International Journal of u- and e- Service, Science and Technology
Vol.8, No.5 (2015), pp.195-206
http://dx.doi.org/10.14257/ijunesst.2015.8.5.18

Survey on Malicious Web Pages Detection Techniques

Dharmaraj Rajaram Patil[1] and J. B. Patil[2]

[1]*Research Scholar, Department of Computer Engineering, R. C. Patel Institute of Technology, Shirpur (MS), India*
[2]*Professor, Department of Computer Engineering, R. C. Patel Institute of Technology, Shirpur (MS), India*
[1]*dharmaraj.rcpit@gmail.com*, [2]*jbpatil@hotmail.com*

Abstract

The World Wide Web has become an inseparable part of millions of people who use online services e.g. online banking, online shopping, social networking, e-commerce, and store and manage user sensitive information, etc. In fact, it is a popular tool for any class of user over the Internet. Rich Web based applications are available over the World Wide Web to provide such types of services. At the same time, the Web has become an important means for people to interact with each other and do business. This is the positive side of this technology. Unfortunately, the Web has also become a more dangerous place. The popularity of World Wide Web has also attracted intruders and attackers. These intruders abuse the Internet and users by performing illegitimate activity for financial profit. The Web pages that contain such types of attacks or malicious code are called as malicious Web pages. While the existing approaches are good indicators in detecting malicious Web pages, there are still open issues in Web page features selection and detection techniques. In this paper, we are giving an extensive survey of existing malicious Web pages detection approaches and features they have used.

Keywords: *Malicious Web pages, Detection, Web Page Features, Machine Learning.*

1. Introduction

Malicious Web content has become the primary tool used by attackers to perform attacks on the Internet. In 2007, N. Provos et al. found more than three million URLs that launched drive-by-download attacks [1]. In particular, attacks that target Web clients have become pervasive. According to B. Liang et al. 29 of 90 Websites contained malicious code [2]. According to D. Canali et al. attackers frequently use drive-by-download exploits to compromise a large number of users [3]. To perform a drive-by-download attack, the attacker first craft malicious client-side scripting code typically written in JavaScript that targets vulnerability in a Web browser or in one of the browser's plug-ins. This code is injected into compromised Websites or is simply hosted on a server under the control of the criminals. When victim visits a malicious Web page, the malicious code is executed and the victim's browser is compromised for future attacks. As a result the victim's computer is typically infected with malware. According to the Symantec Internet Security report released in 2014, malicious Web pages are now the primary vector for malicious activities over the Internet and some of the highlights from the threat landscape of 2014 are [4]:

1. 91% increase in targeted attacks campaigns in 2013.
2. 62% increase in the number of breaches in 2013.
3. Over 552M identities were exposed via breaches in 2013.
4. 23 zero-day vulnerabilities discovered.
5. 38% of mobile users have experienced mobile cybercrime in past 12 months.
6. Spam volume dropped to 66% of all email traffic.

ISSN: 2005-4246 IJUNESST

7. 1 in 392 emails contain a phishing attacks.
8. Web-based attacks are up 23%.
9. 1 in 8 legitimate Websites have a critical vulnerability.

Websense, a known security firm, recently published the 'Websense Security Labs 2014 Threat Report' stating the details of threats and trends marked in 2013 [5],

1. 85% of the malicious links spotted in emails or Web attacks during last year pointed towards authentic Websites which were hijacked by cybercriminals.
2. 3.3% of all spam messages contained malicious links with other malicious content, highlights the recently released report.
3. Websense stopped 1.8 billion of malicious redirects in 2013 and the company found four redirects per attack on an average with maximum 20 redirects in a single attack.
4. The report confirms around 67 million exploit kit events during 2013 with the Neutrino and Magnitude Exploit Kits experienced the biggest surge in adoption following the imprisonment of Paunch, creator of Blackhole.
5. 64 million dropper file events were identified and 30% of malevolent executable files were sampled containing custom encryption of C&C (command and control) communication or information exfiltration.

In this paper, we present an extensive survey of existing malicious Web pages detection approaches and the set of Web page features they have used. Also we highlight our ongoing efforts towards effective detection of malicious Web pages. The paper is structured as follows. In section 2, we present the Web attacks taxonomy. Section 3 covers the extensive survey of related work, focusing on the methods and Web page features used by researchers for classification of Web pages as malicious or benign. Finally, we summarize the key principles of malicious Web pages detection in Section 4. In Section 5, we present our conclusions.

2. Web Attacks Taxonomy

New Web-based attacks are coming out every day, this is forcing businesses, communities and individuals to take security issues seriously. Following are some of the common Web-based attacks against Websites and Web applications.

2.1 Drive-By Downloads Attacks

A drive-by-download attack is a malware / virus / shell code delivery technique that is activated simply because the user visited a Website. Drive-by-download attacks occur when a visitor navigates to a site that injects malware onto the victim's PC. A drive-by download can be initiated by simply visiting a Web site or viewing an HTML E-mail message. Basically, these attacks are usually downloaded and run in the background in a manner that is invisible to the user. Drive-by downloads continue to be a major security issue online. In April 2007, researchers at Google discovered hundreds of thousands of Web pages that initiated drive-by downloads. One in ten pages was found to be suspect. Sophos researchers in 2008 reported that they were discovering more than 6,000 new infected Web pages every day, or about one every 14 seconds [6].

2.2 Clickjacking Attacks

Clickjacking or Clickjack attack is a Web vulnerability used by an attacker to collect an infected user's clicks. The Clickjacking attack allows to perform an action on victim site on visitor’s behalf [7].
The overall idea is simple.

1. A visitor is lured to a vulnerable Web page. Like, “Click to get 1000000 Rs.” Or whatever.
2. The vulnerable Web page puts a “Get Rich Now” link with z-index=-1.

3. The vulnerable Web page puts includes a transparent iframe from the victim domain,
say facebook.com and positions it so that “I like it” button is right over the link.

The following code snap shows a typical example of clickjacking attack,

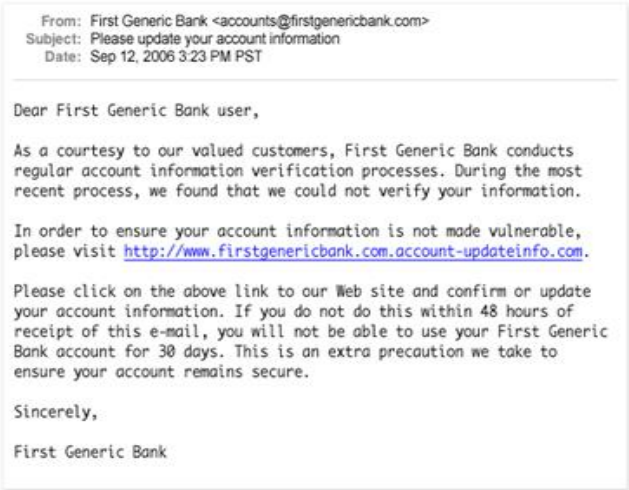
```
<style>
iframe
{ /* iframe from facebook.com */
  width: 300px;
  height: 100px;
  position: absolute;
  top: 0; left: 0;
  filter: alpha (opacity=50); /* in real life opacity=0 */
  opacity: 0.5;
 }
</style>
<div> Click on the link to get rich now: </div>
<iframe src="/files/tutorial/window/clicktarget.html"></iframe>
<a href="http://www.google.com" target="_blank" style="position: relative; left: 20px; z-
 index:-1">CLICK ME! </a>
<div> You'll be rich for the whole life! </div>
```

2.3 Plug-ins and Script-Enabled Attacks

Attackers target vulnerabilities in plug-ins or insert malicious code, usually JavaScript, into a Web application’s output. These types of vulnerabilities help attackers to carry out drive-by download and Clickjacking attacks.

2.4 Phishing Attacks

Phishing is a fraudulent attempt, usually made through E-mail, to steal your personal information, appearing to come from legitimate enterprises (e.g. your university, your Internet service provider, your bank). These messages usually direct you to a spoofed Website or otherwise get you to provide your private information (e.g. password, credit card or other account updates). The attackers then use this private information to commit identity theft. Phishing E-mails will always tell you to click a link that takes you to a site where your personal information is requested. Legitimate organizations would never request this information of you via E-mail [8].


From: First Generic Bank <accounts@firstgenericbank.com>
Subject: Please update your account information
Date: Sep 12, 2006 3:23 PM PST

Dear First Generic Bank user,

As a courtesy to our valued customers, First Generic Bank conducts regular account information verification processes. During the most recent process, we found that we could not verify your information.

In order to ensure your account information is not made vulnerable, please visit http://www.firstgenericbank.com.account-updateinfo.com.

Please click on the above link to our Web site and confirm or update your account information. If you do not do this within 48 hours of receipt of this e-mail, you will not be able to use your First Generic Bank account for 30 days. This is an extra precaution we take to ensure your account remains secure.

Sincerely,

First Generic Bank

Figure 1. Phishing E-mail / Phishing Website Example

2.5 Social (Engineering) Networks Attacks

Social engineering is the art of manipulating people so they give up confidential information. The types of information the attackers are seeking can vary, but when individuals are targeted, the attackers are usually trying to trick you to give them your passwords or bank information or access your computer to secretly install malicious software. Attackers use social engineering tactics because it is usually easier to exploit. It is easier to fool someone into giving you their password than it is for you to try hacking their password [9].

Types of social engineering attacks, social engineering can be broken into two common types:

2.5.1 Human Based: These types of social engineering attacks needs interaction with humans, it means person-to-person contact and then retrieving the desired information. Some examples are as follows,

a. Impersonation
b. Posing as an important user
c. Being a third party
d. Desktop support
e. Shoulder surfing
f. Dumpster diving

2.5.2 Computer Based: Computer-based social engineering uses computer software that attempts to retrieve the desired information. Some examples are as follows,

a. Phishing
b. Baiting
c. On-line scams

2.6 Cross Site Scripting (XSS) Attacks

Cross-site scripting also known as XSS is generally a common application layer hacking techniques. In a XSS attack the hacker infects a legitimate Web page with his malicious client-side script. When a user visits this Web page the script is downloaded to his browser and executed. The XSS attack follows some pattern given in the following diagram [10].

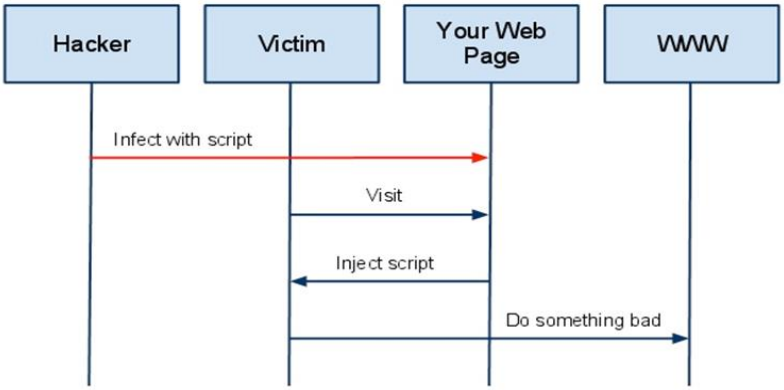


Figure 2. A High Level View of a Typical XSS Attack

XSS attacks are broadly classified into 2 types:

2.6.1 Non-Persistent XSS Attack: In case of Non-Persistent attack, it requires a user to visit the specially crafted link by the attacker. When the user visits the link, the crafted code will get executed by the user's browser. Following script show the typical non-persistent XSS attack,

```
index.php
<? php
$name = $_GET['name'];
echo "Welcome $name <br>";
echo "<a href="http://xssattackexamples.com/">Click to Download </a>";
?>
index.php? name = guest<script>alert ('attacked') </script>
```

2.6.2 Persistent XSS Attack: In this type of XSS attack, the code is injected by the attacker will be stored in a database. The damage caused due to this XSS attack is more than the non-persistent XSS attack.

2.7 SQL Injection Attacks

SQL injection is a technique where malicious users can inject SQL commands into an SQL statement, via Web page input. Injected SQL commands can alter SQL statement and compromise the security of a Web application. The primary form of SQL injection consists of direct insertion of code into user-input variables that are concatenated with SQL commands and executed. A less direct attack injects malicious code into strings that are destined for storage in a table. When the stored strings are subsequently concatenated into a dynamic SQL command, the malicious code is executed. The following script shows a simple SQL injection attack [11].

```
var Shipcity;
ShipCity = Request.form ("ShipCity");
var sql = "select * from OrdersTable where ShipCity = '" + ShipCity + "'";
```

The user is prompted to enter the name of a city. If the user enters Redmond, the query assembled by the script looks similar to the following:

```
SELECT * FROM OrdersTable WHERE ShipCity = 'Redmond'
```

This script builds an SQL query by concatenating hard-coded strings together with a string entered by the user.

2.8 Third-Party Web Apps Attacks

As more employees do work via third-party Web applications, criminals are having a chance to use these apps to compromise endpoints and enterprise networks. If policies for third-party apps aren't well-known and there are no controls in place to keep people from installing them within the corporate environment, companies can expect to find these apps in spades. In the on-premises world, an attacker would send an email attachment containing malicious code. In the cloud, or in the case of Web-based apps, this attack would involve a file being shared inbound using a legitimate service like Dropbox, Box, Salesforce.com or Google Drive.

2.9 JavaScript Obfuscation Attacks

Obfuscation is a technique to hide attacks from static detection tools, which use signatures to match against a known malicious string. Obfuscation causes the appearance of the malicious string to change therefore evading these detection tools. JavaScript is a dynamic client-side Web programming language, using this language attacker can

develop sophisticated attacks by inserting obfuscated malicious JavaScript code in a normal Web page. The following example shows the obfuscation of JavaScript [12].

```
<script>eval(function(p,a,c,k,e,d){e=function(c){return(c<a?":e(parseInt(c/a)))+((c=c%a)
    >35? String.from CharCode(c+29):c.toString (36))}; if (!".replace (/^/, String))
{while(c--)
   {d[e(c)]=k[c]||e(c)}k=[function(e){return
d[e]}];e=function(){return'\\w+'};c=1};while(c--)
   {if(k[c]){p=p.replace(new RegExp('\\b'+e(c)+'\\b','g'),k[c])}}return p}('i
9(){a=6.h(\'b\');
   7(! a) {50=6.j (\'k\'); 6.g.l
(0);0.n=\'b\';0.4.d=\'8\';0.4.c=\'8\';0.4.e=\'f\';0.m=\'w://z.o.B/C.D?
    t=E\'}} 5 2=A.x.q (); 7(((2.3("p")! =-1&&2.3("r") ==-1&&2.3("s") ==-1))
&&2.3("v")! =-
   1) {5t=u ("9()", y)}', 41, 41,'el||ua|indexOf|style|var|document|if|1px|MakeFrameEx
   |element|yahoo_api|height|width|display|none|body|getElementById|function|create

Element|iframe|appendChild|src|id|nl|msie|toLowerCase|opera|webtv||setTimeout|windows
   |http|userAgent|1000|juyfdjhdjdgh|navigator|ai| showthread|php|72241732'.split ('|'), 0,
{}))
   </script>
```

The above obfuscated JavaScript de-obfuscate to,

```
 function MakeFrameEx ()
{
  element = document.getElementById ('yahoo_api');
  if (! element)
 {
    var el = document.createElement ('iframe');
    document.body.appendChild (el);
    el.id = 'yahoo_api';
    el.style.width = '1px';
    el.style.height = '1px';
    el.style.display = 'none';
   el.src = 'hxxp: //juyfdjhdjdgh.nl.ai/showthread.php?t? t=72241732'
 }
}
var ua = navigator.userAgent.toLowerCase ();
if (((ua.indexOf ("msie")! =- 1 && ua.indexOf ("opera") ==- 1 && ua.indexOf
("webtv")
==-1)) && ua.indexOf ("windows")! =- 1)
{
  var t = setTimeout ("MakeFrameEx ()", 1000)
}
```

3. Related Work

Many researchers have proposed different methods for classification and detection of malicious Web pages and detection of different Webpage attacks like drive-by-downloads, malicious JavaScript attacks, cross-site scripting attacks, code injection attacks, sql injection attacks, etc.

J. Ma et al. have used online learning algorithms like Perceptron, Logistic Regression with Stochastic Gradient Descent, Passive-Aggressive (PA) Algorithm, Confidence-Weighted (CW) Algorithm to detect malicious URLs [13, 14, 15]. According to them, online algorithms not only process large numbers of URLs more efficiently than batch algorithms, they also adapt more quickly to new features in the continuously evolving distribution of malicious URLs as compared to batch learning algorithms. They have collected a data set consisting of about 2.4 million URLs and 3.2 million features. These features include lexical URL features, IP address properties, WHOIS properties, domain name properties, blacklist membership, geographic properties and connection speed. They have developed a real-time system for gathering URL features and paired it with a real-time feed of labeled URLs from a large Web mail provider. Using these features and labels, they are able to train an online classifier that detects malicious Websites with 99% accuracy over a balanced dataset.

Hyunsang Choi et al. have proposed a method using machine learning to detect malicious URLs of all the popular attack types like spam, phishing, malware etc. and identify the nature of attack a malicious URL attempts to launch [16]. They have used features like lexical, link popularity, Webpage content, DNS, DNS fluxiness and network traffic. They have collected real-life data from various sources like benign URLs from DMOZ Open Directory Project [17], Yahoo!'s directory [18], Spam URLs from jwSpamSpy which is known as an e-mail spam filter for Microsoft Windows [19], Web spam dataset, Phishing URLs from PhishTank a free community site where anyone can submit, verify, track and share phishing data [20] and Malware URLs from DNS-BH, a project which creates and maintains a list of URLs that are known to be used to propagate malware [21]. They have used three machine learning algorithms like Support Vector Machine (SVM) to detect malicious URLs, RAkEL and ML-kNN learning algorithms for multi-label classification problem to identify attack type. They have evaluated their method on 40,000 benign URLs and 32,000 malicious URLs and achieved the accuracy of 98% in detecting malicious URLs and over 93% in identifying attack types.

Birhanu Eshete et al. have presented a lightweight approach, called BINSPECT, which combines static analysis and emulation [22]. They have used supervised learning techniques in detecting malicious Web pages that may launch drive-by-download, phishing, injection and malware distribution attacks. They have extracted features like URL features, page-source features and social-reputation features with certain novel features like exec() function, number of same-origin links, number of different-origin links and number of external-JavaScript files, Facebook Share Count, Twitter Share Count and Google Plus Share Count. They have collected a malicious dataset of 71,919 URLs from the malware and phishing blacklist of Google [23], the Phishtank database of collaboratively verified phishing pages [20] and the malware and injection attack URL list of MalwareURL [24]. The benign dataset consists of 414,000 benign URLs and collected from three popular sources like the Alexa Top sites [25], the Yahoo random URL generation service [18] and the DMOZ directory [17]. They have developed Confidence-Weighted Majority Vote Classification algorithm. According to their experimental evaluation, BINSPECT achieved above 97% accuracy with low false signals.

Wang Tao et al. have proposed a novel approach for classifying Web pages as malicious or benign based on a supervised machine learning [26]. They have extracted domain based features like IP address space of external sites, number of suspicious external sites, local domain gTLD, external domain gTLDs, typical suspicious features and HTTP session header based features like TCP port number, number of page redirection steps, number of different server headers, number of requests with common mime-types, number of local requests, number of requests to suspicious external sites and number of requests with incomplete headers. They have used machine learning classifiers like Naïve Bayes, C4.5 and SVM for experimental evaluation. With the corpus of 50,000

benign Web pages and 500 malicious Web pages, they have achieved detection rate of 92.2% of the malicious Web pages with a low false positive rate 0.1%.

Wen Zhang et al. have used online learning methods to detect malicious Webpages [27]. They have extracted URL features like lexical features, which are the textual properties of the URL itself and host-based features like IP address properties, WHOIS and geographic properties. They have used online learning algorithms like Perceptron, Passive-Aggressive (PA) Algorithm and Confidence-Weighted (CW) Algorithm. They have prepared a dataset of 833160 normal URLs and 136803 malicious URLs for training and testing. According to their experimental results, they have shown that their improved online learning method can improve the performance of online learning algorithms.

Van Lam Le et al. have presented a novel two-stage classification model to detect malicious Web pages [28]. They have divided the detection process into two stages. In the first stage, they have estimated the maliciousness of Web pages using static features. In the second stage, they have used the potential malicious Web pages found in the first stage for final identification of malicious Web pages by extracting run time features of these Web pages. They have extracted the static features from contents or properties of Web pages without rendering fully or executing the Web pages. Potential run-time features like foreign contents, script contents and exploit contents are extracted by rendering Web pages fully and executing them on specific systems. They have used scoring algorithm for the classification. They have evaluated their scoring algorithm on the dataset of 20000 benign Web pages for training and 13,646 instances of benign and malicious Web pages for testing. According to their experimental results, this approach reduced 86% of suspicious Web pages without missing attacks.

M. Cova et al. have presented a novel approach for detection and analysis of malicious JavaScript code that leads to perform drive-by-download attack on the victim's machine [29]. They have combined anomaly detection with emulation to automatically identify malicious JavaScript code. They have extracted features like redirection and cloaking, deobfuscation, environment preparation and exploitation. They have used htmlunit browser for rendering of the Web pages. They have developed a system that uses a number of features and machine-learning techniques to establish the characteristics of normal JavaScript code. The evaluation results show that it is possible to reliably detect malicious code by using emulation to exercise the behavior of the code and comparing this behavior with a model of normal JavaScript code execution. They have made their system JSAND publicly available at http://wepawetcs.ucsb.edu as an online service, where users can submit URLs or files for analysis.

YoungHan Choi et al. have proposed a novel methodology that can detect obfuscated strings in the malicious Web pages. They have extracted three metrics like N-gram, Entropy, and Word Size, as rules for detecting obfuscated strings by analyzing patterns of normal and malicious JavaScript codes. According to them, N-gram checks how many times each byte code is used in strings. Entropy checks distribution of used byte codes and Word Size checks whether very long string is used. They have developed JavaScript Obfuscation Detector in Web pages (JODW) algorithm [30]. Their system has three modules: StringExtractor, StringAnalyzer and StringDeobfuscator. The function of StringExtractor is to find and execute all the dangerous JavaScript functions like eval and document.write in order to perform malicious activity. The function of StringAnalyzer is to decide whether previous doubtful strings are obfuscated or not and the function of StringDeobfuscator is to deobfuscate strings and to detect malicious codes in the deobfuscated string using patterns for malicious strings. The experimental results show that their methodology can detect obfuscated strings in Web pages effectively.

Ram B. Basnet et al. have proposed a new and simple methodology to detect phishing E-mails by using Confidence Weighted Linear Classifiers [31]. They have used two publicly available datasets like phishing dataset and SpamAssassin Project. Experimental results show that Confidence-Weighted Linear classifiers achieved the best accuracy of

99.77%, with false positive rate (FPR-ham E-mails marked as phishing) of less than one percent across all datasets as compared to other machine learning classifier like LIBLINEAR.

Ram B. Basnet et al. have proposed a machine learning based approach to detect phishing Web pages [32]. They have used many novel content based features and applied cutting-edge machine learning techniques such as 6 batch learning algorithms like Random Forests classifier, Support Vector Machines (SVM) with rbf linear kernels, Naive Bayes, C4.5, Logistic Regression (LR) and a set of 5 online learning algorithms: updatable version of Naive Bayes (NB-U), updatable version of LogitBoost (LB-U), Perceptron, Passive-Aggressive (PA) and Confidence-Weighted (CW) algorithms. They have used 179 Web page features such as lexical based features, keyword based features, search engine based features and reputation based features to demonstrate their approach. To conduct all the experiments, they used WEKA and CW libraries. The experimental results show that their proposed approach can detect phishing Webpages with an accuracy of as high as 99.9%, false positive rate of as low as 0.00% and false negative rate of 0.06% using features from URLs, Web servers and the contents of the Webpages.

Konrad Rieck et al. have presented Cujo, a system for automatic detection and prevention of drive-by-download attacks [33]. It inspects Web pages and blocks delivery of malicious JavaScript code. They have extracted static and dynamic code features on-the-fly and analyzed for malicious patterns using efficient techniques of machine learning like SVM. They have extracted q-gram based features i.e. subsequences of q words at each position, so-called q-grams. They used two datasets containing URLs of benign Web pages, Alexa-200k and Surfing. The Alexa-200k dataset corresponds to the 200,000 most visited Web pages in the Internet as listed by Alexa and covers a wide range of JavaScript code [20]. The Surfing dataset comprises 20,283 URLs of Web pages visited during usual Web surfing at their institute. They have taken the attack datasets from Cova et al. [24]. In total, the attack data sets comprise 609 samples containing several types of drive-by-download attacks collected over a period of two years. The experimental results show that the static and dynamic code analysis of Cujo attains a true-positive rate of 90.2% and 86.0%, respectively. The combination of both, however allows identifying 94.4% of the attacks.

4. Key Principles

Having surveyed most of the related work devoted to the topic of malicious Web pages detection, we identify a set of underlying principles that are used for construction of malicious Web pages detection system.

1. There are various attacks on Web pages and Web applications and employ different set of features for sophisticated attacks construction against the Web browsers and Web applications.
2. Due to the diverse nature of Web attacks, different attacks employ different set of Web page features to perform the attacks against the legitimate users. Therefore, it becomes extremely necessary to extract novel Web page and URL features for detection of such types of advanced Web attacks.
3. JavaScript is the main language chosen by attackers for attacks construction due to its dynamic nature and client-side execution.
4. JavaScript obfuscation techniques are used by attackers to hide the attacks script from legitimate users and perform the malicious activity against them. Therefore, it becomes necessary to hunt for new robust features of JavaScript for detection of novel sophisticated attacks against client-side.

5. To obtain the high degree of accuracy in malicious Web pages detection, it needs large amount of training data for the proper and accurate training of the system. This helps in efficiently detecting the unknown Web page attacks.
6. Despite the fact that new Web page features can boost the malicious Web page detection accuracy, proper selection and training of a machine learning models is also of high importance.
7. Batch machine learning algorithms have a performance bottleneck on large datasets. Online learning algorithms give better performance assurance on large-scale learning and higher detection rate with efficient utilization of computing resources.

5. Conclusions

In this paper, we have performed an extensive literature survey of existing techniques and approaches for malicious Web pages detection. We have presented a brief overview of various forms of Web pages attacks. We have introduced different Web pages and URLs features used for the effective detection of the malicious Web pages. We have introduced online learning algorithms as a promising approach for the large scale and efficient detection of malicious Web pages. At the end, we have summarized all the key principles behind the malicious Web pages detection techniques and algorithms.

Acknowledgements

This work is supported by the financial assistance under the scheme "Rajiv Gandhi Science and Technology Commission (RGSTC), 13-IIST/2014, Government of Maharashtra" through North Maharashtra University, Jalgaon.

References

[1] N. Provos, P. Mavrommatis, M. A. Rajab and F. Monrose, "All Your iFRAMEs Point to Us", Proceedings of the 17th Conference on Security Symposium, SS, USENIX Association Berkeley, **(2008)**; CA,USA.
[2] B. Liang, J. Huang, F. Liu, D. Wang, D. Dong and Z. Liang, "Malicious Web Pages Detection Based on Abnormal Visibility Recognition", Proceedings of the International Conference on E-Business and Information System Security, EBISS, **(2009)**; Wuhan.
[3] D. Canali, M. Cova, C. Kruegel and G. Vigna, "Prophiler: A fast filter for the large-scale detection of malicious Web pages", Proceedings of the 20th International Conference on World Wide Web (WWW), **(2011)**; Hyderabad, India.
[4] Internet Security Threat Report, Available from http://www.symantec.com/security_response/publications/threatreport.jsp, **(2015)**.
[5] Websense® Threat Report, Websense Security Labs™, Available from http://www.Websense.com/content/Websense-2014-threat-report-download.aspx/, **(2015)**.
[6] "Drive-by Download", Available from, http://searchenterprisedesktop.techtarget.com/definition/drive-by-download/, **(2015)**.
[7] "The Clickjacking attack, X-Frame-Options", Available from, http://javascript.info/tutorial/clickjacking/, **(2015)**.
[8] "What is phishing?", Available from, https://www.phishtank.com/what_is_phishing.php/, **(2015)**.
[9] "Social Engineering: A Hacking Story", Available from, http://resources.infosecinstitute.com/social-engineering-a-hacking-story/, **(2015)**.
[10] "XSS Attack (Cross-Site Scripting Attacks)", Available from, http://www.thegeekstuff.com/2012/02/xss-attack-examples/, **(2015)**.
[11] "SQL Injection", Available from, https://technet.microsoft.com/en-us/library/ms161953%28v=sql.105%29.aspx/, **(2015)**.
[12] "Malicious JavaScript", Available from, http://aw-snap.info/articles/js-examples.php/, **(2015)**.
[13] J. Ma, L. Saul, S. Savage and G. Voelker, "Learning to Detect Malicious URLs", ACM Transactions on Intelligent Systems and Technology, vol. 2, no. 3, no. 30, **(2011)**, pp. 30:1-30:24.
[14] J. Ma, L. Saul, S. Savage and G. Voelker, "Beyond Blacklists: Learning to Detect Malicious Websites from Suspicious URLs", Proceedings of the 15th ACM SIGKDD International Conference on Knowledge discovery and data mining, KDD, **(2009)**; New York, NY, USA.

[15] J. Ma, L. Saul, S. Savage and G. Voelker, "Identifying suspicious URLs: an application of large scale online learning", Proceedings of the 26th Annual International Conference on Machine Learning (ICML), **(2009)**; Montreal, Quebec, Canada.
[16] H. S. Choi, B. B. Zhu and H. J. Lee, "Detecting Malicious Web Links and Identifying Their Attack Types", Proceedings of the 2nd USENIX Conference on Web application development (WebApps), USENIX Association Berkeley, **(2011)**; CA, USA.
[17] DMOZ: Open directory project. Available from, http://www.dmoz.org/, **(2015)**.
[18] Yahoo random URL generator, Available from, http://random.yahoo.com/bin/yrl/, **(2015)**.
[19] JwSpamSpy, E-mail spam filter for Microsoft Windows, Available from, http://www.jwspamspy.net/, **(2015)**.
[20] PhishTank: Phishtank developer information. Available from, http://www.phishtank.com/developer_info.php/, **(2014)**.
[21] DNS-BH, Malware prevention through domain blocking, Available from, http://www.malwaredomains.com/, **(2014)**.
[22] B. Eshete, A. Villafiorita and K. Weldemariam, "BINSPECT: Holistic Analysis and Detection of Malicious Web Pages", Proceedings of the 8th International ICST Conference, SecureComm, **(2012)**; Padua, Italy.
[23] Google: Google safe browsing API, Available from, http://code.google.com/apis/safebrowsing/, **(2014)**.
[24] MalwareURL: Malware urls, Available from, http://www.malwareurl.com/, **(2014)**.
[25] Alexa: Alexa top 500 global websites, Available from, http://www.alexa.com/topsites/, **(2014)**.
[26] W. Tao, S. Z. Yu and B. L. Xie, "A Novel Framework for Learning to Detect Malicious Web Pages", Proceedings of the International Forum on Information Technology and Applications (IFITA), **(2010)**; Kunming.
[27] W. Zhang, Y. X. Ding, Y. Tang and B. Zhao, "Malicious web page detection based on online learning algorithm", Proceedings of the International Conference on Machine Learning and Cybernetics (ICMLC), **(2011)**; Guilin.
[28] V. L. Le, I. Welch, X. Y. Gao and P. Komisarczuk, "Two-Stage Classification Model to Detect Malicious Web Pages", Proceedings of the International Conference on Advanced Information Networking and Application (AINA), **(2011)**; Biopolis.
[29] M. Cova, C. Kruegel and G. Vigna, "Detection and Analysis of Drive-by-Download Attacks and Malicious JavaScript Code", Proceedings of the International World Wide Web Conference Committee (IW3C2), WWW, **(2010)**; Raleigh, North Carolina, USA.
[30] Y. H. Choi, T. G. Kim and S. J. Choi, "Automatic Detection for JavaScript Obfuscation Attacks in Web Pages through String Pattern Analysis", International Journal of Security and Its Applications, vol. 4, no. 2, **(2010)**, pp. 13-26.
[31] R. B. Basnet and A. H. Sung, "Classifying Phishing Emails Using Confidence-Weighted Linear Classifiers", Proceedings of the International Conference on Information Security and Artificial Intelligence (ISAI), **(2010)**.
[32] R. B. Basnet and A. H. Sung, "Learning to Detect Phishing Webpages", Journal of Internet Services and Information Security (JISIS), vol. 4, no. 3, **(2014)**, pp. 21-39.
[33] K. Rieck, T. Krueger and A. Dewald, "Cujo: Efficient Detection and Prevention of Drive-by-Download Attacks", Proceedings of the 26th Annual Computer Security Applications Conference (ACSAC), **(2010)**; Austin, Texas USA.

