**Detection of malicious web pages based on hybrid analysis**

**Rong Wang ∗, Yan Zhu , Jiefan Tan , Binbin Zhou**

1.

Introduction

Attacks on websites have become an increasing problem in re- cent years. According to the 2016 Internet Security Threat Report [1] , 78% of websites in 2015 had vulnerabilities that included de- fects, browser flaws and problematic plugins, all of which can be used by unscrupulous people to launch web attacks. Malicious web pages are critical elements in major web attacks. Such web pages are disguised as normal web pages, and they use malicious content to redirect visitors to other spam sites. Even worse, many malevo- lent pages can steal visitor’s private data or automatically down- load rogue programs to the visitors’ computers. Typically, mali- cious web pages are represented by phishing sites and web Tro- jans. Fig. 1 shows a screenshot of a malicious web page, which is a website that looks like PayPal.com, but, instead, it is a malicious website. At present, the two common methods of detecting malicious web pages are static analysis and dynamic analysis. Static analysis usually is combined with the technique of classification learning. It extracts static features from web pages to train a classifier that predicts whether the web pages are benign or malicious. Although static analysis features a simple design and is easy to use, it has the major problem of having excessive false negatives. This occurs because some behaviors of malicious web pages are triggered only in a specified execution environment. Dynamic analysis, however, puts web pages directly into a browser engine. By logging and an- alyzing the execution information, the method can provide extra details about any malicious behaviors. However, it requires more computing resources to emulate the execution environment.

Given that both of the analysis techniques have positive and negative features, a hybrid method is introduced that combines their positive features. Three different kinds of features of a web page, i.e., URL, HTML document and JavaScript in the source code, are analyzed in the first stage, and a classifier is used to filter out certain benign and malicious web pages. Unknown web pages are checked further using dynamic analysis based on an detec- tor, which depends mainly on monitoring some specific Applica- tion Program Interface (API) calls in the second stage. The rest of the paper is organized as follows: The related work is addressed in Section 2 , and the proposed detection method is described in Section 3 . The experiment results are analyzed in Sections 4 , and 5 presents our conclusions and discusses future work. ∗

**BINSPECT: Holistic Analysis and Detection**

**of Malicious Web Pages**

**Birhanu Eshete, Adolfo Villafiorita, and Komminist Weldemariam**

**Fondazione Bruno Kessler (FBK-IRST), Trento, Italy**

1 Introduction

The Web has become an indispensable global platform that glues together daily

communication, sharing, trading, collaboration, and service delivery. Web users

often store and manage critical information that attracts cybercriminals who misuse

the Web and the Internet to exploit vulnerabilities for illegitimate benefits.

Malicious web pages, that exploit vulnerabilities and launch attacks for just

one time visit, take an alarmingly significant share of web-based attacks in recent

years [1–4]. When an innocent victim visits a web page, an attacker might have

compromised the page under visit (or crafted it purposefully) and the outcome

of the visit could be stealing of critical credentials (e.g., credit card details) to

impersonate the victim, installation of a malware binary on the victim’s machine

for future attacks, or even a complete takeover of the victim’s system to remotely

command and control it as a member of botnet [5–7]. In recent years, not only

is the prevalence of malicious web pages on the rise but also the way in which

attackers trick victims to malicious web pages is also getting sophisticated [2].

It has become a daily encounter to get contaminated search results from search

A.D. Keromytis and R. Di Pietro (Eds.): SecureComm 2012, LNICST 106, pp. 149–166, 2013.

c

Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2013

150 B. Eshete, A. Villafiorita, and K. Weldemariam

engines on trendy terms, malicious links shared on social media, and legitimate

web pages injected with malicious scripts [3].

The thus-far proposed defenses against malicious web pages fall into two major

blocks, i.e., static analysis and dynamic analysis techniques. However, the use

of blacklists is still a common way to facilitate and enrich these techniques by

making use of heuristics and learning techniques.

Static analysis techniques, [5, 8–16], inspect web page artifacts without rendering

the page in a browser. The inspection usually involves quick extraction

of discriminative features from the URL string, host identity, HTML, and

JavaScript code. The feature values are then encoded to train machine learning

techniques to build classifiers based on which unknown web pages are classified.

The major assumption in static analysis is that the statistical distribution of features

in malicious URLs (e.g., spam URLs, phishing pages) tend to differ from

that of benign.

In static analysis, it is difficult to detect attacks that require rendering of a

page to take action. More precisely, when using page source there is a high risk

of obfuscated content (e.g., JavaScript) and overlooking of malicious JavaScript

that exploits vulnerabilities of browser plugins. In addition, host details of fresh

(benign) URLs, registered by registrars with low reputation, are likely to be

misclassified as malicious due to their low reputation scores. In effect, there is a

high risk of false positives. On the other hand, false negatives may arise as wellreputed

registrars may host malicious web pages which have escaped the static

analysis effort. Other sources of false negatives are web pages that use free hosting

services or already compromised sites with benign-looking URLs and host

details. For static anlaysis relying on lexical URL features, an attentive attacker

may evade these features to mislead detection techniques by carefully crafting

malicious URLs which look statistically indistinguishable from the benign ones.

Dynamic analysis approaches, [11, 17–25], inspect the execution dynamics

when a page is executed. Such techniques could be deployed at a proxy-level

(e.g., [20]) to intercept requests (responses), visit the URL in a controlled environment

(e.g., disposable virtual machine), analyze its execution dynamics for

hints of malicious activity (e.g., unusual process creation, repeated redirection),

and decide if it is safe to render the page in the browser. Alternatively, clientside

sandboxing of critical page content (e.g., JavaScript) could be used (as in

[18]) to log critical actions (e.g., invoking a plugin) and match logs with known

patterns of malicious activities or apply learning-based techniques to model and

classify malicious intentions.

While effective at uncovering daunting malicious web pages, dynamic analysis

approaches are resource intensive as they need to load and execute the page

under analysis and modern web pages are usually stuffed with rich client-side

code and content which take longer analysis time. Moreover, not all web pages

are likely to launch attacks when visited. There are web pages which require user

interaction or wait for certain conditions to take action.

Blacklisting-based techniques maintain a list of known malicious URLs, IP

addresses, and domain names collected by manual reporting, honeyclients, and

BINSPECT: Holistic Analysis and Detection of Malicious Web Pages 151

custom analysis techniques. For example, Google Safe Browsing service

[26] maintains a blacklist against which it checks URL requests from browsers

to alert users if the requested URL happens to be in the blacklist. Another tool

powered by blacklisting is McAfee Site Advisor [27] which is pluggable to

Mozilla Firefox and Internet Explorer to rate safety of web pages and

search engine results prior to rendering the page in the browser.

Although lightweight to deploy and easy to use, blacklisting is effective only if

one can exhaustively patrol the Web to identify malicious web pages and timely

update the blacklist. In practice, to do so is infeasible due to: fresh web pages are

too new to be blacklisted even if they are malicious right from the outset, some

web pages may escape from the blacklisting due to ‘cloaking’, and attackers may

frequently change where the malicious web pages are hosted. Consequently, the

URLs and IP addresses may also change accordingly [5], [17].

Heuristic-based techniques (e.g., [15]) build signatures of known attack payloads

to be used by antiviral systems or intrusion detection systems to scan a

web page and flag it as malicious if its heuristic pattern matches signatures in

the database. Unfortunately, such signatures are easily bypassed by attackers

(mainly through obfuscation) and the heuristics fail to detect novel attacks. In

addition, the rate at which the signature database of heuristic-based systems is

updated is way slower than the pace at which attackers overwhelm victims with

novel attacks, resulting in zero-day exploits.

In addition to the afore-mentioned limitations, most approaches focus on one

prominent attack while attack techniques are getting more and more complex

whereby attackers use blended attack techniques by combining existing attack

techniques to evade existing countermeasures. More importantly, applying static

or dynamic analysis approaches in a complementary fashion is limited to capturing

partial snapshot of a malicious web page.

To this end, the ideal solution is to leverage static and dynamic analysis

to capture a comprehensive snapshot of a malicious web page and ensure that

the overhead cost of analyzing a web page is optimal. This can be achieved by

holistically characterizing and then analyzing, and detecting malicious web pages

to capture a comprehensive snapshot of malicious web pages while ensuring that

the analysis and detection remains lightweight in terms of its responsiveness and

resource consumption.

In this paper, we present the design, implementation, and experimental evaluation

of a holistic and lightweight system, called BINSPECT, that leverages

a combination of static analysis and minimalistic emulation to apply supervised

learning techniques in detecting malicious web pages pertinent to drive-bydownload,

phishing, injection, and malware distribution. BINSPECT achieved

detection accuracy above 97% with low false signals and an average performance

overhead of at most 5 seconds.

The contributions of this paper are the following:

– we developed a holistic approach to analyze and detect malicious web pages

by leveraging static analysis and lightweight emulation of web page rendering

with low performance overhead.

152 B. Eshete, A. Villafiorita, and K. Weldemariam

– we introduced novel features and enhanced existing ones so as to improve

their discriminative power in the characterization of malicious and benign

web pages.

– we designed, implemented, and evaluated our approach over a large dataset

of malicious and benign web pages and demonstrated that our approach is

effective in practice.

The paper is structured as follows. In Section 2, we present a real motivational

example pertinent to malicious web pages. Section 3 covers details of holistic

characterization of malicious web pages focusing on features we introduce as

new and enhance from existing ones. In Section 4, a high-level description of our

approach is presented. Details of the experimental setup and evaluation of our

approach are discussed in Section 5. Section 6 positions our approach relative to

prior work. Finally, Section 7 concludes the paper.

**A NOVEL APPROACH FOR ANALYZING AND CLASSIFYING**

**MALICIOUS WEB PAGES**

INTRODUCTION

1.1 Overview

The browser application retrieves code, usually written in HTML (HyperText Markup

Language) and other computer languages, from a web server. Then, it interprets this code

and displays it as a web page for you to view. In most cases, user interaction is needed to

tell the browser what website or speci c web page you want to see. Using the browser's

address bar is one way to do this. Web browsers come in many di erent

avors, each with

its nuances. All the best-known ones are free, and each has its own particular set of

options governing privacy, security, interface, shortcuts, and other variables. A web

browser is an essential tool many people browse the internet at work to research a certain

topic, make transactions, use social media, and much more. To enhance the website

experience, programmers add features. However, adding a feature to a website decreases

the browsing security for the visitor by potentially adding a security hole a hacker can

enter. Such cyber criminals focus on looking for vulnerabilities on the most commonly

used site along with browsing features. In some other browsers, hackers can track online

activities, take over the PCs, destroy important information and steal the data.

Nevertheless, there are numerous third-party extensions and tools so that the browser can

become powerful to provide user protection but still. There is room for hackers to get the

bene t and invade essential user information.

Taking account of the sheer quantity of users of the Web and the lack of knowledge of

users, one simple strategy used by cyber-criminals is to lure unsuspecting victims into

visiting a web page that is either deliberately crafted to launch attacks or a vulnerable

web page that is under the control of the cyber-criminal. Most attacks on the Web

1

happen when victims visit malicious websites. They attract them to give away sensitive

information (e.g., phishing sites) or utilize vulnerabilities in the web browser, its

extensions, and plugins (e.g., drive-by-downloads, malicious advertisements) to drop a

malware binary on the victim's computer. To maximize the gain rate of malicious

activities, cyber-criminals employ several tra c attractions such as spam email, black-hat

Search Engine Optimization (SEO), pay-per-install services, web blogs, and social

networking websites.

In the past, composing such malicious exercises required a skilled attacker to craft

malicious web pages. The motivation was frequently fame and curiosity. Now, the

motivation following cyber-crime is largely unlawful monetary gain. Unsophisticated

attackers can acquire attack toolkits, called Exploit Kits, to craft malicious web pages

from which they can scale attacks. The Exploit Kits are produced and marketed by expert

attackers in the covered market. Similar to legitimate software, Exploit Kits are

systematically upgraded with di erent attack payloads and avoidance techniques to

stimulate exposure mechanisms.

There are many solutions to this problem like using and customized extension to monitor

the web page behavior and alert the users, other browsers like Brave and Tor protect users

by giving an option to enable or disable user tracking, both have disadvantages.

Extensions are prede ned and loaded to the browser they cannot detect new

vulnerabilities or avoid them, same with the said browser monitoring new API calls and

analyzing its behavior is di cult. Many extensions, antivirus, intrusion detection systems

also other malware detection applications have been developed to prevent damages caused

by these malicious programs. However, there are still many issues that need attention.

The main reason is the changing nature of malware and defects in existing solutions.

2

Several techniques from various methods have been introduced for malicious detection.

For our work, we have categorized into two divisions as static feature-based techniques

and dynamic feature-based techniques. Static techniques analyze the malicious activity by

pre-de ne features without and dynamic updates on a feature, this a ects the control

ow

and we might get results without executing.

The major shortcoming is to detect a novel malware until the new features are updated.

Dynamic approach analysis the malicious samples during execution this technique also

analyzes the behavior of malware samples. Malicious programmers are developing more

complex and advanced malware using obfuscation and encryption techniques dynamic

techniques have bene ted over static techniques because it1 is more di cult to mask the

behavior of malware during its execution. In recent years, many researchers employed

machine learning (ML) techniques to tackle the constantly changing behavior of malware

detection dynamically. The ML technique stakes a labeled dataset as a training dataset

and develops a model representing the behavior of malware and benign samples. The

trained model is found capable of classifying test samples.

1.2 Thesis Motivation and Hypothesis

Increasing online and web malicious activities is a challenging task for web browsers,

extensions, and third-party to develop analysis techniques that use static analysis which

understands web attacks. Few techniques are e cient, they often have the drawback of

low accuracy, This is where machine learning shines because it can recognize patterns and

predict threats in massive data sets, all at machine speed. By automating the analysis,

the browser can rapidly detect threats and isolate situations that need to stop executing.

We wanted to accomplish automated detection and attribution of malicious web pages

3

that are inspired by the need for e cient techniques to complement dynamic web

malicious page detection and analysis. Machine learning can predict \attacks" online to

help prevent people from connecting to malicious websites. Machine learning analyzes

Internet activity to automatically identify attack infrastructures staged for current and

emergent threats. Algorithms can detect never-before-seen malware that is trying to run

on endpoints. It identi es new malicious  les and activity based on the attributes and

behaviors of known malware.

1.3 Contributions

Our contribution to thesis work include:

• We introduce a novel static analysis method to classify the malicious, benign code

from JavaScript, JQuery, and external JS(third party JS) in the form of abstract

syntactic structure. Also, we count the Malicious and benign features appeared in

respective  le from our tool.

• We introduce a novel dynamic analysis method by developing a browser extension

and monitoring the web pages in DOM. We classifying and analyze based on the

malicious and benign web pages.

• We have performed di erent machine learning algorithms on our static and dynamic

analysis.

• We calibrate our two novel analyses by performing machine learning algorithms and

verifying our results.

4

1.4 Thesis Structure

The rest of the thesis is organized as follows. Chapter 2, exhibits the background of

malicious web pages and di erent kinds of malicious attacks, we have also demonstrated

the types of attack how the attacker trick victims. In Chapter 3, we propose the

state-of-art of malicious web detection. We classi ed that into 2 categories, static analysis,

and dynamic analysis. Also illustrated the capabilities of each research and their

boundaries which motivated me to develop a comprehensive feature extraction tool

targeting HTML, JavaScript, and JQuery. In Chapter 4, we present our methodology with

two distinct approaches static and dynamic analysis. These give detailed extraction

methods of our technique. Chapter 5, explains more regarding features and classifications.

Chapter 6, we have reviewed the machine learning technique implemented furthermore

evaluated that technique, and presented the accuracy for each technique we implemented.

**JStrong: Malicious JavaScript detection based on code semantic representation and graph neural network**

**Yong Fang, Chaoyi Huang, Minchuan Zeng, Zhiying Zhao, Cheng Huang ∗**

1.

Introduction

With the improvement and enrichment of network technology, more and more websites begin to provide services in the form of web applications, which lead to multiple growths of web-based ap- plications ( Eshkevari et al., 2017 ). As a language with complete functions, JavaScript is widely used in the front-end development of Web applications ( Song et al., 2018 ). The javaScript language is cross-platform, can be embedded remotely, and can be dynami- cally executed ( Zhou and Evans, 2015 ). Although it brings many conveniences and wonderful interactive experiences to users, it also brings many threats and risks to web user terminals ( Brown et al., 2017; Hedin and Sabelfeld, 2015 ). Therefore, JavaScript ma- licious code has become the main way of a network attack. At- tackers can implement malicious behaviors by injecting malicious JavaScript code into web pages, such as spreading the Trojan virus, obtaining sensitive information from users, and mining data. The detection of JavaScript malicious code is usually realized by the detection of JavaScript scripts in Web applications. To avoid detection and defense, increasing confusing forms appear in JavaScript malicious code, which makes the code more concealed and less detectable. Meanwhile, existing detection methods mainly rely on feature matching or code text word embedding. The analy- sis method based on feature matching can only detect known ma- licious samples. Despite the fact that the detection speed of this method is fast, there are still false positives. Feature learning based on word embedding of code text cannot represent the actual con- text information of code. If only the syntax unit information of the code is extracted, the semantic code information will be lost, which means that the problem of comprehensiveness and ambi- guity in the code has not been resolved, so the detection effect is not good. To date, few studies have investigated the associa- tion between malicious JavaScript detection and code semantics. In 2019, Fass et al. (2019) proposed an abstract code representa- tion method for JavaScript, which expanded the existing AST-based code representation capability by using data flow and control flow information. They extracted N-Gram features and node value fea- tures from the code representation and used the random forest model for training. This method effectively enhances the compre- hensive representation of code, but it only trains the text fea- tures extracted from graph data, and cannot learn the complete information from graph structure. The existing research ( Zhou et al., 2020, 2019 ) shows that graph neural network has a good learning effect on graph data and abstract code representation. Catal et al. (2021) proposed a Graph Attention Network (GAN)- based framework to detect malware attacks addressing ITS, the re- sults show that graph neural network is powerful in the malware detection field.

Aiming at the poor detection effect caused by the inability to fully capture the code information in the existing methods, we propose a malicious JavaScript code detection model JStrong based on graph neural network to comprehensively learn the syntax and semantic features of the code and improve the detection accuracy. The main contributions of this paper are as follows: • We propose a new method to detect malicious JavaScript that embeds both grammar and semantic features information of code into a comprehensive code representation graph, and then learn the features of the whole graph by the improved graph neural network to achieve the purpose of classification. To en- able the removal of redundant content in JavaScript code, we propose a new graph pruning algorithm to extract and retain complicated high-level semantic features including data- and control-flows of a program. • We study the performance of various code graph representation methods in JavaScript and the influence of these representation methods on graph neural network classifier. Then we comprehensively discuss the causes of these situations. • We present a novel model named JStrong to detect malicious JavaScript instances. Evaluation results show that JStrong out- performs the state-of-the-art method for detecting JS malware. JStrong achieves a better prediction of 99.95% accuracy under extensive comparison experiments. JStrong also detected malicious JavaScript code in several real websites with not low popularity.

**Detection and Analysis of Drive-by-Download Attacks**

**and Malicious JavaScript Code**

1. INTRODUCTION

Malicious web content has become the primary instrument used

by miscreants to perform their attacks on the Internet. In particular,

attacks that target web clients, as opposed to infrastructure components,

have become pervasive [28].

Drive-by downloads are a particularly common and insidious

form of such attacks [29]. In a drive-by download, a victim is lured

to a malicious web page. The page contains code, typically written

in the JavaScript language, that exploits vulnerabilities in the user’s browser or in the browser’s plugins. If successful, the exploit

downloads malware on the victim machine, which, as a consequence,

often becomes a member of a botnet.

Several factors have contributed to making drive-by-download

attacks very effective. First, vulnerabilities in web clients are widespread

(in 2008, such vulnerabilities constituted almost 15% of the

reports in the CVE repository [18]), and vulnerable web clients

are commonly used (about 45% of Internet users use an outdated

browser [8]). Second, attack techniques to reliably exploit web

client vulnerabilities are well-documented [4, 33–35]. Third, sophisticated

tools for automating the process of fingerprinting the

user’s browser, obfuscating the exploit code, and delivering it to the

victim, are easily obtainable (e.g., NeoSploit, and LuckySploit [15]).

The mix of widespread, vulnerable targets and effective attack

mechanisms has made drive-by downloads the technique of choice

to compromise large numbers of end-user machines. In 2007, Provos

et al. [28] found more than three million URLs that launched

drive-by-download attacks. Even more troubling, malicious URLs

are found both on rogue web sites, that are set up explicitly for

the purpose of attacking unsuspecting users, and on legitimate web

sites, that have been compromised or modified to serve the malicious

content (high-profile examples include the Department of

Homeland Security and the BusinessWeek news outlet [10, 11]).

A number of approaches have been proposed to detect malicious

web pages. Traditional anti-virus tools use static signatures

to match patterns that are commonly found in malicious scripts [2].

Unfortunately, the effectiveness of syntactic signatures is thwarted

by the use of sophisticated obfuscation techniques that often hide

the exploit code contained in malicious pages. Another approach

is based on low-interaction honeyclients, which simulate a regular

browser and rely on specifications to match the behavior, rather

than the syntactic features, of malicious scripts (for example, invoking

a method of an ActiveX control vulnerable to buffer overflows

with a parameter longer than a certain length) [14, 23]. A problem

with low-interaction honeyclients is that they are limited by the

coverage of their specification database; that is, attacks for which a

specification is not available cannot be detected. Finally, the stateof-

the-art in malicious JavaScript detection is represented by highinteraction

honeyclients. These tools consist of full-featured web

browsers typically running in a virtual machine. They work by

monitoring all modifications to the system environment, such as

files created or deleted, and processes launched [21, 28, 37, 39]. If

any unexpected modification occurs, this is considered as the manifestation

of an attack, and the corresponding page is flagged as

malicious. Unfortunately, also high-interaction honeyclients have

limitations. In particular, an attack can be detected only if the vulnerable

component (e.g., an ActiveX control or a browser plugin)

targeted by the exploit is installed and correctly activated on the detection system. Since there exist potentially hundreds of such vulnerable

components, working under specific combinations of operating

system and browser versions, the setup of a high-interaction

honeyclient and its configuration is difficult and at risk of being incomplete.

As a consequence, a significant fraction of attacks may

go undetected. (Indeed, Seifert, the lead developer of a popular

high-interaction honeyclient, says, “high-interaction client honeypots

have a tendency to fail at identifying malicious web pages,

producing false negatives that are rooted in the detection mechanism”

[32].)

In this paper, we propose a novel approach to the automatic detection

and analysis of malicious web pages. For this, we visit web

pages with an instrumented browser and record events that occur

during the interpretation of HTML elements and the execution of

JavaScript code. For each event (e.g., the instantiation of an ActiveX

control via JavaScript code or the retrieval of an external resource

via an iframe tag), we extract one or more features whose

values are evaluated using anomaly detection techniques. Anomalous

features allow us to identify malicious content even in the case

of previously-unseen attacks. Our features are comprehensive and

model many properties that capture intrinsic characteristics of attacks.

Moreover, our system provides additional details about the

attack. For example, it identifies the exploits that are used and the

unobfuscated version of the code, which are helpful to explain how

the attack was executed and for performing additional analysis.

We implemented our approach in a tool called JSAND (JavaScript

Anomaly-based aNalysis and Detection), and validated it on over

140,000 web pages. In our experiments, we found that our tool

performed significantly better than existing approaches, detecting

more attacks and raising a low number of false positives. We also

made JSAND available as part of an online service called Wepawet

(at http://wepawet.cs.ucsb.edu), where users can submit

URLs and files that are automatically analyzed, delivering detailed

reports about the type of observed attacks and the targeted

vulnerabilities. This service has been operative since November

2008 and analyzes about 1,000 URLs per day submitted from users

across the world.

In summary, our main contributions include:

• A novel approach that has the ability to detect previouslyunseen

drive-by downloads by using machine learning and

anomaly detection.

• The identification of a set of ten features that characterize

intrinsic events of a drive-by download and allow our system

to robustly identify web pages containing malicious code.

• An analysis technique that automatically produces the deobfuscated

version of malicious JavaScript code, characterizes

the exploits contained in the code, and generates exploit signatures

for signature-based tools.

• An online service that offers public access to our tool.

**Malicious URL Detection based on Machine Learning**

**Cho Do Xuan1, Hoa Dinh Nguyen1, Tisenko Victor Nikolaevich3**

I. INTRODUCTION

Uniform Resource Locator (URL) is used to refer to resources on the Internet. In [1], Sahoo et al. presented about the characteristics and two basic components of the URL as: protocol identifier, which indicates what protocol to use, and resource name, which specifies the IP address or the domain name where the resource is located. It can be seen that each URL has a specific structure and format. Attackers often try to change one or more components of the URL's structure to deceive users for spreading their malicious URL. Malicious URLs are known as links that adversely affect users. These URLs will redirect users to resources or pages on which attackers can execute codes on users' computers, redirect users to unwanted sites, malicious website, or other phishing site, or malware download. Malicious URLs can also be hidden in download links that are deemed safe and can spread quickly through file and message sharing in shared networks. Some attack techniques that use malicious URLs include [2, 3, 4]: Drive-by Download, Phishing and Social Engineering, and Spam.

According to statistics presented in [5], in 2019, the attacks using spreading malicious URL technique are ranked first among the 10 most common attack techniques. Especially, according to this statistic, the three main URL spreading techniques, which are malicious URLs, botnet URLs, and phishing URLs, increase in number of attacks as well as danger level.

From the statistics of the increase in the number of malicious URL distributions over the consecutive years, it is clear that there is a need to study and apply techniques or methods to detect and prevent these malicious URLs.

Regarding the problem of detecting malicious URLs, there are two main trends at present as malicious URL detection based on signs or sets of rules, and malicious URL detection based on behavior analysis techniques [1, 2]. The method of detecting malicious URLs based on a set of markers or rules can quickly and accurately detect malicious URLs. However, this method is not capable of detecting new malicious URLs that are not in the set of predefined signs or rules. The method of detecting malicious URLs based on behavior analysis techniques adopt machine learning or deep learning algorithms to classify URLs based on their behaviors. In this paper, machine learning algorithms are utilized to classify URLs based on their attributes. The paper also includes a new URL attribute extraction method.

In our research, machine learning algorithms are used to classify URLs based on the features and behaviors of URLs. The features are extracted from static and dynamic behaviors of URLs and are new to the literature. Those newly proposed features are the main contribution of the research. Machine learning algorithms are a part of the whole malicious URL detection system. Two supervised machine learning algorithms are used, SVM and random forest.

**A Novel Approach for Malicious URL Detection Based on the**

**Joint Model**

**JianTing Yuan ,1 YiPeng Liu ,2 and Long Yu 3**

With the continuous improvement of the network environment,

Internet applications have penetrated deeply into

all aspects of life. Simultaneously, the vast Internet applications

group also attracted many network attacks to make a

profit through malware, spam, and phishing websites.

According to the Check Point’s report in 2020 [1], more than

100,000 malicious websites are used to steal users’ personal

information or cause damage to users’ systems every day

around the world. Kaspersky’s report [2] stated that the

number of malicious URLs identified by web antivirus

components in 2020 was 173 million. Besides, the report also

mentioned that malicious URLs accounted for 66.07% of the

20 most active malicious programs. With the emergence of

more and more malicious websites, more and more individuals

and companies will suffer immeasurable losses

worldwide. ,e web page represented by the malicious URL contains

malicious interactive code, such as HTML tags [3], Java-

Script (JS) [4], and Cascading Style Sheets (CSS) [5]. ,e

attacker writes the source code containing malicious JS tags

into the website, and thereby, the malicious code is executed

while the user is visiting the website. For example, a remote

download program is executed in the background when a

user clicks on an advertisement implanted with malicious

code by a hacker. ,e user terminal is finally controlled to

collect user personal information. In addition, phishing

websites are also the main battlefield for malicious URLs.

,e attacker establishes an illegal site and leads users into

malicious web pages through inducements and other means

to complete malicious acts such as network fraud. To dispel

the user’s precautionary psychology, the attacker will construct

these websites very similar to the legitimate website,

indistinguishable by the human eye. Accelerating the development of malicious URL detection has become an essential task of network security in such a network

environment.

So far, predecessors have proposed a lot of malicious

URL detection methods. In previous research on malicious

website detection, researchers usually manually extract one

or more of the following features: web content feature

HTML, JavaScript code, host information feature WHOIS,

lightweight feature web URL, and visualization features, and

then input them into the machine learning or heuristic

learning system to detect malicious websites. For example,

Kumar et al. [6] used an HTML parser and JavaScript

simulator to extract web content features and input them

into a heuristic system. Chu et al. [7] used domain-related

information as the main feature and used machine learning

for detection research. However, the feature engineering of

machine learning technology is more cumbersome and relies

on the subjective judgment of researchers. ,e emergence of

deep learning has solved this well. Ren et al. [8] extracted the

word embedding of the URL character to identify malicious

URLs effectively. Peng et al. [9] added texture fingerprint

features based on extracting URL and host information and

then used a deep learning model for detection research. ,is

study only focuses on URL features and uses deep learning

techniques to detect and research malicious websites.

Designers generally design URLs as meaningful words to

facilitate memory, and some meaningless words usually

convey information in their character sequence. ,erefore, we

use word embedding and character embedding technology to

extract the semantic features of URLs. Since the URLs generated

by the same tool or organization have similar structures,

we also extracted the URL texture fingerprint features

(Section 3). ,erefore, a joint neural network algorithm

model was proposed to capture URL features. First, the attention

mechanism is used to give higher weight to key

features. Second, we used an improved independently recurrent

neural network (IndRNN) [10] called the bidirectional

IndRNN (Bi-IndRNN) model to encode the fusion

feature information. Finally, the CapsNet is to extract highlevel

semantic features. ,rough experiments, it is found that

the stacked CapsNet has made significant progress, and the

joint model is a precious exploration. ,e innovations of this

study are summarized as follows:

(1) We constructed a joint neural network algorithm

model that combines the attention mechanism, Bi-

IndRNN, and CapsNet for malicious URL detection

(2) To obtain more specific and natural features, we have

integrated different malicious URL feature information

to extract combined semantic and image

information

(3) A series of comparative experiments show that the

joint model proposed in this study achieves better

performance than some state-of-the-art methods

,e subsequent chapters of this study are organized as

follows. Section 2 introduces the contributions of previous

researchers to malicious URLs, Section 3 introduces the details

of the proposed method, Section 4 explains the experimental

results and analysis, and Section 5 summarizes this study.

**Malicious website identification using design attribute learning**

1 Introduction

Malicious websites form a major cyberattack vector [1].

Detecting malicious websites is a challenging task, as malicious

websites come in different formats and are often

bundled with useful content, such as software, that is downloaded

by naive users.

Traditional detection techniques rely on domain expertise

and leverage advanced industry knowledge to detect

indications of malicious activity. This approach results in

a constant need for the research and development of detection

capabilities, such as signatures and tailor-made features,

to detect malicious activity. Among these techniques, we

can find Document Object Model (DOM) analysis methods

[2–4], JavaScript scanning techniques [5, 6], analyses of the

software properties linked to websites [7–10], URL analysis

approaches [11–14], user navigation path analysis methods

[15, 16], text mining [17–19], and process mining strategies

[20].

B Or Naim

ornaim@mail.tau.ac.il

1 Tel Aviv University, Tel Aviv, Israel

One problematic effect related to the domain expertise

approach is the unending arms race that it creates. The tailormade

features designed as part of the detection method will

eventually be bypassed by the attacker, resulting in a need to

create new tailor-made features. In addition, this approach is

mainly relevant for detecting known threats and attack vectors;

as a result, it seems to be far less effective for detecting

emerging threats and zero-day attacks.

Another problematic effect of the traditional detection

techniques is the symmetry they create; the attacker can

essentially have access to the same data that the defender

uses to train their detection model, reverse engineer it, and

bypass the model.

As previously suggested, malicious websites need to balance

two opposing requirements to successfully function:

escaping detection tools while attracting visitors [21, 22]. To

attract visitors, a website needs to signal its claimed functionality

to potential users by leveraging appearance, content, and

experience [23]. This fundamental conflict can be exploited

to create a robust and sustainable classification approach. As

suggested in a previous work by Cohen et al. [24], websites

can be accurately classified and categorized by their design

attributes.

In this paper, we propose a framework for detecting malicious

websites by extensively learning their design attributes.

123

O. Naim et al.

The suggested approach was tested on a large-scale imbalanced

dataset that included a total of 35,707 website records,

697 of which were malicious. This dataset was assembled

to accurately represent the commercial, real-life scenario of

malicious website identification. Much attention was given

to properly representing the malicious website ratio out of

the entire population [25] and to ensuring that the malicious

websiteswere generated from the same initial list and ranking

system as the legitimate website population [26].

Validating the suggested approach on a real-life largescale

dataset poses some key challenges. For instance, the

noise and variance are much greater than those of a carefully

selected dataset. In addition, an extremely imbalanced

dataset requires proper measures when analyzing the data

and training the relevant models.

The suggested approach can effectively detect more than

83% of malicious websites while maintaining a low falsepositive

rate (FPR) of 2%. In addition, itwas proved effective

in detecting malicious and suspicious websites that allegedly

slipped under the radar in previous studies. The suggested

framework also offers explainability and can leverage the

cybersecurity practitioner’s experience and feedback to perform

better and respond to emerging threats.

Another part of our contribution lies in assembling and

sharing this unique and high-quality dataset, consisting of

multiple design attribute features and third-party enrichment,

with the research community.

**Detecting Malicious Web Links and Identifying Their Attack Types**

1 Introduction

While the World Wide Web has become a killer application on the Internet, it has also brought in an immense

risk of cyber attacks. Adversaries have used the Web as

a vehicle to deliver malicious attacks such as phishing,

spamming, and malware infection. For example, phishing typically involves sending an email seemingly from

a trustworthy source to trick people to click a URL (Uniform Resource Locator) contained in the email that links

to a counterfeit webpage.

To address Web-based attacks, a great effort has been

directed towards detection of malicious URLs. A common countermeasure is to use a blacklist of malicious

URLs, which can be constructed from various sources,

This work was done when Hyunsang Choi was an intern at Microsoft Research Asia. Contact author: Bin B. Zhu (binzhu@ieee.org).

particularly human feedbacks that are highly accurate yet

time-consuming. Blacklisting incurs no false positives,

yet is effective only for known malicious URLs. It cannot detect unknown malicious URLs. The very nature of

exact match in blacklisting renders it easy to be evaded.

This weakness of blacklisting has been addressed by

anomaly-based detection methods designed to detect unknown malicious URLs. In these methods, a classification model based on discriminative rules or features is

built with either knowledge a priori or through machine

learning. Selection of discriminative rules or features

plays a critical role for the performance of a detector.

A main research effort in malicious URL detection has

focused on selecting highly effective discriminative features. Existing methods were designed to detect malicious URLs of a single attack type, such as spamming,

phishing, or malware.

In this paper, we propose a method using machine

learning to detect malicious URLs of all the popular attack types including phishing, spamming and malware

infection, and identify the attack types malicious URLs

attempt to launch. We have adopted a large set of discriminative features related to textual patterns, link structures, content composition, DNS information, and network traffic. Many of these features are novel and highly

effective. As described later in our experimental studies, link popularity and certain lexical and DNS features

are highly discriminative in not only detecting malicious

URLs but also identifying attack types. In addition, our

method is robust against known evasion techniques such

as redirection [42], link manipulation [16], and fast-flux

hosting [17].

Identification of attack types is useful since the knowledge of the nature of a potential threat allows us to

take a proper reaction as well as a pertinent and effective countermeasure against the threat. For example,

we may conveniently ignore spamming but should respond immediately to malware infection. Our experiments on 40,000 benign URLs and 32,000 malicious

URLs obtained from real-life Internet sources show that

our method has achieved an accuracy rate of more than

98% in detecting malicious URLs and an accuracy rate of more than 93% in identifying attack types.

This paper has the following major contributions:

• We propose several groups of novel, highly discriminative features that enable our method to deliver

a superior performance and capability on both detection and threat-type identification of malicious

URLs of main attack types including spamming,

phishing, and malware infection. Our method provides a much larger coverage than existing methods

while maintaining a high accuracy.

• To the best of our knowledge, this is the first study

on classifying multiple types of malicious URLs,

known as a multi-label classification problem, in a

systematic way. Multi-label classification is much

harder than binary detection of malicious URLs

since multi-label learning has to deal with the ambiguity that an entity may belong to several classes.

**Context-sensitive and keyword density based supervised machine learning.**

**techniques for malicious webpage detection**

1. Introduction

The Internet usage has become essential for our common daily activities such as shopping, education, and entertainment. In accordance with the statistics of International Telecommunication Union (ITU), the number of individuals

using the Internet was over three billion throughout the world in 2015 [1]. Unfortunately, the huge number of users of

the Internet and its facilities may cause great danger in security because of cyber criminal activities. Webpages used

to have static HTML content but now they include technologies that interact with users. This situation may cause

significant risks in online security of the computers.

Webpages containing threats for users are called malicious webpages and the most important security threats

included in these pages are Phishing and Cross Site Scripting (XSS). Phishing (Fishing), is an attempt to obtain

personal information of the Internet users by making use of social engineering. Malicious webpages that use phishing

try to steal user names, passwords, e-mail addresses, phone numbers, photos, social security numbers and even credit

card details of victims [2]. The operation mechanism of phishing is based on impersonating another user and/or

official web site. For example, webpages may include links to web sites different from where the user thinks he/she

is reaching. These links can download a harmful executable to victims’ computers or can open another malicious or

undesirable document. The phishing webpages may request personal information and users can be easily deceived

(see Figure 1). The users may be exposed to fake advertising and counterfeit products purchases because of the

fake webpages. If the user buys a product, the sold item can be an imitation, an illegal one or even an empty box.

∗Corresponding author

Email address: tdokeroglu@thk.edu.tr (Tansel Dokeroglu)

Preprint submitted to Soft Computing January 22, 2018

The technique, XSS, gives opportunities to attackers to inject malicious code into webpages. After the injection,

the victim’s browser/computer becomes vulnerable to further attacks and/or sensitive information leaks [3]. It has

become a major issue especially after the advances in webpage design by using the scripting technologies. The driveby-download techniques make the web service development easier, powerful and flexible [4]. However, the power and

flexibility in recent webpages provides a new tool that can be misused by attackers. A recent study shows that there

are too many malicious webpages in search results [5]. Because of their importance , variety, and intensity, filtering

malicious webpages has become an important research topic.

Figure 1: Flow of Phishing process [6]

In order to handle this problem, various solutions have been proposed. The most widely known technique is the

blacklist approach. Browsers and security tools have these lists that contain malicious web domains and URLs. If

a requested URL is found on the blacklist of Google Safe Browsing, browser does not accept the page. However,

blacklist approach has deficiencies; (1) the lists include only the crawled webpages, (2) crawlers can not reach intranets, (3) crawled pages may be hacked after the crawling, (4) they need a malicious webpage detection mechanism

or human assistance during the production of the list [7]. The second method is the creation of honeypots with Virtual Machines (VM). By using VM environments, visiting a webpage is simulated and its effects may be observed.

It is a successful method but not efficient due to its high execution time. Therefore, this method may help creation

of blacklists but it is not a suitable real time classification processes [8]. The third method is signature check that

is implemented only for classifying executable codes, not for phishing or scripting. Its performance is not good [9]

[10]. Recent studies have focused on automated solutions using Machine Learning (ML) methods. In this study, we

propose novel context-sensitive and keyword density based supervised ML algorithms using Support Vector Machine

(SVM), Maximum Entropy (MaxEnt), and Extreme Learning Machine (ELM).

Since it is easy to prepare data and training is fast with a small number of features, non-content based features

have been preferred generally on malicious webpage detection with ML techniques. However, in our opinion, each

word of an HTML content can provide some clues about the behavior of the webpage. Therefore, we focus on the

content of the webpages rather than adding other features like URLs, screen-shots, DNS server relationships etc.

These mentioned features may have totally different characteristics/semantics and they may degrade the expected

results. By considering the data as not only a text but also a webpage, we propose a new method (keyword density)

for deciding weights of features while the other studies use conventional methods, existence or frequency of keywords.

In addition to feature modifications, we study the ML techniques, SVM, MaxEnt, and ELM.

SVM has been used in most of the related works and proved its success on text classification. We propose MaxEnt

because of its success on document classification [11] [12] and it has not been implemented for any malicious webpage

detection study until now. ELM provides faster learning speed and less human intervention than SVM [13] [14]. We

study with ELM because of its learning speed with 100 thousand webpages and 800 thousand features. By increasing

2

the efforts on data processing phase, we are able to increase the accuracy level of detecting the malicious webpages

up to 98.28% true positive ratio [10]. Even the most successful recent studies provide 97.8% true positive ratio with

2.2% false positive. Therefore, our method can be reported among the best performing state-of-the-art approaches.

**URLNet: Learning a URL Representation with Deep Learning for Malicious URL Detection**

**Hung Le, Quang Pham, Doyen Sahoo, Steven C.H. Hoi**

1 INTRODUCTION

Malicious URLs are one of the primary mechanisms to perpetrate

cyber crimes. They host unsolicited content and attack unsuspecting users, making them victims of various types of scams (theft of

money, identity theft, malware installation, etc.). This has resulted

in billions of dollars worth of losses every year [16]. It has thus

become imperative to design robust techniques to detect malicious

URLs in a timely manner. Traditionally, and most popularly, this

detection is done through the usage of blacklisting methods. These

are essentially lists of URLs collected by anti-virus groups which are

"known" to be malicious. They are often collected through crowd

sourcing solutions (e.g. PhishTank [31]). While these methods are

fast (requiring a simple database lookup), and are expected to have

low False Positive rates, a major shortcoming is that they cannot

be completely exhaustive, and in particular they fail against newly

generated URLs. This is a severe limitation as new URLs are generated everyday. To address these limitations, there have been several

attempts to solve this problem through the use of machine learning [33]. In particular machine learning models offer the ability to

generalize their predictions on new unseen URLs.

Malicious URL Detection through machine learning typically

comprises two steps: first to obtain an appropriate feature representation from the URL, and second, to use this representation of

the URL to train machine learning based prediction models. The

first step of obtaining feature representation deals with obtaining

useful information about the URL that can be stored in a vector so

that machine learning models can be applied to it. Various types

of feature have been considered, including lexical features, hostbased features, content features, and even context and popularity

features [33]. However, the most commonly used features are lexical features, as they have demonstrated good performance and

are relatively easy to obtain [2, 26]. Lexical features describe the

lexical properties obtained from the URL string. These include statistical properties such as length of the URL, number of dots, etc. In

addition, Bag-of-Words like features are often used. Bag-of-Words

indicate whether a particular word or string appears in the URL

or not. Consequently, every unique word in the training dataset

becomes a feature. Using these features, in the second step, prediction models such as SVMs are trained. These models can be viewed

as a form of fuzzy blacklists.

While the above approaches have shown successful performance,

they suffer from several limitations, particularly in the context of

very large scale Malicious URL Detection: (i) Inability to effectively

capture semantic or sequential patterns: Existing approaches rely on

using Bag-of-Words features, which essentially give information

about the presence of a word in the URL. They fail to effectively

capture the sequence in which words (or characters) appear in

the URL String; (ii) Require substantial manual feature engineering:

arXiv:1802.03162v2 [cs.CR] 2 Mar 2018

Conference’17, July 2017, Washington, DC, USA Hung Le, Quang Pham, Doyen Sahoo, Steven C.H. Hoi

many of these approaches require expert guidance to determine the

important features for the task (e.g. which statistical properties of

the URL to use, what type of n-gram features would be better, etc.);

(iii) Inability to handle unseen features: During prediction, test URLs

are likely to contain new words that did not exist in the training

data. Under these circumstances, the trained models are unable to

extract any useful information about the URL from these words.

Moreover, the number of unique words in URLs can be extremely

large, causing severe memory constraints while training models.

To address the above issues we propose URLNet, a Deep Learning based solution for Malicious URL Detection. Deep Learning

[13, 23, 35] uses layers of stacked nonlinear projections in order

to learn representations of multiple levels of abstraction. It has

demonstrated state of the art performance in many applications

(computer vision, speech recognition, natural language processing,

etc.). In particular, Convolutional Neural Networks (CNNs) have

shown promising performance for text classification in recent years

[18, 39]. Following their success, we propose to use CNNs to learn

a URL embedding for Malicious URL Detection.

Specifically, URLNet receives a URL string as input and applies

CNNs to both characters and words in the URL. For Character-level

CNNs we first identify unique characters in the training corpus,

and represent each character as a vector. Using this, the entire URL

(a sequence of characters) is converted to a matrix representation,

on which convolution can be applied. Character CNNs identify

important information from certain groups of characters appearing

together which could be indicative of maliciousness. For Word-level

CNNs, we first identify unique words in the training corpus, delimited by special characters. Using a word-embedding matrix, we

obtain a matrix representation of the URL (which in this context, is

a sequence of words). Following this convolution can be applied.

Word-level CNNs identify useful patterns from certain groups of

words appearing together. However, using word-embeddings faces

some challenges: (i) it cannot obtain embeddings for new words

at test time; and (ii) too many unique words (specifically in malicious URL detection) - resulting in memory constraints while learning word embeddings. To alleviate these, we propose advanced

word-embeddings where the word embedding is learned using the

character-level information of each word. This also helps recognize

subword level information. Both Character-level and Word-level

CNNs are jointly optimized to learn the URLNet prediction model.

URLNet allows us to alleviate the shortcomings of traditional

approaches such that (i) Character and Word CNNs automatically

identify and learn the semantic and sequential patterns in which

the characters and words appear in the URL; (ii) Expert feature

engineering required is reduced, since the CNN automatically learns

features to represent the URL, and we do not rely on any other

complex or expert features for the learning task; and (iii) The model

learns patterns based on both character and word embeddings.

Due to the limited number of characters, this character embedding

can generalize to new URLs easily. For word-embeddings, even

if the test URLs contain new unseen words, the character-based

(advanced word) embedding of the words still allows us to obtain

representation for these new words. This way URLNet has superior

generalization ability compared to existing approaches. We conduct

extensive experiments, analysis and ablation studies to show the

efficacy of proposed method.

**Click traffic analysis of short URL spam on Twitter**

SECTION I.Introduction

Social networks attract millions of users who want to share information and connect with people. Twitter, a popular social network, has over 400 million members and it allows them to post 140-character tweets (messages) to their network of followers. Given the limited length of a tweet, URL shorteners have quickly become the de facto method to share links on Twitter [1].

Twitter, due to its large audience and information reach, attracts spammers [2], [3], [4], [5], [1], [6], [7], [8]. Even though spammers have limited flexibility with the 140-character limit for a tweet, they utilize URL shorteners to camouflage their spam links [9], [10], [11], [12]. This enables spammers to hide the true domain of the URL, thereby might prevent Twitter from effectively applying blacklists to filter out such spam.

The popular URL shortener websites such as Bit.ly (hence-forth referred to as Bitly) provide interfaces that allow users to convert long URLs into short URLs [13], [14]. After receiving a long URL, the services typically use a hash function to map the long URL to a short string of alphanumeric characters, which is then appended to the domain name of the shortener and returned as the short URL. For instance, the long URL http://www.google.com might be shortened as http://bit.ly/olDmsz. The hash function takes into account several factors, such as whether the long URL has already been mapped to a short URL. In this case, the shorteners typically return the existing short URL rather than generating a new one for the input long URL.

In this paper, we perform an analysis on short URL spam by investigating their click traffic with the following goals. First, we aim to determine the feasibility of efficiently collecting the click traffic of short URLs. This is important because a social network typically contains a massive number of short URLs and an efficient mechanism is needed to collect their click traffic. Second, we aim to discover significant patterns in the click traffic of a given set of spam short URLs. Third, we aim to determine the feasibility of detecting short URL spam effectively. This is particularly important because spam can lead to loss and damage [15], [16].

The highlights of our work can be summarized as follows:

We generate a large-scale click traffic dataset for short URL spam;

We obtain several findings about short URL spam through an in-depth analysis of creators and click sources of short URLs;

We demonstrate the feasibility of detecting short URL spam by classification based on the click traffic features.

**You Look Suspicious!! : Leveraging Visible Attributes to Classify Malicious**

**Short URLs on Twitter**

1. Introduction

Twitter is a very popular micro-blogging Online

Social Network (OSN) where people post short

messages, up to 140 characters, known as tweets.

Due to the tweet length limitation on Twitter, URL

shortening services are widely used to save text space

in tweets. As of June 30, 2015, Twitter has 316

million monthly active users and 500 million tweets

sent everyday [1]. The large user base of Twitter has

also attracted the attention of cyber criminals who

often target Twitter users for phishing, malware, and

scams. The attacks can be disguised using relevant

Twitter trends and propagate in the network by

exploiting social relationships on Twitter. Thus, these

attacks are likely to have high success rate [2]. Due to

the URL shortening service used on Twitter, it is easy

for cyber criminals to hide the real malicious URLs.

Detecting malicious URLs is critical but also very

challenging.

A URL shortening service takes a long URL,

typically complicated, from users and converts it into

a Short URL (alias to the long URL). The short URL

can then be shared with other users. Short URLs

reduce long URLs up to 91% in length [3]. When

users click on a short URL, they will be directed to

the URL shortening service provider, which will then

redirect the users to the original long URL, i.e.

landing page. The process happens automatically

without user intervention. This helps users share long

and complicated URLs. URL shortening services

were introduced in 2001, but the adoption was very

slow until it became popular in online social

networks. There are more than 500 URL shortening

services available today. Some of them are free while

many are proprietary. Bitly and Tinyurl are the most

famous free URL shortening services. Twitter uses its

own URL shortening service where every URL

posted on Twitter is converted to t.co/xxxxx.

Unfortunately, malicious users also leverage short

URLs to their advantage. Short URLs have been used

in phishing attacks, spamming, scams, malware

propagation, etc. Cyber criminals use short URLs to

exploit the limited text space on Twitter and hide the

destination of the URLs [4], [5]. A short URL works

as a cloak to its nefarious purpose. Online Social

Networks often use blacklisting services to remove

malicious URLs. However, blacklists are not always

comprehensive and up-to-date [6]. In addition, social

network users often have an expectation of strong

security from the URL shorting service providers and

online social networks while both URL shortening

services and OSNs fail to provide [5].

Because of these threats, short URLs have become

a problem in information sharing and security in

many fields including e-Government. The

willingness of users to use e-Government services

depends on the trust they have on security provided

2016 49th Hawaii International Conference on System Sciences

1530-1605/16 $31.00 © 2016 IEEE

DOI 10.1109/HICSS.2016.332

2648

Authorized licensed use limited to: University of Dayton Libraries. Downloaded on September 08,2023 at 07:29:04 UTC from IEEE Xplore. Restrictions apply.

by the services. According to the research conducted

in [7], 81.6% e-Government websites are vulnerable.

In addition to the vulnerabilities in the system itself,

inadequate protection from OSNs can also leave

users vulnerable to these threats. Users can be social

engineered with tweets and directed to malicious

URLs. This would not happen if the malicious URLs

could be detected effectively and removed from the

OSNs.

Government, corporations, education, and general

public, all are connected in information exchange and

sharing in e-Government. Although many

governments and organizations have adopted social

media policy and awareness program, short URLs do

not receive the attention they deserve. Government

and organization policy makers should understand

the scope of the problem better to protect

organizational assets and public interests.

In this research, we propose a novel approach to

classify short URLs on Twitter. We leverage visible

features of tweets such as content, context, and social

information to classify short URLs as malicious or

benign. We tested four different machine learning

classification algorithms and achieved an accuracy of

up to 97% detection rate using random forest

algorithm with 10-fold cross validation. Our

approach can be used by OSNs or any organizations

that want to assess the malignity of the short URLs.

**Deep Approaches on Malicious URL Classification**

I. INTRODUCTION

With the rapid digital transformation taking place, most

of our activities are happening online, thereby increasing

the chances of online crimes.Attackers are making their way

into the sensitive information by tricking users into revealing them , leading to theft of money, reveal of identity,

blackmailing,installing malware in the user’s system. Attacking techniques include explicit hacking attempts, drive-by

download [5], social engineering, phishing,man-in-the middle, SQL injections, loss/theft of devices, denial of service

etc.According to the research of [14] one-third of the URLs

nowadays are malicious thus marking the impact it has on

cyber-crimes. Now-a-days being successful in any field be

it in business, education, sports an online presence proving

their skills is necessary. People also need handful of resources

which are just a few clicks away. As a result the importance

of World Wide Web (WWW) is increasing.Websites have an

address referring to them, which are known as URLs (Uniform

Resource Locator).There are client side or the users and server

side or the providers.The connection between the client and

server is protected by some protocols or rules, yet they are

vulnerable to the outsiders who have a sharp intention of

attacking them.A URL constitutes of two major components1. Protocols identifier (it helps to find what protocol to use)

and 2. Resource name (IP address or name of the domain

where resource is present). Now as the modern technology

advances, the risk of corruption increases.As a first step if we

can detect which of the URLs are malicious and which are

not a great deal of attacks can be prevented,because most of

the times malicious URLs are the door-ways for these attacks.

So we can realize the importance of the topic, as a result

a lot of research work has also been done on this problem.

But most of them used traditional machine learning methods.A

few tried the deep learning methods.In our work we have used

three different architectures of