LinkMan: Hyperlink-Driven Misbehavior Detection in Online Security Forums

Risul Islam
UC Riverside
risla002@ucr.edu

Ben Treves
UC Riverside
btrev003@ucr.edu

Md Omar Faruk Rokon
UC Riverside
mroko001@ucr.edu

Michalis Faloutsos
UC Riverside
michalis@cs.ucr.edu

Abstract—How can we detect and analyze hyperlink-driven misbehavior in online forums? Online forums contain enormous amounts of user-generated content, with threads and comments often posted with malicious intention and we refer to this as ‘hyperlink-driven misbehavior’. We present *LinkMan*, a systematic suite of capabilities, to detect and analyze hyperlink-driven misbehavior in online forums. We take a unique perspective focusing on hyperlink sharing practices of the users to spot misbehavior. *LinkMan* can categorize these hyperlinks as: a) phishing, b) spamming, and b) promoting malicious products. Our approach consists of three high-level phases: (a) extracting hyperlinks from the textual data, (b) identifying misbehaving hyperlinks, and (c) modeling the behavioral patterns of hyperlink sharing, where we identify key hyperlinks and analyze the collaboration dynamics of hyperlink sharing. In addition, we implement our approach as a powerful and easy-to-use open platform for practitioners. We apply *LinkMan* to spot misbehavior from three online security forums, where we expect the users to be more security-aware. We show that our approach works very well in terms of retrieving and classifying hyperlinks compared to previous solutions. Furthermore, we find non-trivial and often systematic misbehavior: (a) we find a total of 637 misbehaving hyperlinks, and (b) we identify 30 colluding groups of users in terms of promoting hyperlinks. Our work is a significant step towards mining online forums and detecting misbehaving users comprehensively.

Index Terms—Security Forums, URL Extraction, Phishing Detection, Misbehavior Detection.

INTRODUCTION

How pervasive is spamming and phishing in online forums? This is the question that motivates our work. Spamming and phishing have emerged as key nuisances and threats with the widespread adoption of email. Recently, email filters have improved and the use of spam-aware services like Gmail have contained the reach of these activities. However, hackers are tenacious and are always on the search for new ways to accomplish their goals as new technologies emerge. One such new opportunity is presented by online forums, which have seen a tremendous increase in both number and user engagement. Currently, there are 1M+ forums with an estimated 550M+ registered users [1].

Here we take a more niche angle and we focus on security forums, where one would assume that users are more aware and

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ASONAM '21, Brijuni, Croatia, May 2021, The Netherlands

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-9128-3/21/05...\$5.00.

<https://doi.org/10.1145/3487351.3488323>

Figure 1: Overview of the three-phase approach of *LinkMan*: (a) extract the URLs, (b) identify and classify URLs, and (c) model the misbehaving group behavior.

thus less likely to fall victim to spamming and phishing. Online security forums bring together a wide variety of users generating enormous amounts of security-related content [2] through their comments which are often supplemented by hyperlinks. Clearly, there are benign usages of these hyperlinks that can point to useful information. However, we find that sharing hyperlinks often accompanies malicious intentions which we call ‘hyperlink-driven misbehavior’. This misbehavior can be broadly grouped into: (a) phishing, (b) spamming, and (c) sharing of malicious products. Fully quantifying this misbehavior can reveal malicious hackers using security forums for malicious purposes. Throughout this paper, we use the terms *hyperlink*, *link*, and *URL* interchangeably.

The problem we address here is the following: *How can we detect and analyze hyperlink-driven misbehavior in online forums?* The question is motivated by the studies showing that malicious hackers have a strong presence in public forums: (a) they spread malware often masquerading as technical solutions or antivirus and create an online brand [3], and (b) they promote malicious services [4]. We narrow down our focus to: (a) security forums, and (b) misbehavior that is enabled via hyperlinks in posts. Therefore, the input to our problem is posts from online forums, which include the author and date. The outputs are: (a) the misbehaving hyperlinks, (b) the type of misbehavior, and (c) the persistent offenders and their collaboration patterns.

There has been limited work focusing on the question as we frame it here. We are not aware of any work which identifies misbehavior from the perspective of hyperlink sharing in online forums. In fact, mining security forums in general has received relatively recent little attention. We can identify two main categories of related efforts: (a) security forum studies and (b) hyperlink classification studies. We discuss these efforts in our related work section.

As our key contribution, we propose *LinkMan*, a comprehensive methodology for detecting and analyzing hyperlink-driven misbehavior in online security forums. Our approach consists of the following key functions: a) URL extraction, b) systematic classification of URLs, and c) modeling of emerging URL posting behaviors as shown in Fig. 1.

From an algorithmic point of view, *LinkMan* makes the following contributions. We develop *RPhish*, a novel machine learning-based approach to detect phishing websites. In this approach, we consider an exhaustive set of features along three dimensions: (a) name-related features, (b) network/reputation level information, and (c) web-page content. For the classification, we combine Principle Component Analysis (PCA) for feature compression with a five-layer neural network. In addition, we develop *ExtLink*, a systematic approach to extract hyperlinks from raw text. In modeling user and group behaviors, we use a combination of tensor decomposition and DBSCAN for detecting outlier behaviors.

Our key results can be summarized in the following points.

a. Our proposed methods perform well. Our phishing detection algorithm, *RPhish*, demonstrates 98.2% accuracy, 97.01% True Positive Rate, and 1.3% False Positive Rate, beating the current best phishing detection models. In addition, our URL extraction method, *ExtLink*, exhibits a 26.7% increase in precision without any reduction in recall compared to baseline regular expression-based extraction.

b. There is a non-trivial amount of malicious URLs in security forums. We find a total of 637 misbehaving URLs including phishing, spamming, and malicious product sharing. Some of these activities are even aggressive: 9 URLs are aggressively promoted 1176 times by a group of 80 users.

c. We identify significant collaborative groups which promote the same URLs. Using tensor decomposition, we identify 30 tight-knit groups of users in terms of their URL posting over time. For example, we find a group of 17 users who jointly promoted decryption tools during a ransomware outbreak in December 2015. Interestingly, two of these users are associated with the creation of ransomware.

Our work in perspective. The proposed work is part of an ambitious goal: we want to track down malicious hackers and understand their activities. Our initial results are promising: malicious activities are prevalent in security forums. With appropriate follow-up work, achieving this goal can have a huge practical impact: security analysts could prepare for emerging threats, anticipate malicious activity, and identify their perpetrators.

Open-sourcing for maximal impact. We intend to make our tool and datasets public for research purposes. In our development, we use Python v3.6.2 and several related packages.

DATASET AND TERMINOLOGY

We provide a brief description of our dataset and explain the terminology used throughout this paper.

Dataset. We use data from three security forums: Offensive Community, Hack This Site, and Ethical Hacker [5] spanning 5 years from 2013 to 2017. The datasets were collected by our early efforts which can provide more details [2], [6]. The data of a forum consists of the following: forum ID, thread ID, post ID, username, date, and post content. These forums

Table I: Basic statistics of our forums.

Entity	OC	HTS	EH
Total User	5412	9420	2970
Total Thread	3214	9303	8740
Total Post	23918	68434	54116
Total Active Days	1860	3720	4464

are in English and their users discuss a wide range of security-related topics. The users range from security professionals to hobbyists, but some are also malicious hackers. Some basic statistics of the dataset are shown in Table I.

Terminology. A *thread* is started by its first *post*, and we refer to subsequent posts as *comments*. The term *entity* refers to either a *user*, *thread*, *post* or *day*. If a *user*, *thread*, *post*, or *day* contains at least one hyperlink in the contents, we refer to them as ‘*LinkUser*’, ‘*LinkThread*’, ‘*LinkPost*’, or ‘*LinkDay*’, respectively.

METHODOLOGY

As our key contribution, we develop *LinkMan*, a systematic suite of capabilities, to detect URL-driven misbehavior in security forums. Fig. 1 demonstrates the overview of our approach. It consists of three high-level phases: (a) extracting URLs, (b) identifying and classifying misbehaving URLs, and (c) modeling behavioral patterns.

Phase 1: Extracting URLs

We describe the challenges in extracting URLs from text.

Challenges in URL extraction. Extracting URL is a non-trivial task because: (a) URLs can follow a relatively complex structure, with significant diversity in the types of protocols and Top Level Domains, e.g. https, .com, .co, .io, (b) sometimes URLs include IP addresses and port numbers instead of text, and (c) human errors such as typos and missing characters can introduce noise. Here is a list of legitimate URLs: <http://213.32.103.5/cgi-sys/defaultwebpage.cgi>, facebook.com, <https://facebook.com>, www.facebook.com.

Limitations of using regular expressions. Most well-established approaches extract hyperlinks from the HTML code, but this is not possible here since we have unstructured text. For such text, most studies [7], [8] and popular online tools [9] use methods that are based on regular expressions (**RegEx**). These methods do not overcome the limitations we mentioned above, and we quantify this in our ‘Results’ section.

Our URL extraction method. We propose *ExtLink*, a method to extract URLs of a variety of formats, types, and structures efficiently from textual data. Note that we define a string to constitute a legitimate website URL if it represents a registered Top Level Domain (domain suffix) e.g. facebook.com is legitimate, while facebook.aa is not.

We start by tokenizing the raw text. Even this operation is not trivial if we want to handle typos, missed or added blank spaces, common misspellings, etc. The next challenge is to determine if a token (word) is a legitimate URL, which we do in three steps. First, we identify candidate URL structures containing at least one dot, such as facebook.aa/groups. Second, we parse the structure to identify the domain name and the domain suffix, e.g. domain name = “facebook” and domain suffix = “.aa”, assuming that the input is an URL. Third, we

Algorithm 1: Algorithm to decide whether a given token is a legitimate URL.

```

Input: Token
Output: URL = True or False
1  URL = False
2  if isCandidate(token) == True then
3      domain, domain-suffix = FindParts(token)
4      if isLegit(domain-suffix) == False then
5          if isValidIP(domain) == True then
6              URL = True
7          else
8              URL = False
9      else
10         if isLegit(domain) == False then
11             URL = False
12         else
13             URL = True
14 return URL==True

```

validate our assumption by checking if the domain name and the domain suffix make a legitimate URL. In our example, "facebook.aa" is not a legitimate URL. We present the logic in deciding whether a token is a URL in Algorithm 1. We discuss the functional modules presented in Algorithm 1 later in the next section.

Phase 2: Identifying and Classifying Misbehaving URLs

We classify the websites that we find from the previous step into the following categories: (a) phishing, (b) spamming, (c) malicious products, and (d) miscellaneous. The miscellaneous category includes websites that are benign or that we can not confidently label as malicious.

Part 1. Detecting phishing. We propose a novel machine learning-based method to detect phishing URLs. Phishing websites try to steal user account passwords or other confidential information by tricking visitors into believing they are on a legitimate website. Naturally, attackers attempt to attract as many visitors as possible to such websites.

Our phishing detection algorithm. Our phishing URL detection approach consists of three high-level steps: (a) feature engineering using compression, (b) training the model, and (c) classifying the unknown URLs.

(a) Feature engineering using compression. As a first step, we compress the dimensions of our training data. The features in the training data spread along three dimensions as mentioned in the Introduction. The details of our ground truth data are described in the 'Results' section. We use the traditional, well-established dimension reduction algorithm PCA [10] to compress the features. We find that feature compression enhances the separability between the target classes (phishing and legitimate) and yields better performance.

(b) Training the model. After the dimension (feature) reduction, we train a neural network to detect phishing websites. The architecture consists of the input layer with 3 neurons, 3 hidden

layers with 50 neurons in each, and one output layer with a single neuron (3-50-50-50-1). Out of all the architectures we tested, this architecture delivers the best performance, which we present in the 'Results' section.

(c) Identifying phishing URLs. Finally, we use the trained model to classify the unknown URLs as either phishing or legitimate.

The details of the parameter choices of PCA and the neural network as well as the analysis of the model performance are discussed in the next section.

Part 2. Detecting spamming. In general, the term spammer refers to a user who repetitively posts the same content. In our URL-centric study, the definition focuses on the repetitive posting of the same URL across many posts.

We follow the observations below to detect spamming URLs. First, spammers usually comment more than the general users. The frequency of the hyperlink contained in these *spam comments* generally outnumbers the frequency of hyperlinks contained in regular *comments*. That means the sole purpose of the spammers is to spam a hyperlink, which we call a 'dominant' hyperlink, although they may rarely post other general hyperlinks as well. Therefore, a higher percentage of 'dominant' hyperlinks in *comments* is a distinctive feature of spammers. Second, higher average similarity in all possible pairs of *comments* is also a distinctive feature of the spammers as they intentionally post the same content over and over again.

In a nutshell, we consider a user to be a spammer if: (i) the user posts a particular hyperlink more than a threshold, T_{freq} , (ii) his/her percentage of the *dominant* hyperlink is above a threshold, T_{dom} , and (iii) his/her average similarity in all possible pairs of *posts* is more than a threshold, T_{sim} . We then tag that user as 'spammer' and the corresponding *dominant* hyperlink as 'spamming hyperlink'. We present the choices of the thresholds and results from the security forums in the 'Results' section.

Part 3. Finding malicious products. Users in security forums often advertise malicious hacking tool-selling websites which we refer to as 'malicious product' URLs. To identify URLs that share malicious products, we search for the URLs in a ranked list of malicious product-selling websites. First, there are several platforms, such as Alexa [11] and Hackerone [12], which maintain extensive lists of well-known or popular websites of different categories. Second, an analysis of the landing page of a website can often indicate the type of services that it provides. This makes it possible to miss lesser-known websites, but as we will see later, we already find significant activities in this space. In this work, we use available lists of malicious product selling websites and will perform a content analysis of these websites in the future.

Part 4. Miscellaneous. In this category, we place all the hyperlinks that we can not confidently assign to any of the previous three categories. We attempt to classify them into the following three sub-categories: (a) technical security tutorials and information, (b) financial institutions and services, and (c) file and code-snippets sharing such as GitHub. Performing this classification is a challenging problem in its own right. Here, we leverage external resources that classify websites by their primary function. Specifically, we look for the URLs which are present in a list of Alexa top 500 (a) video sharing, (b)

Table II: The precision and recall of each method. Our *ExtLink* provides the highest precision without sacrificing recall.

Method	Precision	Recall
RegEx	72.8	98.7
ConvertCSV	84.7	51.5
<i>ExtLink</i>	99.5	98.7

financial organization, and (c) file and code sharing websites. In the future, we intend to develop a more exhaustive solution for this category. We present our findings in the ‘Results’ section.

Phase 3: Modeling Behavioral Patterns

LinkMan analyzes the behavioral patterns of users regarding the sharing of URLs. We answer the following two questions in this analysis.

(a) *What are the most popular URLs?* We detect the URLs that are promoted by a lot of individuals by finding outliers from a 2D scree plot where each point in the plot represents each URL’s frequency (Y-axis) and the number of users that promoted the URL (X-axis) for each security forum. We use the DBSCAN algorithm to identify the outlier URLs. The findings are discussed in the ‘Results’ section.

(b) *How do the users collaborate to promote URLs?* We find groups of URLs that are promoted by groups of individuals over the time to understand the collaboration dynamics of URLs posting. We leverage tensor decomposition-based analysis, a very well-established approach to analyze group dynamics. To analyze the dynamics, we resort to the state-of-the-art tensor-based tool, TenFor [2].

a. Our tensor-based decomposition. The input to TenFor is a 3D tensor T , where each 3D element, $T[i,j,k]$, captures the interaction of *LinkUser* i promoting URL j at weekly discretized time k . We provide the *LinkPosts* to TenFor as well. The output is a number of clusters where each cluster consists of a group of users, URLs, and weeks and captures a significant event or activity. For example, one such cluster in the OC forum represents a group of 29 users that are active in the first weekend of July 2016 and promotes a group of 43 video-sharing URLs for novice hackers.

b. Identifying clusters of interest. Our goal is to report surprising and interesting clusters. To do that, we perform the following three tasks. First, we create a 3D scree plot where each point denotes a cluster and each axis denotes the number of elements in each dimension (axis 1- number of users, axis 2- number of URLs, axis 3- number of week bins for that cluster). Second, we find the outlier clusters using DBSCAN outlier detection algorithm. Finally, we report the events hiding in the outlier clusters using “Storyline View” from TenFor, which captures a textual and visual summary of the essence of each cluster. We discuss the interesting findings from each forum in the next section.

RESULTS

We apply our method on the three security forums in our archive. *LinkMan* provides misbehaving URLs of different categories as well as other entities of interest (users, threads, posts, time intervals) associated with these. Table IV shows the summarized output from *LinkMan*. We present the results, evaluations, and interesting findings from each category separately.

Phase 1: Extracting URLs

Our approach, *ExtLink*, extracts hyperlinks from the textual data of a forum. Specifically, upon tokenizing the posts using ‘space’ as the delimiter, we follow the steps presented in Algorithm 1 to decide whether each token is a URL.

The *FindParts* function is responsible for factorizing the candidate token into domain name and domain suffix (TLD). In our current implementation, we use the ‘tldextract’ package of Python for factorization. If the domain suffix is not in ‘Public Suffix List (PSL)’, a periodically-updated list of private and public domain suffixes available on internet maintained by Mozilla, it sets *domain suffix* = *empty* and the domain name accordingly [13]. We find that ‘tldextract’ package is the most suitable tool for our purpose since other tools, such as ‘purse_url’ package of R language, do not utilize the PSL as ‘tldextract’ does.

We found that it was necessary to develop our own URL extraction method because traditional RegEx-based URL extraction methods have drawbacks that lead to poor performance. First, while generating the format of the URLs is easier using RegEx, manually configuring different types of protocols, TLDs, IP addresses(v4 and v6), optional port numbers, etc. becomes extremely hard and tedious to generalize. Second, using automated tools to generate RegEx by providing examples requires numerous handpicked examples which is a tough task. An alternative method of URL extraction is to use online tools such as convertCSV, browserlink, miniwebtool, etc. but they also use RegEx internally and can report URLs that adhere to a few specific formats only.

We evaluate *ExtLink* and report the findings below.

Evaluation. Our evaluation suggests that *ExtLink* outperforms existing methods in terms of both precision and recall. We compare the performance with: (i) **RegEx**-based approach, which we explain below, and (ii) popular online tools: **ConvertCSV** and **Browserlink** [9]. For consistency, we use the same RegEx-based method that was used in previous works [7], [8]. Our method is able to “recognize” many standard and commonly-used URL formats: [scheme:][//authority]path[?query][#fragment] where scheme can be http, https, file, etc. and authority = [user-info@]host[:port] [14].

***ExtLink* outperforms RegEx: better precision with the same recall.** We find that *ExtLink* outperforms the RegEx-based approach in terms of precision due to the RegEx results containing more false positives. For example, many RegEx-returned URLs are typos where the user failed to add a space at the end of a sentence. As a result, phrases like ‘quickly.we’ and ‘go.I’ are returned as URLs.

In the absence of established ground truth, we conducted the following study to quantify both precision and recall. We generated a set of 500 randomly selected posts, D_{post} , from the OC forum and manually extracted a total of 237 URLs from it. We applied all approaches on D_{post} and compared the outcomes. Table II summarizes the comparative results. We found that *ExtLink* extracted 234 valid URLs out of its 235 reported URLs yielding a precision of 99.5%. *ExtLink* missed only 3 URLs yielding a recall of 98.7%. RegEx yielded the same recall as *ExtLink* but a lower precision of 72.8% (234 valid out of 321 extracted URLs).

Table III: The URLs extracted from each forum by each approach.

Forum	ExtLink	RegEx	ConvertCSV	Browserlink
OC	22599	24577	8990	4573
EH	5458	6306	2450	1221
HTS	13880	14350	4687	2917

Table III shows the number of URLs from each security forum with *ExtLink* and reference methods. *ExtLink* identifies significantly more URLs than *ConvertCSV* and *Browserlink*. These online tools report fewer URLs because they use very stringent URL formats. *RegEx* returns the largest number of URLs from all three methods, but as we previously discussed, this comes at the cost of poor precision.

We utilize another systematic approach to assess the precision. We generate a set of 500 randomly sampled URLs, D_{url} , from the URLs detected by both *ExtLink* and *RegEx* and manually cross-check the validity of the URLs in D_{url} using another domain expert. *ExtLink* identified 498 valid URLs (precision 99.6%) from D_{url} whereas *RegEx* found only 391 valid URLs (precision 78.2%).

Results from applying *ExtLink* on real data. We find that our algorithm extracts a total of 41937 URLs (OC 22599, HTS 13880, EH 5458) from the security forums. We find a total of 3718 *LinkUsers*, of which 820 are from OC, 2264 are from HTS, and 634 are from EH forum. In total, we report a sum of 8236 *LinkThreads*, 6702 *LinkPosts*, and 2986 *LinkDays* from the forums.

We provide some basic statistics that help us understand the URL posting behavior of the users in security forums. For instance, we find that 10% of the users are responsible for posting 77% of the URLs. The members of the OC community tend to share almost three times more URLs (4.17 URLs/user) than members of EH (1.83/user) and HTS (1.47/user). For OC forum, 82% of *LinkUsers* post at most 6 URLs whereas only 0.1% post more than 700 URLs. Similar trends are observed for HTS and EH forums as well.

Phase 2: Identifying and Classifying Misbehaving URLs

After extracting the URLs, we classify them and report the types of misbehavior that could be of interest to a security analyst. Table IV summarizes the results. Below, we evaluate our classification and highlight some key observations.

Part 1. Detecting phishing. We show that our proposed phishing detection algorithm performs well. We describe the ground truth data, parameter and hyper-parameter choices, the performance of our model, and some interesting findings below.

Ground truth dataset. We use the ground truth dataset from [15] which contains 30 features including 8 URL name-based features (e.g. ‘containing IP address?’, ‘has HTTPS?’), 11 network/reputation-based features (e.g. ‘HTTPS issuer trusted?’, ‘using non-standard port?’), and 11 content-based features (e.g. ‘contain website forwarding?’, ‘use pop-up windows?’). The details of these features can be found in [15].

Parameter and hyper-parameter tuning. As mentioned earlier, we use PCA followed by a Neural Network (NN) for the prediction. Among all compression algorithms such as PCA, Gaussian Random Projection (GRP), and Sparse Random Projection (SRP) that we try, PCA gives the best results in our study (Table V). Furthermore, we show the effect of varying the compressed feature dimension, n , on accuracy and

Figure 2: The effect of the number of features, n , in our dimension-reduction on the performance on Accuracy and False Positive Rate results of PCA + NN. Selecting $n=3$ yields the best performance.

False Positive Rate in Fig. 2. We reach the best performance (accuracy 98.2%, FPR 1.3%) when we set $n=3$ in PCA.

From other NN architectures that we experimented with, 3-50-50-50-1 architecture yields the best performance. We use the sigmoid and categorical cross-entropy as activation and loss functions, respectively. The training-to-test dataset ratio is 0.6:0.4. A comparison against other architectures with a different set of parameters that we tried is presented in Table V. Finally, we use a drop-out ratio of 0.2 and 10-fold cross-validation to ensure that our model is not suffering from over-fitting. We also manually verified all the phishing websites predicted from the security forums to verify that our model does not suffer from over-fitting.

***RPhish* outperforms the baseline approaches.** We identified two methods, CANTINA [16] and Stacking Model [17], as reference points for our approach. These two methods demonstrated consistently strong performance in their respective work. Table VI summarizes the comparison results. The performance of *RPhish* (accuracy of 98.20%, TPR of 97.01%, and FPR of 1.3%) outperforms the existing heuristic-based method CANTINA (accuracy of 95%, TPR of 95%, and FPR of 3%) and DNN-based work Stacking Model (accuracy of 97.2%, TPR of 95.3%, and FPR of 1.61%).

Applying our phishing detection method on the real dataset reveals interesting findings. We find a total of 57 phishing URLs from all three security forums. Table IV shows the breakdown of phishing URLs from each forum.

Some interesting findings about these phishing websites are: (i) 40% (23) of these phishing sites are mimicking Facebook with the obvious intent of stealing user login credentials, (ii) 18% (10) are trying to mimic financial or e-commerce websites (Chase: 3, Discover: 3, Bank of America: 1, Amazon: 2, eBay: 1), and (iii) 41% (24) of these phishing websites URL names are typosquatting versions of the original domain name. Typosquatting is a form of cyber-crime where hackers create fake websites which names are a slight variation of the well-known websites [18].

Part 2. Detecting spamming. The intuition behind our spamming detection is to make use of the nature of spamming. We identify spammers and spamming hyperlinks based on high activity in terms of: (a) hyperlink frequency, (b) percentage of *dominant* hyperlink, and (c) average post similarity. Fig. 3 shows the percentage of *dominant* hyperlinks by users who post more than twenty hyperlinks in the OC forum. We can observe that four users, manually verified as spammers, demonstrate

Table IV: Summary of the output of *LinkMan*. We showcase the aggregated numbers while reporting the results.

Output Type	OC	HTS	EH	Total
Phishing URLs	27	21	9	57
Spamming URLs	4	3	7	14
Malicious Product URLs	178	168	220	566
Tutorial URLs	263	1447	699	2409
Financial URLs	32	5	3	40
File and Code URLs	201	456	137	794
Highly-promoted URLs using scree plot analysis	9 Unique URLs 1176 Posts, 80 Users	17 Unique URLs 12756 Posts, 360 Users	5 Unique URLs 264 Posts, 125 Users	31 Unique URLs 14196 Posts, 565 Users
Colluding clusters Tensor Decomposition	11 Clusters of 237 Users, 197 URLs, 15 Weeks	9 Clusters of 145 Users, 154 URLs, 13 Weeks	10 Clusters of 102 Users, 101 URLs, 11 Weeks	30 Clusters of 484 Users, 452 URLs, 39 Weeks

Table V: Phishing detection accuracy: our phishing detection approach outperforms other approaches. (NN epochs=150, batch size= 128, SVM kernel=RBF. Here, n is the number of features to be compressed to in PCA.

Model	Parameters	Layers	Accuracy
SVM (30 features)	kernel: RBF	N/A	92.5
SVM + PCA	n=4	N/A	96.2
NN(30 features)	sigmoid	3-50-50-50-1	92.2
NN + SRP	n=5	5-100-50-1	90.4
NN + GRP	n=3	3-50-50-50-1	89.4
<i>LinkMan</i>	n=3	3-50-50-50-1	98.2

Table VI: Comparison results: *LinkMan* outperforms prior approaches CANTINA and Stacking Model.

Approach	Accuracy	TPR	FPR	F-score
CANTINA	95.0	95.0	3.0	91.0
Stacking Model	97.2	95.8	1.6	95.0
<i>LinkMan</i>	98.2	97.0	1.3	97.1

a higher percentage of *dominant* hyperlink. Similarly, Fig. 4 demonstrates that the same spammers exhibit significantly higher average post similarity than others.

For the spamming detection algorithm, we use the following threshold values: $T_{freq} = 2*$ average hyperlink per *LinkUser* for the respective forum, $T_{dom} = 50\%$, and $T_{sim} = 50\%$.

Spamming in security forums. We identify a total of 14 spammers of which 4 are from OC, 3 are from HTS, and 7 are from EH. These spammers usually try to advertise new websites. For example, in OC, user *Ghost_lite* promotes an adult site and user *Guru* aggressively advertises a domain ‘mtgoox.com’ which is for sale. In HTS, user *cyberdrain* tried to promote a new security forum, ‘viphackforums.net’, saying that the forum may go down if they can not find more users. In EH, user *Don* tried to disseminate the news of ‘WannaCry’ ransomware outbreak sharing his own post from ‘bitshacking’ forum and tried to sell an anti-ransomware tool in 2017. We manually check these *spamming posts* and verify them as spam.

Part 3. Finding malicious products. In our current implementation, we characterize URLs by using databases that classify websites. Here, we use the Alexa list of the top 500 websites that sell malicious products. The marketing of such products is dynamic. We find that the number of URLs for both malicious and defensive tools increases during major events of malware outbreaks as we discuss below.

Malicious products in security forums. We detect 566 URLs pointing to websites that sell malicious products across our three security forums. In OC, a group of 7 users tried to promote a WiFi hacking tool from ‘virustotal’ in January 2015. In HTS, another group of 23 people recommended a decryption tool from ‘sourceforge’ in the event of spreading

‘Locky’ ransomware in February 2016. Thus, a peak in the posting of these URLs suggests the outbreak of major security events [4], which serve as interesting indirect information for a security analyst.

Part 4. Miscellaneous. We place here all URLs that do not belong to any of the above-mentioned misbehavior categories. We further sub-categorize these URLs to understand more about them. We find a total of 2409 URLs of technical security tutorials and information, 40 financial institutions and services related URLs, and 794 file and code-snippet sharing related URLs. Table IV shows the breakdown of these sub-categorized URLs.

a. Technical security tutorials and information. We detect 2409 security tutorial URLs from all three forums. We report some of the interesting findings here. In OC, user *Dragunman* shared tutorials on hacking into banks throughout June 2015. *VandaDGod*, an expert Linux hacker, shared a popular tutorial series on Hacking in Kali Linux in November 2017 in EH.

b. Financial institutions and services. Posting URLs to financial institutions seems to be related to the selling of security or hacking tools taking advantage in the event of malware outbreaks. For example, one interesting finding is that, among the 32 mentions of financial URLs in OC, 21 of them occurred in the month of December 2015, and February 2016. At that time, ransomware named ‘SimpleLocker’ broke out and some users tried to sell decryption tools for it. These sellers’ post followed almost the exact same pattern: ‘*Decryption key for SimpleLocker. Payment only in chase.com*’.

c. File and code-snippet sharing. Forum users frequently share different files and code-snippets in data storage platforms such as GitHub, Bitbucket, MediaFire, etc. We report 794 URLs and associated entities from the file and code sharing sub-category. Interestingly, we find one URL pointing to ‘MediaFire.com’ in 2015 claiming that it allows the hosting of many malware and viruses.

Phase 3: Modeling Behavioral Patterns

Although there are many interesting behaviors that we can consider, due to space restrictions we are forced to narrow our discussion down to the following two questions.

(a) *What are the most popular URLs?* We detect outlier URLs from a 2D scree plot for each security forum. Fig. 5 presents the scree plot from OC where each point represents each URL’s frequency (Y-axis) and the number of users promoting that URL (X-axis).

Identifying heavy-hitter URLs using scree plots. A total of 31 URLs are found from all three forums. The breakdown is presented in Table IV. We highlight to observations. First, in

Figure 3: Non-increasing percentage of *dominant* hyperlink of the users who post more than twenty hyperlinks in their comments in OC. Four users have significantly higher *dominant* hyperlink percentage.

Figure 4: Non-increasing average post similarity of the users who post more than twenty hyperlinks in their comments in OC. Four users have significantly higher average post similarity.

OC, a group of 18 users promoted ‘crackcommunity.c0.pl’ 577 times to help them have some traffic because someone reported against this forum. Second, in EH, 4 users announced some rule change in ‘hackthissite.com’, because they were administrators of HTS but regular users of EH.

(b) *Is there any collaborative behavior in promoting URLs?* We identify groups of users which seem to work synergistically to promote groups of URLs. To capture this behavior systematically, we use TenFor [2], which utilizes the temporal dimension.

Tensor analysis findings from security forums. We find a total of 30 clusters including 10 surprising and interesting clusters (4 from OC, 3 from HTS, and 3 from EH). All of the entities in these clusters are reported in *LinkMan*.

We manually investigate 10 peculiar clusters and found them interesting. For instance, one cluster in OC is a group of 13 users promoting 43 hacking tutorial URLs in June 2016, and again in August 2016. Another outlier cluster in HTS consists of only two users (administrators) who are promoting a URL pointing to the posting rules of HTS in 10 different weeks again and again when anything severe happens in the security world. Another outlier cluster in EH consists of 17 black market decryption tool sellers posting 19 hyperlinks to their respective selling websites in December 2015, and February 2016, correlating with the ‘SimpleLocker’ ransomware outbreak.

DISCUSSION

We discuss the practical considerations and limitations of our approach.

a. How do we handle misspelled hyperlinks? In our current implementation, *ExtLink* extracts URLs with the correct

Figure 5: Scree plot for OC forum. Each point represents each URL’s frequency (Y-axis) and the number of user promoted that URL (X-axis). We find 9 URLs are aggressively promoted 1176 times by a group of 80 users.

format and spelling. Our rationale is that we detect URLs that would lead to a website if they were copy-pasted to a browser. In this work, we do not report URLs with erroneous formats or typos such as ‘facebook .com’, ‘http://www.google.com’, and ‘ww.127.0.0.1’. Considering erroneous URLs introduces a trade-off where we could end up reporting URLs that are: (a) plain wrong or non-functional or (b) not intended to be used e.g. `www[dot]malware[dot]com`. In this instance, saying that the user is sharing malware would be incorrect.

b. How can a practitioner use our tool? Our approach is ready for use in a straightforward way. Our plan is to share our code and datasets. Therefore, a practitioner needs to only provide the forum dataset: $F := (\text{thread ID}, \text{post ID}, \text{username}, \text{date}, \text{and post content})$. *LinkMan* takes care of the rest and generates the desired output.

Note that our approach can work with any online forum, not just security forums. We have tested *LinkMan* on different types of forums, including a gaming forum, and observed interesting findings. Due to space constraints, we can not report these results here. Focusing on security forums is interesting in its own right: (a) these forums are hardly explored, and (b) previous works suggested that they hide a wealth of information including malicious activities [19], [20], [4], [21].

RELATED WORK

Studying hyperlink-driven misbehavior in security forums has received very little attention from the research community. Most studies differ from our work in that either: (a) they do not focus on detecting misbehavior online security forums or (b) they focus on identifying key players, threads, or events without an analysis from the perspective of hyperlink posting.

We briefly discuss the related works below.

a. Online security forum studies. This is a recent and less studied area of research. Some recent studies focus on identifying key actors and emerging concerns in security forums using supervised techniques and NLP by analyzing social and linguistic behavior [22], [23]. Some of these works are empirical studies that do not develop a systematic methodology. Recent efforts include analyzing the dynamics of black market hacking services [21], extracting malicious IP addresses reported by users [24], classifying threads given keywords of interest [4], and identifying communities of important hackers, events, and threats [6], [25], [20], [2], [26] in security forums.

Our work is different from these efforts in the sense that we focus on the identification of hyperlink-based misbehavior in a comprehensive way.

b. Hyperlink classification studies. There are only a few works that extract URLs from raw text [7], [8]. They focus on the age prediction of Twitter users by analyzing the contents in the URLs [7] and on extracting information from scientific paper PDF files [8]. These works place no emphasis on detecting misbehavior.

Most of the misbehavior detection works do not develop a systematic suite of capabilities. Instead, they focus on detecting only single type of misbehavior such as phishing. Many of these works are heuristic-based [16] and machine learning-based [17], [27], [28]. They exhibit good performance in terms of some selective performance metrics but tend to suffer from overfitting and/or low TPR and/or high FPR. CANTINA utilizes a bunch of heuristics and uses TF-IDF to gain the highest accuracy of 95% whereas [17] utilizes a DNN-based Stacking Model to gain an accuracy of 97.2%. Phishdef [29], on the other hand, utilizes only URL name-based features showing an average accuracy of 93%. Our proposed method, *RPhish*, is unique because it first compresses the features and then uses a NN to gain the best performance.

Very few works focus on detecting misbehavior from online platforms, and those that do not take the URL posting perspective like ours. These works focus on binary classification of misbehaving or benign user detection [30], deteriorating and non-deteriorating behavior prediction [31], or medical misbehavior of drug non-compliance detection [32]. None of the above-mentioned studies focus on systematic classification and identification of misbehavior from a hyperlink sharing perspective in online forums.

CONCLUSION

We propose and develop *LinkMan*, a comprehensive suite of capabilities to systematically identify URLs from text and URL-driven misbehavior from online security forums.

Our approach has the following main advantages: (a) it extracts URLs from raw text effectively with high precision and recall, (b) it classifies hyperlinks in an accurate and comprehensive way, and (c) it explores the entities and the collaborative behavior of users utilizing the power of tensor decomposition.

The current work is a building block to mine the wealth of information that exists in online forums. Follow-up efforts can use our approach to (a) monitor hacker activity, (b) detect emerging trends, and (c) identify influential hackers towards safeguarding the Internet.

ACKNOWLEDGEMENT

This work was supported by the UC Multicampus-National Lab Collaborative Research and Training (UCNLCRT) award #LFR18548554.

REFERENCES

- [1] "Estimated number of online forums and users," <https://quora.com/How-many-online-forums-are-in-existence>, [Accessed 2-June-2021].
- [2] R. Islam, M. O. F. Rokon, E. E. Papalexakis, and M. Faloutsos, "Tensor: A tensor-based tool to extract interesting events from security forums," in *2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2020, pp. 515–522.
- [3] "Hackers posing as mcafee antivirus," <https://pcmag.com/news/google-chinese-hackers-are-posing-as-mcafee-antivirus-to-phish-victims>, [Online; accessed 2-June-2021].
- [4] J. Gharibshah *et al.*, "REST: A thread embedding approach for identifying and classifying user-specified information in security forums." *ICWSM*, 2019.
- [5] "Online security forums," <https://offensivecommunity.net,https://hackthissite.org,https://ethicalhacker.net>, [Accessed 2-June-2021].
- [6] R. Islam, M. O. F. Rokon, A. Darki, and M. Faloutsos, "Hackerscope: The dynamics of a massive hacker online ecosystem," in *2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2020, pp. 361–368.
- [7] A. Pandya *et al.*, "On the use of urls and hashtags in age prediction of twitter users," in *IEEE IRI*, 2018, pp. 62–69.
- [8] R. Ahmad *et al.*, "Information extraction from pdf sources based on rule-based system using integrated formats," in *Semantic web evaluation challenge*. Springer, 2016, pp. 293–308.
- [9] "Popular online url extractor," <https://convertcsv.com/url-extractor.htm>, browserling.com/tools/extract-urls, [Online; accessed 2-June-2021].
- [10] S. Wold *et al.*, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1–3, pp. 37–52, 1987.
- [11] "Alexa web ranking," <https://www.alexa.com/siteinfo/>, [Online; accessed 2-June-2021].
- [12] "Hackerone: top 100 haking tools," <https://www.hackerone.com/blog/100-hacking-tools-and-resources>, [Online; accessed 2-June-2021].
- [13] "Tldextract package," <https://github.com/john-kurkowski/tldextract>, [Online; accessed 2-June-2021].
- [14] "Regular expression format," <https://en.wikipedia.org/wiki/URL/>, [Online; accessed 2-June-2021].
- [15] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: <https://archive.ics.uci.edu/ml/datasets/phishing+websites>
- [16] Y. Zhang *et al.*, "Cantina: a content-based approach to detecting phishing web sites," in *16th international conference on World Wide Web*. ACM, 2007, pp. 639–648.
- [17] Y. Li, Z. Yang, X. Chen, H. Yuan, and W. Liu, "A stacking model using url and html features for phishing webpage detection," *Future Generation Computer Systems*, vol. 94, pp. 27–39, 2019.
- [18] A. Banerjee *et al.*, "Sut: Quantifying and mitigating url typosquatting," *Computer Networks*, vol. 55, no. 13, pp. 3001–3014, 2011.
- [19] M. O. F. Rokon, R. Islam, A. Darki, E. E. Papalexakis, and M. Faloutsos, "Sourcefinder: Finding malware source-code from publicly available repositories in github," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses ({RAID} 2020)*, 2020, pp. 149–163.
- [20] R. Islam, M. O. F. Rokon, E. E. Papalexakis, and M. Faloutsos, "Recten: A recursive hierarchical low rank tensor factorization method to discover hierarchical patterns in multi-modal data," in *Proceedings of the International AAAI Conference on Web and Social Media*, 2021.
- [21] R. S. Portnoff *et al.*, "Tools for automated analysis of cybercriminal markets," in *International Conference on World Wide Web*, 2017, p. 657.
- [22] E. Marin *et al.*, "Community finding of malware and exploit vendors on darkweb marketplaces," in *ICDIS*. IEEE, 2018, pp. 81–84.
- [23] M. O. F. Rokon, P. Yan, R. Islam, and M. Faloutsos, "Repo2vec: A comprehensive embedding approach for determining repository similarity," in *2021 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2021.
- [24] J. Gharibshah *et al.*, "RIPEX: Extracting malicious ip addresses from security forums using cross-forum learning," in *PAKDD*. Springer, 2018.
- [25] R. Islam, M. O. F. Rokon, A. Darki, and M. Faloutsos, "Hackerscope: The dynamics of a massive hacker online ecosystem," *Social Network Analysis and Mining*, vol. 11, no. 1, pp. 1–12, 2021.
- [26] M. A. Jonas, M. S. Hossain, R. Islam, H. S. Narman, and M. Atiquz-zaman, "An intelligent system for preventing ssl stripping-based session hijacking attacks," in *MILCOM 2019-2019 IEEE Military Communications Conference (MILCOM)*. IEEE, 2019, pp. 1–6.
- [27] S. D. V. Prasad and K. R. Rao, "A novel framework for malicious url detection using hybrid model," *TURCOMAT*, pp. 68–76, 2021.
- [28] S. Deepa *et al.*, "Phishing website detection using novel features and machine learning approach," *TURCOMAT*, no. 7, pp. 2648–2653, 2021.
- [29] A. Le, A. Markopoulou, and M. Faloutsos, "Phishdef: Url names say it all," in *2011 Proceedings IEEE INFOCOM*. IEEE, 2011, pp. 191–195.
- [30] T. C. Li *et al.*, "Trollspot: Detecting misbehavior in commenting platforms," in *IEEE/ACM ASONAM 2017*, 2017, pp. 171–175.
- [31] J. M. Tshimula *et al.*, "On predicting behavioral deterioration in online discussion forums," in *IEEE/ACM ASONAM*, 2020, pp. 190–195.
- [32] É. Bigeard and N. Grabar, "Detection and analysis of medical misbehavior in online forums," in *SNAMS*. IEEE, 2019, pp. 7–12.