used as part of a malicious operation. Phishing is type of wholesale fraud that combines social engineering systems and advanced assault vectors to gather financial data from clueless customers. Regularly a phishes endeavors to bait her casualty into clicking a URL indicating a maverick page [10]. The structure of URLs utilized in different phishing assaults. We find that usually conceivable to tell regardless of whether a URL has a place with a phishing assault without requiring any information of the relating page information. a few highlights that can be utilized to recognize a phishing URL from a benign one.

These highlights are utilized to demonstrate a strategic relapse channel that is productive and has a high accuracy. We utilize this channel to perform exhaustive estimations on a few million URLs and measure the commonness of phishing on the Web today. Downside is URL-based procedures generally experience the ill effects of high false positive rates. Static approaches Rely on the structural and Lexical properties A webpage and do not execute the Content Of the webpage. One such system of malicious URLs is utilizing measurable Strategies for URL Order in light of a URL's lexical and host-based Properties. In any case, URL-based Methods for the most part experience the ill effects of high false Positive rates. Utilizing HTML and JavaScript Highlights removed from a Website page Not withstanding URL arrangement helps address. The drawback for this approach is able to provide deeper visibility into the Behavior of Web page. To avoid this drawback Dynamic approaches is used. Dynamic approach uses Virtual machines and honey client systems [11]. Therefore, such Systems have a very low false positive rate and are more accurate. Be that as it may, Downloading and executing every site page Effects Execution and upsets Versatility of dynamic methodologies. All these approaches for malicious Web page detection have focused on Websites built for desktop browsers in the Past. Mobile browsers have been shown to Differ From their desktop counterparts in Terms of security Although Differences in mobile and desktop websites Been observed before [12], it is Unclear How these differences impact security. Furthermore, the threats on mobile and Desktop websites are somewhat different Static analysis techniques using Features of desktop webpages have been Primarily studied for drive-by-downloads

On desktop websites whereas, the Biggest threat on the mobile web at present Is believed to be phishing Efforts in Relieving phishing attacks on desktop Web sites incorporate separating program Utilizations of various trust level Email sifting utilizing content-based Highlights and boycotts The Best-known non-restrictive Substance based Way to deal with recognize phishing website pages is Saloon. Saloon experiences Execution issues Because of the time slack Associated with questioning the Google Internet searcher. Also, Bar does not function admirably On website pages written in dialects Other than English. At long last, existing Strategies don't represent new versatile site pages.

III. Methodology

This section discusses the algorithms developed to collect URL from different types of websites, and how the tools Virustotal, are used to classify the collected Webpages as benign or malicious.

1. Data Collection
2. Malware detection using Virustotal API

1. Data Collection

In the initial step, we built a collection of URL's by crawling a sum of 600 sites. We choose these sites from two unique information sustains: a good and an bad feed, to ensure we have sensible number of website pages with benign and malicious content. The good data feed was taken from Alexa's [11] top million websites list while the bad data feed was taken from a blacklist, which contains web pages that have malicious behaviors such as spyware, reducing bandwidth use, and embedding in spam. We slithered 400 sites from Alexa's rundown feed 200 were chosen from the highest point of the feed and another 200 were selected from the middle of the feed. Moreover, we crawled 200 websites from the blacklisting feed: 100 were selected from the top of the feed and the other 100 were selected from the bottom of the feed.

2. Malware Detection Using Virustotal

Subsequent to getting to 200 sites and URLs.

Next, we executed a framework that peruses these URLs, each one in turn and consequently submit them to Virustotal site. For every URL, it opens Virustotal site, looks for the URL content field, puts the URL, and taps on the "Output it!" catch. Concentrates the identification proportion and breaks down it. On the off chance that the location proportion is higher than one, at that point it will characterize the website page as a malicious, save the URL in "VirusTotalResults.txt" record and additions malicious counter by one. Else, it will increase the benign counter by one. Toward the end, we will get the quantity of malicious website pages grouped by Virustotal. Algorithm shows the pseudo code of our Algorithm.

Algorithm Detecting Malware Using Virustotal
1: benign ← 0 int
2: malicious ← 0 int
3: driver ← new ChromeDriver() WebDriver
4: Open "URLs: txt" ForInputAsInputFile
5: Open "VirusTotalResults:txt" For Output as Outut File
6: ReadLinefromInputFile(String)
7: loop:
8: if Line 6= null then
9: driver:browse("https : ==virustotal:com=")
10: driver:findURLTabElement:click()
11: driver:findURLTextFieldElement:write(Line)
12: driver:findScanItButton:click()
13: if DetectionRatioText 6= 0 then
14: malicious malicious+1
15: WriteLineToOutputFile(String)
16: else
17: benign ← benign+1
18: ReadLinefromInputFile(String)
19: goto loop.
20: close;
21: Write"malicious="maliciousToOutputFile(String)
22: Write"benign = "benignToOutputF ile(String)
23: driver:close

IV. Design and Implementation

We describe the dynamic techniques of Virustotal API to tackle the problem of classifying mobile specific web pages as malicious. We build and evaluate our chosen model for accuracy.

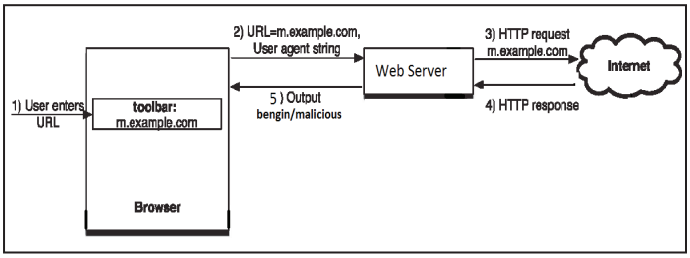

Fig. 1: System Architecture

Virustotal inspects items with over 70 antivirus scanners and URL/ domain blacklisting services, Any user can select a file from their computer using their browser and send it to Virustotal. Virustotal offers a number of file submission methods, including the primary public web interface, desktop up loaders, browser extensions and a programmatic API. As with files, URLs can be submitted via several different means including the Virustotal webpage, browser extensions and the API.

Upon submitting a file or URL basic results are shared with the submitter, and also between the examining partners, who use results to improve their own systems. As a result, by submitting files, URLs, domains, etc.

This core analysis is also the basis for several other features, including the Virustotal Community: a network that allows users to comment on files and URLs and share notes with each other. Virustotal can be useful in detecting malicious content and also in identifying false positives -- normal and harmless items detected as malicious by one or more scanners.

V. Conclusions

Mobile web pages are significantly different than their desktop counterparts in content, functionality and layout. Therefore, existing techniques using static features of desktop web pages to detect malicious behaviour do not work well for mobile specific pages. We designed and developed an application that detects mobile malicious web pages. It provides accuracy in classification, and detects a number of both mobile and desktop malicious web pages in the wild that are not detected by existing techniques such as Google Safe Browsing. We conclude that an application detects both mobile and desktop specific threats such as websites and takes the first step towards identifying new security challenges in the modern mobile web. In future we implement a technique that detects more web pages on both mobile and desktop.

References

[1] Nelms, T., Perdisci, R., Antonakakis, M., Ahamad, M., "Webwitness: Investigating, categorizing, and mitigating malware download paths," In Proceedings of the 24th USENIX Conference on Security Symposium, SEC'15, (Berkeley, CA, USA), pp. 1025 {1040, USENIX Association, 2015.

[2] Cialdini, R. B. Pearson Education, 5th ed., 2000.

[3] Whaley, B.,"Toward a general theory of deception", 1982. Miliary Deception and Strategic Surprise.

[4] Snort 3.0. [Online] Available: http://www.snort.org/snort-downloads/snort-3-0/.

[5] Paxson, V.,"Bro: A system for detecting network intruders in real-time," Computer Network., Vol. 31, pp. 2435-2463, 1999.

[6] Google safe browsing api. [Online Available: https://developers.google.com/safe-browsing/

[7] Dinaburg, A., Royal, P., Sharif, M., Lee, W.,"Ether: malware analysis via hardware virtualization extensions," In Proceedings of the 15th ACM conference on Computer and communications security, CCS '08, (New York, NY, USA), pp. 51-62, ACM, 2008.

[8] Li, C., Peng, G., Gopalan, K., Cker Chiueh, T.,"Performance guarantee for cluster-based internet services", in Parallel and Distributed Systems, 2002. Proceedings. Ninth International Conference on, pp. 327-332, 2002.

[9] Virustotal. [Online] Available: https://www.virustotal.com/en/.

[10] Google developers: Safe Browsing API. [Online] Available: https://developers.google.com/safe-browsing/, 2012.

[11] Alexa, the web information company. [Online] Available: http://www.alexa.com/topsites, 2013.

[12] Dotmobi. internet made mobile. anywhere, any device. [Online] Available: http://dotmobi.com/, 2013.

[13] C. Amrutkar, K. Singh, A. Verma, P. Traynor,"Vulnerable Me: Measuring systemic weaknesses in mobile browser security". In Proceedings of the International Conference on Information Systems Security (ICISS), 2012.

[14] C. Amrutkar, P. Traynor, P. C. van Oorschot,"Measuring SSL indicators on mobile browsers: Extended life, or end of the road?", In Proceedings of the Information Security Conference (ISC), 2012.

[15] L. Bilge, E. Kirda, C. Kruegel, M. Balduzzi,"EXPOSURE : Finding malicious domains using passive DNS analysis", In Proceedings of the 18th Annual Network and Distributed System Security Symposium (NDSS), 2011.

[16] M. Boodaei,"Mobile users three times more vulnerable to phishing attacks. [Online] Available: http://www.trusteer.com/blog/mobile-users-threetimes-more-vulnerable-to-phishing-attacks, 2011.

[17] D. Canali, M. Cova, G. Vigna, C. Kruegel,"Prophiler: a fast filter for the large-scale detection of malicious web pages. In Proceedings of the 20th International Conference on World Wide Web (WWW), 2011.

[18] P. Likarish, E. Jung, I. Jo.,"Obfuscated malicious javascript detection using classification techniques", In Proceedings of Malicious and Unwanted Software (MALWARE), 2009.

[19] C. Ludl, S. Mcallister, E. Kirda, C. Kruegel,"On the effectiveness of techniques to detect phishing sites", In Proceedings of the 4th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment(DIMVA), 2007.

[20] J. Ma, L. K. Saul, S. Savage, G. M. Voelker,"Beyond blacklists: Learning to detect malicious web sites from suspicious URLs", In Proceedings of the SIGKDD Conference, 2009.

[21] Y. min Wang, D. Beck, X. Jiang, R. Roussev, C. Verbowski, S. Chen, S. King,"Automated web patrol with strider honeymonkeys: Finding websites that exploit browser vulnerabilities. In Proceedings of the Networking and Distributed Systems Security (NDSS), 2006.

[22] A. Moshchuk, T. Bragin, S. D. Gribble, H. M. Levy. A crawler-based study of spyware on the web", In Proceedings of Network and Distributed System Security Symposium (NDSS), 2006.

[23] J. Nazario. Phoneyc: A virtual client honeypot", In Proceedings of the 2nd USENIX conference on Large-scale Exploits and Emergent Threats: botnets, spyware, worms, and more (LEET), 2009.