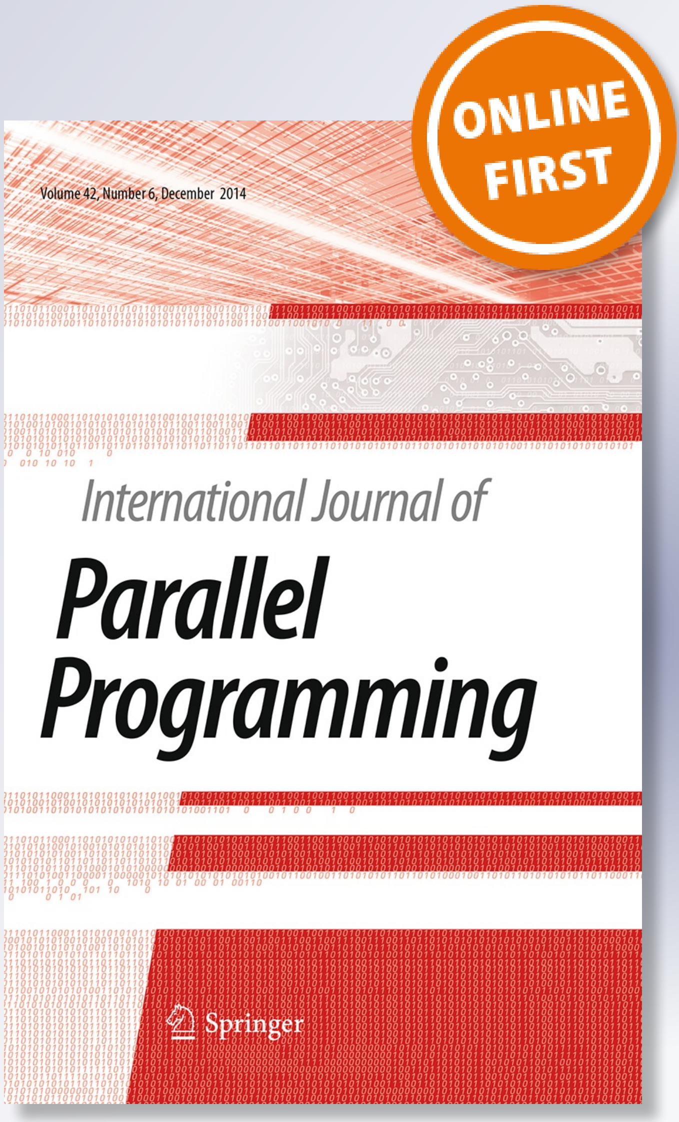
Detection of Forwarding-Based Malicious URLs in Online Social Networks

Jian Cao, Qiang Li, Yuede Ji, Yukun He & Dong Guo

International Journal of Parallel Programming

ISSN 0885-7458

Int J Parallel Prog
DOI 10.1007/s10766-014-0330-9



Springer

Int J Parallel Prog
DOI 10.1007/s10766-014-0330-9

Detection of Forwarding-Based Malicious URLs in Online Social Networks

Jian Cao · Qiang Li · Yuede Ji · Yukun He · Dong Guo

Received: 28 June 2014 / Accepted: 8 October 2014

Abstract In recent years, online social networks (OSNs), such as Facebook, Twitter and Sina Weibo, have become extremely popular among Internet users. Unfortunately, attackers also utilize them to hide malicious attacks. Due to the significance of detecting malicious URLs in OSNs, multiple solutions have been offered by OSN operators, security companies, and academic researchers. Most of these solutions use machine-learning methods to train classification models based on different kinds of feature sets. However, most are ineffective because their selected features are conventional. In this paper, we focus on forwarding-based features because of the special connections between forwarding behavior and the propagation of malicious URLs. First, we conduct a comprehensive analysis of conventional URL feature sets. Then, we design some forwarding-based features and choose several graph-based features to combine with them in order to train a detection model. We evaluate the system using about 100,000 original messages collected from Sina Weibo, which is the largest OSN website in China. The high accuracy rate and low false positive rate show that

J. Cao · Q. Li · Y. Ji · Y. He · D. Guo
College of Computer Science and Technology, Jilin University, Changchun, China
e-mail: caojian13@mails.jlu.edu.cn

Y. Ji
e-mail: jiyd12@mails.jlu.edu.cn

Y. He
e-mail: heyk12@mails.jlu.edu.cn

J. Cao · Q. Li (✉) · Y. Ji · Y. He · D. Guo
Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun, China
e-mail: li_qiang@jlu.edu.cn

D. Guo
e-mail: guodong@jlu.edu.cn

forwarding-based features are much more effective in detecting malicious URLs in OSNs than are other more conventional features. To the best of our knowledge, this work is the first to analyze forwarding-based features in OSNs and offers a valuable contribution to this area of research.

Keywords Online social network · Forwarding-based · Machine learning · Malicious URL

1 Introduction

Online social networks (OSNs), such as Facebook, LinkedIn, Twitter, and Sina Weibo, have become people's daily platform for communication and sense of community. College students may create friend-groups and share funny things with one another. Business professionals can communicate with their partners through online meetings, thus greatly improving efficiency. In China, Sina Weibo has become the most popular and largest OSN website. According to financial results from 2013 first quarter (Q1), the number of registered users for Sina Weibo is over 536 million and the number of active daily user has reached 49.8 million [1]. Sina Weibo has a huge influence on public topics and reflects the opinions of general users.

Unfortunately, attackers have turned their focus toward OSNs and are exploiting them by conducting phishing attacks [2–4], injecting malicious codes, spreading malware, and launching drive-by-download attacks [2,3,5]. These malicious behaviors can cause serious privacy and economic problems. Users' private data are popular on the black market and access to them may lead to economic crimes [5].

Due to the significance of detecting malicious URLs in OSNs, multiple solutions have been offered by OSN operators, security companies, and academic researchers. Most of the existing studies exploit machine-learning techniques to detect malicious URLs[6–12]; they attempt to build a detection model based on content or graph features [6,8]. There are also some interesting studies that rely on the statistical analysis of language and Markov Clustering [13,14]. Several of these methods have developed professional tools based on unique OSN platforms, such as PhishAir for Twitter and MyPageKeeper for Facebook [15].

However, battles between researchers and attackers are really fierce. If a detection tool has been employed, attackers can evade it within a short amount of time [6,8]. Existing studies focus keywords, conventional URL-based features, and graph-based features. Yet, attackers can change keywords or exploit unique, malicious URLs to evade URL- and content-based detection methods [6,8]. Shortened URLs also decrease the detection rate since several methods focus only on the original URL. Although attackers can change their symbols to evade existing methods, neither their position in the whole graph nor the forwarding-based behaviors can be easily changed.

In this paper, we have designed several forwarding-based features and also use some URL- and graph-based features from existing studies to build a detection model. The retweet in Twitter and the forward in Sina Weibo are typical forwarding-based behaviors. Malicious URLs are a kind of URL that can redirect to malicious web pages or conduct malicious behavior on the victim account, such as following a strange account and automatically forwarding a message that contains the malicious URL. To build and evaluate our system, we have collected about 100,000 original messages from Sina Weibo. From those, we retrain about 12,006 messages that contain URLs and forwarding-behaviors. We then labeled malicious URLs using the URL datasets Google Safe Browsing and honeypot [16,17]. In order to prove the effectiveness of our feature sets, we apply them on datasets and receive higher accuracy and lower false positive rates than other conventional methods.

In summary, our contributions are as follows:

(1) We propose that forwarding behaviors in OSNs can expand and accelerate the propagation of malicious URLs because forwarding messages contain original URLs. Victims are more likely to click on malicious URLs if the messages are sent from their friends.
(2) We design several forwarding-based features, and also use some URL- and graph-based feature sets, to detect malicious URLs. To the best of our knowledge, this is the first work to analyze forwarding-based features in OSNs and offers a valuable contribution to this area of research.
(3) We have collected more than 100,000 original Sina Weibo messages to evaluate the effectiveness of forwarding-based features. Through the experiment, we received higher accuracy and lower false positive rates.

The rest of this paper is organized as follows. We first discuss related works in Sect. 2. We then provide the datasets we collected from OSNs and offer a system overview in Sect. 3, introduce several conventional features and propose forwarding-based features in Sect. 4, and evaluate the accuracy and false positive rates in Sect. 5. In Sect. 6, we discuss limitations and future work, and, finally, in Sect. 7, we offer our conclusions.

2 Related Work

Malicious URL is an important Internet security issue; they have become the main way to conduct phishing attacks, inject malicious codes, spread malware, and launch drive-by-download attacks. Attackers embed different modes of attack in the web pages to which malicious URLs direct [2,3], and once the victim clicks on a malicious URL, they are taken to that web page without notice. Then the attacker may steal any of the victim's information that is saved on the local host, which may lead to serious economic loss. When malicious URLs are sent by friends, victims are more likely to click them.

Since malicious URLs have become a challenge to web security, more and more detection methods for conventional malicious URLs and malicious web pages have been published. Conventional detection methods include blacklists and signatures as well as training detection models using machine learning with conventional features [2,13,18,19]. The blacklist method needs to compare the target URL with blacklist databases in either a local or remote server, yet blacklist datasets are always large and cannot be updated as soon as new malicious URLs come out. Thus, this method may overlook the latest malicious URLs and cause delays because of the time required to compare individual URLs to the whole dataset [2,6,8]. Both static and dynamic methods are used when dealing with the conventional features of malicious URLs and redirected pages [2,3]. Static method features are URL- and content-based [2]; these features can directly extract from the message and URL themselves, collecting such information as the length of the URL. Dynamic method features regard the behavior values created by malicious URLs running in a controlled environment [2,18,19] such as a simulation browser or virtual machine environment. Malicious URLs may lead to phish sites or create malicious threads in a simulation browser [2,3,6] and attackers have ways of disguising malicious URLs as benign in order to evade static detection methods [6,8]. However, dynamic detection methods may fail to detect interactive attacks because of the triggers used, such as mouse clicks or input.

As malicious URLs have become a greater threat to OSNs, three main methods for dealing with them in OSNs have been used: (1) Apply conventional detection methods with little modification [9–12]. However, such detection methods just directly extract URL- and content-based features, ignoring features unique to OSNs; for example, these methods are unable to handle shortened URLs, which are common in OSNs. (2) Combine a few account-based features from OSNs with selected conventional features [7,8]. These kinds of detection methods consider the unique features of OSN accounts or messages, such as the number of followers, the number of followings, and the ratio between the two. However, these detection methods have only a few OSN features; the bulk of their features come from conventional methods [12]. (3) Focus on OSN behaviors and analyze graph-based features[6]. Graph-based features include the number of followers in the entire OSN graph as well as the cluster size of the malicious account group [6,11]. This detection method relies on the structure of the unique OSN and connects with the special OSN platform.

3 Dataset and System Overview

In this section, we first describe how we collected data both for analysis and for building a true positive dataset of Sina Weibo messages containing malicious URLs. Then, we will introduce, in detail, the detection model we build based on machine learning.

3.1 Dataset

In order to collect messages containing the original URL as well as forwarding-based features, we try to crawl them from Sina Weibo. Sina Weibo is a popular OSN in China and attracts many attackers' eyes. It only takes a click to forward Weibo messages in such a way that the original URL can be seen by public users. We have developed a crawler based on Weibo API [20]. Over a long period of time, the crawler has collected about 100,000 original Weibo messages. To detect malicious URLs, we filter out about 12,006 Weibo messages containing URLs. We then label malicious and legitimate URLs on our dataset.

Because all URLs are shortened before being published, we need to first transform them back to the original URL. To deal with this problem, we have developed a short-URL-transform script. We exploit Google Safe Browsing and URL honeypot in order to label malicious URLs. Google Safe Browsing runs fast but is likely to miss

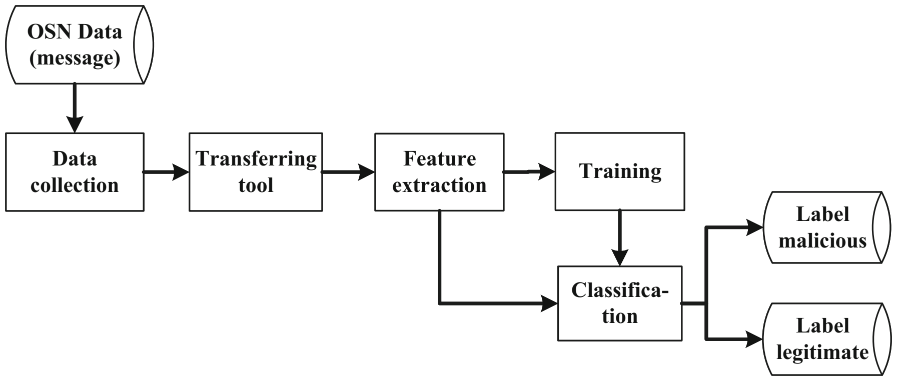


Fig. 1 System overview

some of the latest URLs. To compensate for this lack, we also use URL honeypot. We may label a URL as malicious if some malicious behaviors have been created and damaged a victim's computer security. Through this method, we have built a malicious dataset based on 12,006 Weibo messages. The whole dataset is divided into two parts; one for training a detection model and the other for evaluating the detection model.

3.2 System Overview

For clear and effective comprehension of the detection mechanism, we have designed a circuit model based on the detection process (see Fig. 1). The detection system has several modules on different functions. Our detection system has original data collection, shortened URL transformation, feature extraction, training and classification modules. The style of this detection system is similar to some existing studies in that the machine-learning mechanism contains most of these modules.

3.2.1 Data Collection

The data collection module utilizes the methods mentioned above to collect original messages in OSNs. Message containing URLs will be picked out as we focus on the malicious URLs' propagation behaviors. We collect useful information from the original dataset, such as number of forwards, URLs in the message, and so on. The new dataset, containing all the useful information, will become the one we use to analyze URLs. All other functions will focus on this dataset.

3.2.2 Shortened URL Transformation

Because of character limitations in OSN messages, URLs need to be shortened before message are sent. In order to analyze URL-based features and confirm the validity of the original URL, we need to transform the shortened URL back to its original form. As mentioned above, we have designed a transformation tool based on python. The transformation tool's engine is learned from existing methods' documents. After transforming shortened URLs back to their original form, we send them to the feature extraction module.

3.2.3 Feature Extraction

Features are the main point in machine learning. We have designed several forwarding-based features and combine them with several graph- and URL-based feature sets in order to train a detection model.

Please note that URLs appearing in the next several modules are the original ones that have been transformed from their shortened form in the previous module. First, we analyze the URL-based features; these may include the length of a URL's redirection chain and the presence of conditional redirects. Then, we extract the forwarding-based features from the message. To deal with the number of forwards, the number of comments, or the number of @ accounts associated with a forwarded message, we need to focus on both the message and the account. All of these basic features need to be extracted with some sort of mathematical operation in order to calculate the forwarding-based feature's value. We should also extract graph-based features, which are related with certain accounts' followings and followers. After extraction, we will transfer the value set to the next module.

3.2.4 Training

The training module will create a malicious detection model based on a machine-learning algorithm. In a related work [2], the BINSPECT method is applied to almost every mechanism of machine learning. We use the Bayes net, J48, and random forest to train a detection model; three of these machine-learning strategies are frequently used in existing studies [2,3,8,9]. We run these strategies several times with the same dataset and then compare the results to each other until we finally get a valuable model with high accuracy and low false positive rates.

3.2.5 Classification

After the training module, a model for detecting malicious URLs is finally produced. The model will be used to classify strange URLs in other messages. When a new message containing URLs appears in the queue of a user's webpage, the target URLs will first be compared with local blacklists. If there is no answer, the detection model will pick out basic features and analyze them. After extraction, the features will be transferred into the classification module, instead of the training model, since the model has already been produced. When the detection model has labeled a URL, a sign meaning either benign or malicious will show on the local screen.

4 Features

OSNs have their own graphic structures and connections between users could reflect the whole OSN's state. If there is a line between two users on the OSN's graph, those users can communicate with each other. Malicious URLs will not propagate to others if the user point is isolated. Thus, it is necessary to analyze graphic structure changes among different OSN points when there has been some suspicious behavior. We focus our study on forwarding behavior and concentrate on analyzing graph-, URL-, and forwarding-based features. Our feature sets contain three parts: URL-based, forwarding-based and graph-based.

4.1 URL-Based

The URL-based features in our detection model are selected from existing studies and can reflect the natural features of original URLs. Our URL-based sets include two features that focus on the redirection of original URLs: the length of the URL redirection chain and the presence of conditional redirects.

The length of the URL redirection chain is one of the most popular features regarding detection of malicious URL redirections. If the URL is a legitimate one, it will just jump to the destination from one or two top-level domain websites. However, attackers may exploit legitimate-like URLs to lure victims and evade detection through the use of conventional methods. Legitimate-like URLs will jump to the destination, which is a malicious web page, from various numbers of point. If the length of the redirection chain is huge, we find it highly probable that the URL is a malicious one.

The conditional redirection phenomenon has been proposed in related work [7]. URL redirection chains may be long, but the important factor regarding conditional redirections is the decision point. The decision point will be redirected to malicious pages if the current environment is safe, which means that the current host has no detection thread; otherwise, it will jump to a legitimate search engine or some other page to evade detection tools in the current environment. We adopt methods from the related work [7] to define conditional redirects. If the redirection chain contains conditional redirection behavior, we find it highly probable that the URL is a malicious one.

4.2 Forwarding-Based

We focus on what happens when users forward the message, paying special attention to whether attackers can easily change the position that they existed in the whole network. If you want to send a private message to another user or inform that user about a previously sent message, you use an signal "@" in the message in OSNs. Our forwarding-based sets include three features: forward-comment-ratio, forward-@-following-ratio, and forward-follower-ratio.

Forward-comment-ratio When messages are forwarded, legitimate messages and malicious messages perform differently. If the message is legitimate, the number of comments is similar to the number of forwards. Most legitimate users like to discuss the

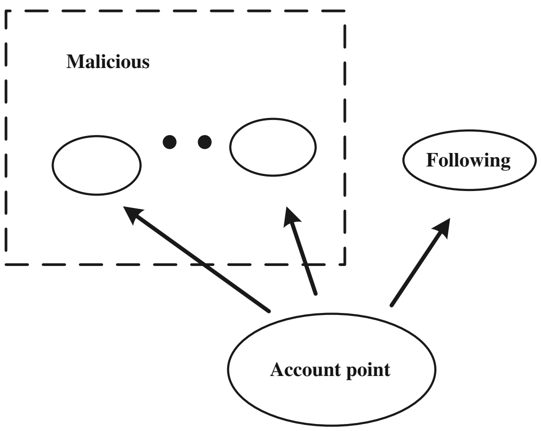


Fig. 2 Forward-@-following-ratio

forwarded message. Malicious accounts just forward the message containing malicious URL without comment as mentioned before, URLs from malicious accounts will be labeled as malicious. This feature is defined below:

$N_{forward}$:	The number of forwards for the forwarded message
$N_{comment}$:	The number of comments for the forwarded message

$$F_{forward-comment} = (N_{forward} - N_{comment})/N_{forward} \tag{1}$$

The dataset shows that a legitimate message may have comments without forwards. If $N_{comment}$ is larger than $N_{forward}$, the message will be regarded as legitimate. We define that the value of $F_{forward-comment}$ tends toward 1 if the URL is malicious.

Forward-@-following-ratio Attackers need to propagate malicious URLs as soon as possible because messages containing malicious URLs may be reported to an OSNs' administrator at any time. We focus on the moment malicious URLs are widely forwarded and propagated. If a victims' account is infected by a malicious URL, it will forward the malicious message using "@" to all of its followings. The malicious URL can easily propagate because of the trust-connection between users, which is illustrated in Fig. 2. A legitimate user will most often only "@" someone who has a connection with the message. This feature is defined below:

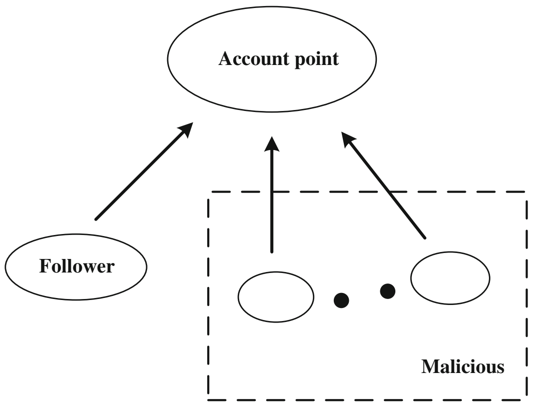
$N_{forward-@}$:	The number of "@" used with the forwarded message. The accounts associated with each @ are linked to the forwarding account (e.g., by following).
$N_{following}$:	The number of accounts following the forwarding account.

$$F_{forward-@-following} = N_{forward-@}/N_{following} \tag{2}$$

We know that the message containing malicious URLs will be propagated to almost everyone in the following list of its sender. The $N_{forward-@}$ will quickly equal $N_{following}$. We define that the value tends toward 1 if the URL is malicious.

Forward-follower-ratio: If OSN accounts have been infected by malicious accounts, they will propagate to the groups to which they belong. After several infections,

Fig. 3 Forward-follower-ratio



these groups will turn malicious themselves. URLs sent from these groups should be regarded as malicious. We focus on malicious group behaviors that occur when malicious URLs are forwarded. A legitimate group member may forward only a few interesting messages, such as interesting movie, cheerful song, or a positive quote. However, as we can see in Fig. 3, each member of a malicious group will forward the malicious message in order to propagate it as soon as possible. This feature is defined below:

$N_{forward-follower}$:	The number of followers that have forwarded the message and the followers belong to the forwarding account.
$N_{follower}$:	The number of followers of the forwarding account.

$$F_{forward-follower} = N_{forward-follower}/N_{follower} \tag{3}$$

As mentioned above, if almost all of the followers of the forwarding account forward the same message, the URL in the message tends to be malicious. We define that the value tends toward 1 if the URL is malicious.

4.3 Graph-Based

OSNs have their own structures and the features based on their graphs are usually influenced by each individual OSN's behavior. These behavior models cannot easily be modified. We can focus on these natural factors to detect malicious URLs. Because each graph is produced by natural behaviors based on an OSN user's purpose, fake behaviors can easily be picked out. In this section, we discuss two graph-based features: following-follower ratio and local clustering coefficient.

Following-follower ratio As mentioned in related work and other existing research, the ratio between a legitimate account or number of followings, and number of followers is relatively low. We have observed that a legitimate ratio is around one value, while the ratio for a malicious account may be several times that. The malicious account may follow random accounts in order to propagate more. Most victims add malicious accounts without paying much attention. Legitimate accounts have almost the same number of followings and followers, because the followings and followers of legitimate accounts are generally actual friends, coworkers, and family in real life. Meanwhile, a malicious account may follow other accounts randomly upon meeting a stranger. This feature is defined below:

$N_{following}$:	The number of followings of the forwarding account
$N_{follower}$:	The number of followers of the forwarding account

$$F_{following-follower} = N_{following}/N_{follower} \tag{4}$$

Malicious accounts have smaller numbers of followers than they do numbers of followings; thus, the ratio will be higher. Legitimate accounts usually have similar numbers of followings and followers, so their ratio tends towards 1. We define higher ratios as more likely of having malicious URLs.

The local clustering coefficient The local clustering coefficient reflects the similarity and connections among points in the same group. The friends of legitimate accounts are in the same groups, and connections among them are strong. However, malicious accounts' followings or followers often do not have such close connections. We can define the formula as its original standard:

e^{jk}:	the number between the current point and its neighbors
$K_i(K_i - 1)$:	the possible number of edges between the current point and its neighbors.

$$C_i = 2|e^{jk}|/(K_i(K_i - 1)) \tag{5}$$

We know that the connection will be close if the point is a legitimate one, and the value will tend toward 1. We define the URL as malicious if the value is lower.

5 Evaluation

In this section, we first introduce the confusion matrix. Then, we analyze the effectiveness of forwarding-based features, which has two aspects: the universality of forwarding-based behaviors in Weibo, and the obvious improvement of detection performance when forwarding-based features are used. Finally, in order to evaluate the performance of our detection system, we gradually increase our dataset to observe any changes and obtain a blacklist to compare with our system. We discuss results and analyze potential influence factors with each of these steps.

5.1 Confusion Matrix

We use several evaluation metrics to comprehensively evaluate our prototype, including false positive, F-measure, accuracy, and so forth. The confusion matrix presents the relationship between true positive, true negative, false positive, and false negative as shown in Table 1.FP refers to false positive; its value represents how often benign URLs have been falsely judged as malicious by our system. F-measure is a normal

Table 1 Confusion matrix

Predicted			
Actual		Malicious	Benign
	Malicious	TP	FN
	Benign	FP	TN

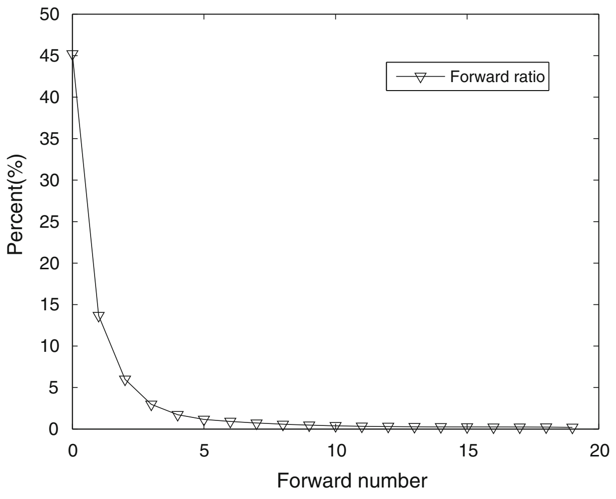


Fig. 4 Forwarding ratio

evaluation standard in the IR field. The higher the F-measure, the better the system's performance. Formula 6 shows the inter-connection among the matrix.

$$\begin{aligned} (R)Recall &= TP/(TP+FN) \\ (P)Precision &= TP/(TP+FP) \\ F-measure &= (\beta^2+1)PR/(\beta^2 P+R) \\ Accuracy &= (TP+TN)/(TP+TN+FP+FN) \end{aligned} \tag{6}$$

5.2 Effectiveness of Forwarding-Based Features

To verify whether forwarding messages is common, we first have to collect statistics regarding the number of times a message is forwarded. We then draw a figure to visually portray the likelihood of a message being forwarded when it contains a URL. As shown in Fig. 4, forwarding message is obviously common in OSNs. As shown in Table 2, when a message has been forwarded between 4,001 to 1,847,151 times, its forwarding ratio is still 10.01 %, which shows that studies on forwarding behavior is necessary and our focus on forwarding-based features is truly important and right. We also analyze the forward-comment feature on our dataset. As shown in Fig. 5, there is an obvious correlation between the number of comments and the number of forwards. As we have mentioned before, legitimate accounts have a similar number of

Table 2 Forwarding ratio

Forward numbers	Ratio (%)
0	31.80
1–4,000	58.19
4,001–1,847,151	10.01

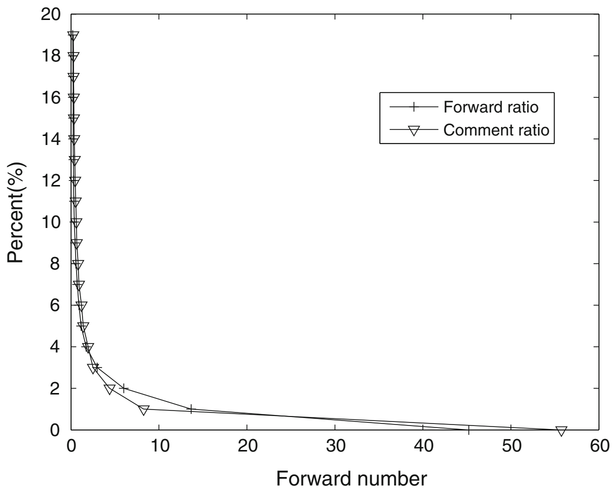


Fig. 5 Forward comment ratio

forwards and comments. Still, even when the number of comments has been almost 0, the likelihood that the message has been forwarded more than a hundred times is still quite high.

To prove the effectiveness of the forwarding-based features we designed, we compare the system with and without forwarding-based features. Table 3 shows the complete feature sets that we have used to train our model. We trained the dataset using three machine-learning methods: Bayes net, J48, and random forest. Figure 6 shows average-value comparison results, looking at the accuracy rate, the false positive rate, and the F-measure; both the accuracy rate and F-measure are obviously increased when the model is trained with forwarding-based features, and the false positive rate decreases comparatively. The average accuracy rate is about 83.21 %, and average false positive rate is about 10 %.

The comparison shown in Fig. 6 obviously show the effectiveness of the forwarding-based features we designed. Common forwarding behaviors make propagation much faster and more widespread. However, if we cut the propagation line used by malicious URLs, we can stop propagation in time and identify malicious accounts in order to support further detection.

5.3 System Evaluation

In this section, we evaluate our detection system from three aspects. First, we train a classifier model and test it using the dataset we built according to the previous

Table 3 Feature sets

Without forward	Length of URL redirection chain	Number of redirection chains from the shortened URL to the final destination URL
	Conditional redirection	Whether the landing URL is different based on the loading environment
	Following-follower ratio	The ratio between the number of followings and followers of a user count
	The local clustering coefficient	The similarity and connection between points in a local group
With forward	Forward-Comment ratio	The ratio between the number of forwards and comments of one message
	Forward-@-Following-ratio	The ratio between the number of forwards and the people called out in the message. The people called out belong to following accounts.
	Forward-Follower-ratio	The ratio between the number of forwards and the number of followers of the original forwarding account

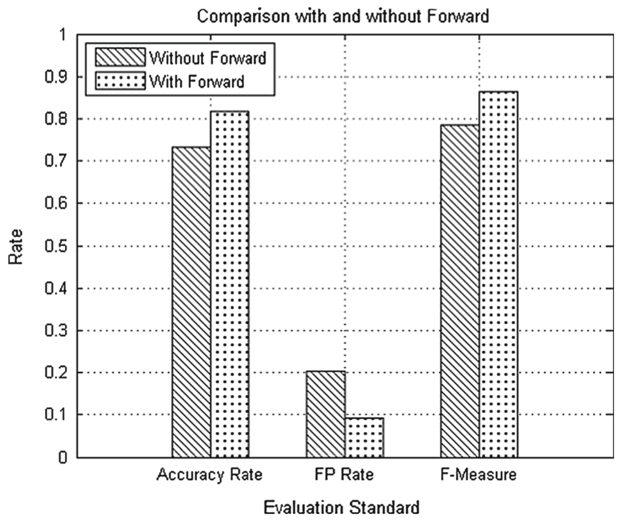


Fig. 6 Comparison with and without forward

sections. In order to verify the robustness and balance of our system, we then re-train the classifier with ten-fold cross-validation within certain machine-learning methods by increasing samples. Finally, we compare our system with the blacklist method.

5.3.1 Train and Test a Classifier

As mentioned before, we built a dataset to train a detection model and test it. We collected 12,006 samples that have both forwarding behaviors and shortened URLs. There are 4,426 samples containing malicious URLs. The other 7,580 samples are

Table 4 System evaluation

Machine learning methods	Accuracy (%)	FP rate (%)	F-measure
Bayes net	84.74	9.09	0.83
J48	82.22	10.23	0.88
Random forest	79.07	10.77	0.84

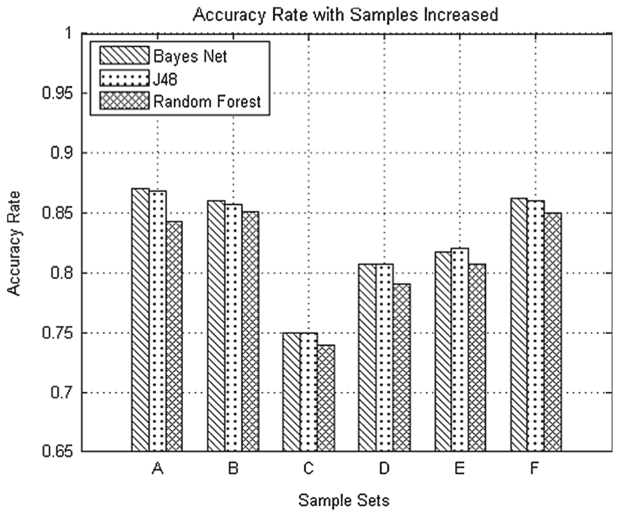


Fig. 7 Accuracy rate with samples increased

benign. All of the samples have been divided into two parts using a 7:3 ratio. The larger part of each sample is used to train the model and the smaller part is used to test the model. We also train the model using ten fold cross-validation within certain machine-learning methods. The final reported accuracy rate, false positive rate, and F-measure are all averaged values because we use three machine-learning methods: Bayes net, J48, and random forest. As shown in Table 4, the average accuracy rate is 83.21 %, the average false positive rate is 10.03 %, and the average F-measure is 0.86.

5.3.2 Train a Model with Increased Samples

In order to verify the robustness and balance of our system, we try to observe system changes by increasing training samples. We have divided the samples into six parts: A, B, C, D, E, and F, as shown in Fig. 7. Samples are increased by 2,000 each time: malicious and benign sub-samples are increased by a fixed ratio and as a new set. We still train the model with three machine-learning methods: Bayes net, J48, and random forest. The setting used in the machine-learning methods is ten fold cross-validation. Accuracy rate fluctuations of the system can be seen in Fig. 7. The false positive rate is shown in Fig. 8. Figure 9 shows the change of F-measure. Due to the relationships between these figures, we can talk about their fluctuations together.

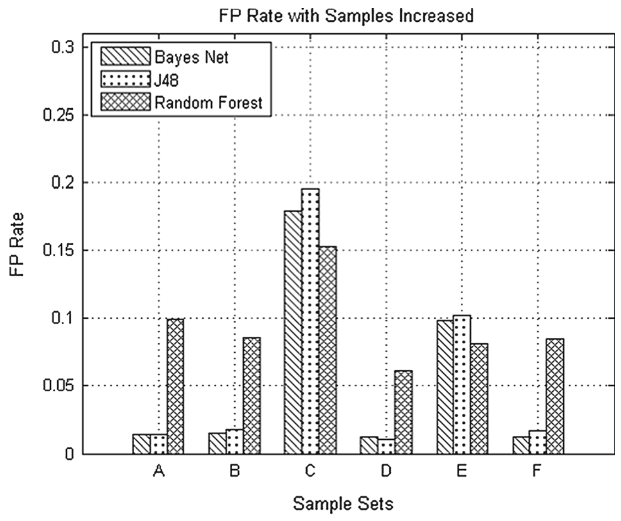


Fig. 8 FP rate with samples increased

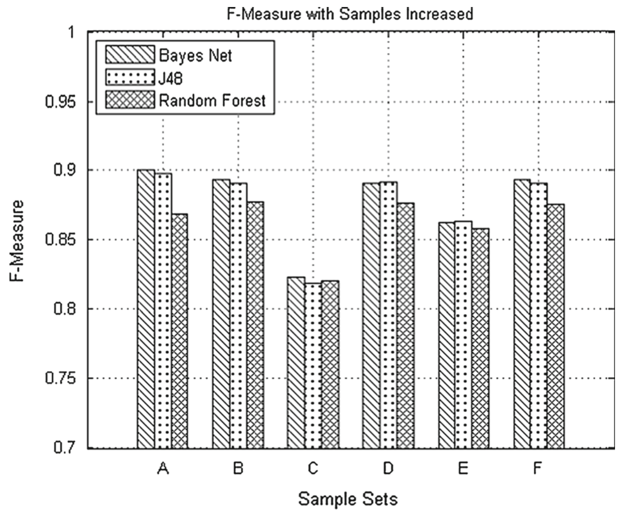


Fig. 9 F-measure with samples increased

We see an obvious decrease in the average values shown in Figs. 7 and 9 (accuracy rate and F-measure) when sample B increases to C, and we see an associated increase in the average value shown in Fig. 8 (false positive rate). The suddenly change in sample C makes us analyze potential influences. Because the values continue to increase as the samples continue to increase beyond sample C, we turn our attention to the samples that have increased. After analyzing the increased samples, we find that the fluctuation range of forwarding behaviors has grown. The samples do not show a regularity of change in forwarding behaviors. In other words, the dataset is still too small to train a robust model. Thus, when the samples increased to much larger ones, the values of different attributes came to a stable state as shown in Figs. 7, 8 and 9. Additionally, we propose another potential factor of influence. The malicious behaviors of attackers may have experienced a period of explosive growth. The increased samples from B to C may have just come at the lowest point of the cycle of malicious behavior.

Although the figures reveal a sudden fluctuation in the training process, the performance state does become stable. Thus, our system is still a robust one that is effective in detecting malicious behavior.

5.3.3 Comparison with Blacklist

As we cannot totally re-implement existing models, we merely compare the efficiency of our system with that of the blacklist method. For comparison, we sampled another 2,110 messages. All of the messages were submitted in a matter of minutes and verified by ourselves. Of the 2,110 samples, 821 are malicious; the rest are benign. To compare efficiencies, we only analyze the accuracy rate and false positive rate of our detection system and of the blacklist.

The detection system can verify 649 malicious samples and 1028 benign samples. The accuracy rate is about 80 %, and the false positive rate is about 12 %. However, the accuracy rate of the blacklist is less than 60 %. We believe the reason for this difference is that the blacklist needs time to update its dataset; new malicious URLs have not yet been included in the blacklist. The shortened URLs found in OSNs, especially, require time for the blacklist to transfer and include. However, our system can detect malicious URLs in OSNs in time to cut off their propagation routes and, thus, protects users to the fullest extent.

6 Limitations and Future Work

As we can see from this experiment, although the forwarding-based features we have designed do improve the detection of malicious URLs, our system still has several limitations.

The original dataset is always a challenge when it is time to handle larger amounts of data. We have collected a considerable number of original Sina Weibo messages, where the data is platform-limited. Because Facebook, Twitter, and LinkedIn have different platform types than Sina Weibo, the current dataset may need to expand in order to work with those platforms. However, big-data collection is also a challenge for data mining. Additionally, our evaluation needs to be more comprehensively compared with existing studies, which is also a challenge: repeating the detailed work of others' usually has some level of discrepancy. In this paper, we focus on the malicious URLs found in messages, and the selected features used would need to be modified in order to contain other aspects of forwarding behavior.

Much time is still needed to crawl and analyze other platforms' original data. Twitter and Facebook data will be covered in future work in order to have a complete analysis of OSN forwarding behavior. To complete the comparative experiment with other existing work, we will also need to confer with the corresponding authors and re-implement all of their detailed work. By completing all these tasks, our system will become much better.

7 Conclusion

Online social networks (OSNs) have become peoples' daily platform for communication and sense of community. The forwarding-based features in OSNs accelerate messages' propagation speed and range, and attackers utilize the trust and convenience found among OSN accounts to propagate malicious URLs contained in messages. In this paper, we focus on forwarding-based behaviors to train a detection model for malicious URLs. These behaviors are natural produced by the forwarding operation. Evaluation shows that our method has higher accuracy and lower false positive rates. We would like to complete our study in the future work previously mentioned. We would sincerely like to communicate with other researchers working this area.

Acknowledgments This work is supported by the National Natural Science Foundation of China under Grant Nos. 61170265 and 61472162.

References

1. Seeking alpha: Sina corporation's ceo discusses q1 2013 results - earnings call transcript. http://seekingalpha.com/article/1442711-sina-corporations-ceo-discusses-q1-2013-results-earnings-call-transcript, Accessed Dec 2013
2. Eshete, B., Villafiorita, A., Weldemariam, K.: Binspect: holistic analysis and detection of malicious web pages. In: Security and Privacy in Communication Networks, pp. 149–166. Springer (2013)
3. Eshete, B., Villafiorita, A., Weldemariam, K.: Einspect: Evolution-guided analaysis and detection of malicious web pages. Technical report, Fondazione Bruno Kessler (2012)
4. Aggarwal, A., Rajadesingan, A., Kumaraguru, P.: Phishari: automatic realtime phishing detection on Twitter. In: eCrime Researchers Summit (eCrime), 2012, pp. 1–12. IEEE, (2012)
5. Rahman, M.S., Huang, T.-K., Madhyastha, H.V., Faloutsos, M.: Efficient and scalable socware detection in online social networks, In: USENIX Security (2012)
6. Yang, C., Harkreader, R.: Empirical evaluation and new design for fighting evolving Twitter spammers. IEEE Trans. Inf. Forensics Secur. **8**(8), 1280–1293 (2013)
7. Lee, S., Kim, J.: Warningbird: detecting suspicious urls in Twitter stream. In: Symposium on Network and Distributed System Security (NDSS) (2012)
8. Gao, H., Chen, Y., Lee, K., Palsetia, D., Choudhary, A.N.: Towards online spam filtering in social networks, In: Symposium on Network and Distributed System Security (NDSS) (2012)
9. Xiang, G..: Toward a phish free world: a feature-type-aware cascaded learning framework for phish detection. PhD thesis, Carnegie Mellon University, (2013)
10. Wen, S., Zhou, W., Zhang, J., Xiang, Y., Zhou, W., Jia, W.: Modeling propagation dynamics of social network worms. IEEE Trans. Parallel Distrib. Syst. **24**(8), 1633–1643 (2013)
11. Egele, M., Stringhini, G., Kruegel, C., Vigna, G.: Compa: detecting compromised accounts on social networks. In: NDSS (2013)
12. Lam, K.C., Lau, W.C., Yue, O.: Hitchbot-delivering malicious urls via social hitch-hiking. In: Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE, pp. 1–6. IEEE, (2011)
13. Martinez-Romo, J., Araujo, L.: Detecting malicious tweets in trending topics using a statistical analysis of language. Expert Syst. Appl. **40**(8), 2992–3000 (2013)
14. Ahmed, F., Abulaish, M.: An mcl-based approach for spam profile detection in online social networks. In: IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 , pp. 602–608. IEEE, (2012)

15. Rahman, M.S., Huang, T.-K., Madhyastha, H.V., Faloutsos, M.: Frappe: detecting malicious facebook applications. In: Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies, pp. 313–324. ACM, (2012)
16. Google safe browsing api. https://developers.google.com/safe-browsing/?hl=zh-CN, Accessed Dec 2013
17. Honeypot. http://old.honeynet.org/, Accessed Dec 2013
18. Egan, S., Irwin, B.: An evaluation of lightweight classification methods for identifying malicious urls. In: Information Security South Africa (ISSA), 2011, pp. 1–6. IEEE, (2011)
19. Ma, J., Saul, L.K., Savage, S., Voelker, G.M.: Learning to detect malicious urls. ACM Trans. Intell. Syst. Technol (TIST) **2**(3), 30 (2011)
20. Sina weibo api. http://open.weibo.com/, Accessed Dec 2013