

# Anomaly Based Malicious URL Detection in Instant Messaging

D. J. Guan<sup>1</sup>, Chia-Mei Chen<sup>2</sup>, and Jia-Bin Lin<sup>1</sup>

<sup>1</sup> Department of Computer Science and Engineering

<sup>2</sup> Department of Information Management

National Sun Yat-Sen University, Kaohsiung, Taiwan

**Abstract.** Instant messaging (IM) has been a platform of spreading malware for hackers due to its popularity and immediacy. To evade anti-virus detection, hacker might send malicious URL message, instead of malicious binary file. A malicious URL is a link pointing to a malware file or a phishing site, and it may then propagate through the victim's contact list. Moreover, hacker sometimes might use social engineering tricks making malicious URLs hard to be identified. The previous solutions are improper to detect IM malicious URLs in real-time. Therefore, we propose a novel approach for detecting IM malicious URLs in a timely manner based on the anomalies of URL messages and sender's behavior. Malicious behaviors are profiled as a set of behavior patterns and a scoring model is developed to evaluate the significance of each anomaly. To speed up the detection, the anomaly behavior patterns can identify known malicious URL features efficiently, while the scoring model is used to detect unknown malicious URLs. Our experimental results demonstrate that the proposed approach can effectively detect IM malicious URL with 0.93% false positive rate and 0.37% false negative rate on average.

**Key words:** Instant Messaging, Malicious URL, IM Worms

## 1 Introduction

Instant messaging (IM) is an internet application that used daily by most people to communicate with each other. It provides real-time messaging and presence awareness. There are also many additional features such as files transferring, video chatting, VoIP, and games. Due to the prevalence and immediateness of IM, it has become a platform for hackers spreading malware through social engineering. Moreover, the contact list becomes a shortcut for attacking more victims without finding any vulnerable machines. By advantage of IM features, hackers can spread malware quickly and increase the successful probability of attacks. This is why IM incurred the security threats.

Hackers have two mainly attack methods: (1) issue a file transfer, or (2) sending message with a link. File transfer injects malware directly to an IM user. Once the IM user downloads and executes this malware, the user is then

compromised, and the malware will be propagated thru the victim's contact list. To evade malware detection, hacker may send message with malicious URL pointing to either a malware or a phishing site. The current anti-virus software is unable to detect such IM malicious URLs. Therefore, we propose a solution on malicious URLs detection in IM environment.

IM malicious URL is usually a link to malware or a phishing site stealing account and password. Once an IM user clicks the link to malware or types account and password on a phishing site, the user subsequently is compromised and send the malicious URL to users of contact list by malware or bots. A real-time detection of malicious URL is desirable to alert a targeted IM user for such dangerous links or malicious websites and to prevent further infection and damage through social engineering. However, the current study has not yet proposed a real-time detection for identifying IM malicious URLs. Therefore, in this paper, we propose an approach to detect IM malicious URLs in real-time. Our approach is based on the anomalies of URL messages and sender's behavior in client side. We define some known malicious behavior to check whether a URL message is malicious. When a URL message does not match the behavior patterns, we will use a score model to predict the URL is malicious or not. The experimental results demonstrate that the effectiveness of detection is acceptable.

In section 2, we discuss related work. We describe the detail of the proposed approach in section 3. Experimental evaluations are presented in section 4. Section 5 concludes this paper.

## 2 Related Work

Previous work about IM worms, [5, 7] presented schemes to throttle the spread of IM worms by limiting transmission rates. But they also degrade the usability of IM and unable to detect IM worms in real-time. HoneyIM [8] uses IM decoy users to detect the existence of IM malware in an enterprise-like network. When the IM decoy user receive a file or a URL message, it represents the sender that has a decoy user in his contact list is infected. However, as long as IM worms do not yet propagate to the user that have a decoy account in his contact list, IM worms do not be detected.

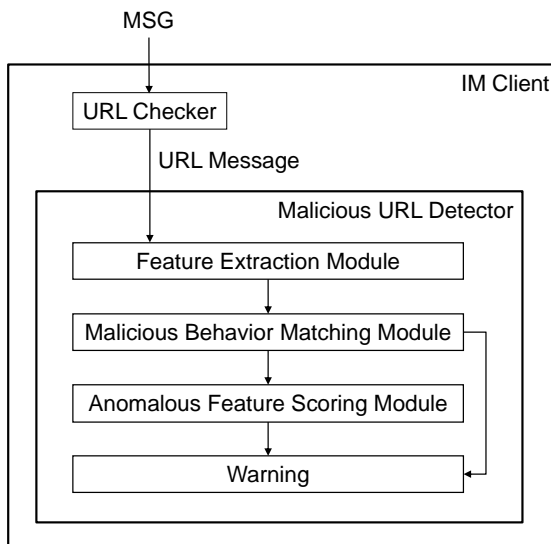
Another chat service, public chat room, exists chat bots to spread spam URLs or malware links. [1] observes the inter-message delay and message size between human and chat bots in Yahoo! chat room. They found that some type of chat bots have regular behavior, and the behavior of human are more complex. Therefore, they proposed an entropy classifier and a machine learning classifier to distinguish chat bots from human. Because public chat room is different from IM environments that usually chat one-on-one. This approach is not enough to apply in IM environments.

Although IM phishing URLs steal the IM account and password which is unlike conventional phishing sites, abnormal URL features are common. Phishing sites detection have been study a lot of phishing URL features [3, 4, 9]. We

apply some usable features and new features we observed to IM malicious URLs detection.

### 3 Proposed Approach

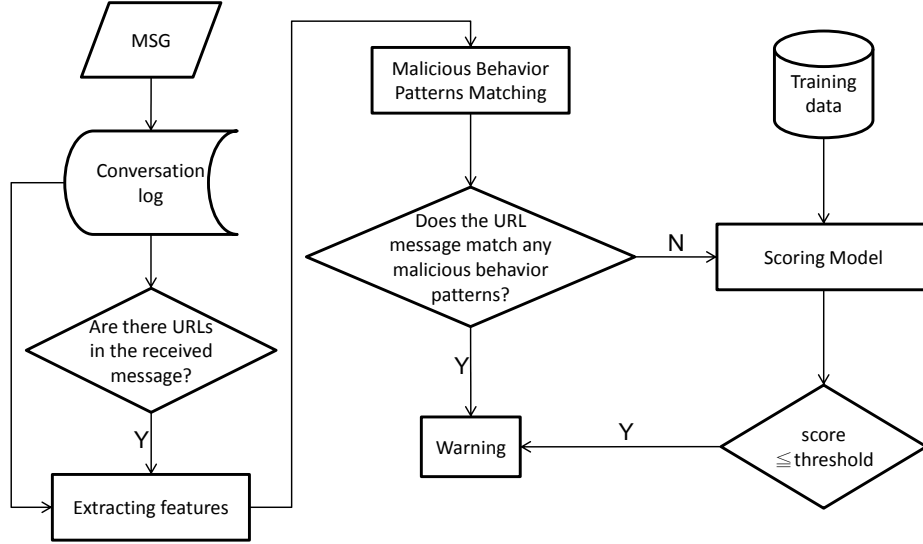
In this section, we propose an approach for detecting IM malicious URL based on anomalies of URL messages and sender’s behavior in IM client side. Fig. 1 shows the architecture of our IM malicious URL detector, which comprises three main modules. As displayed, the feature extraction module is responsible for extracting some anomalous features. These features are used by the other two modules. The malicious behavior matching module matches some known malicious behavior we defined. These malicious behavior patterns are composed of features from the feature extraction module. The anomalous feature scoring module uses a scoring model to determine the score of a URL message. Finally, a warning is generated if the score is below the threshold or a URL message is matched by the malicious behavior matching module.



**Fig. 1.** The architecture of the IM malicious URL detector.

Fig. 2 illustrates the detection flow of our approach. When a IM user receive a message, we first check whether the message contain any URLs. If a message include a URL, the URL may be malicious or not. Then we input the URL message to the malicious URL detector, and extract suspicious features about chat behavior and URL. Next, we use a set of malicious behavior patterns to match the URL message. Once a URL message matches any malicious behavior patterns, we directly consider it as a malicious URL, and a warning is generated.

Otherwise, we subsequently use a scoring model to determine whether the URL is malicious or not. The scoring model is based on a training dataset and a set of features. If the score of a URL message is less than or equal to the threshold, zero, the URL is regarded as a malicious URL and a warning is generated.



**Fig. 2.** The detection flow of our approach.

To achieve faster detection, we only examine whether a URL is abnormal instead of checking the source code of a URL. We use some existing features previous work presented and some new features we observed to distinguish abnormal URLs from legitimate URLs. In addition, we check whether the sender is likely to be a bot by the regularity of transmit delay.

To find features of IM malicious URLs, we observed IM conversation logs that have URL messages. A conversation logs contain all sessions of a day between two IM users. All logs are from two mainly IM services, Microsoft Windows Live Messenger (MSN) and Yahoo! Messenger. We also review historical malicious URLs produced by known IM malware from the internet. By observing these samples, we decide some features for malicious behavior patterns matching and a scoring model which are described as follows respectively.

### 3.1 Malicious Behavior Patterns

We defined 8 known malicious behavior to increase detection speed. As long as a URL message satisfy any malicious behavior patterns described below, the URL is classified as malicious and the system issue a warning. These malicious behavior are based on our observations of normal and abnormal chat logs.

1. **IM Username in Text + First URL Message:** We found that malicious URL messages usually contain an IM username of a sender or a receiver in text parts. (If IM account is an e-mail address, an IM username is a string that appears before the @ sign.) In normal conversation logs, we did not observe existence of this situation, but it still has a little likelihood. To increase the accuracy of malicious behavior pattern, we additionally check another feature at the same time. We think that it is suspicious that the first received message contains a URL in a day. Because hackers usually send only one URL message without other conversations and through a valid IM account that have no contact with a receiver for a long time. By combining these two features, we can identify some malicious behavior. Therefore, when we received the first URL message from a IM user and the text of the URL message contains an IM username, we regarded the URL as malicious.
2. **Regular Delay Time of Sender:** Delay time of a sender means the time difference that a sender transmits messages. We check whether delay time is regular, that is, the difference of any two successive delay time is either zero or one second. [1] mentioned the types of chat bots determined by their triggering mechanisms include periodic, random and responder bot. By periodic bots, they have fixed time intervals, that is, all delay time are equal. Because the bandwidth of the internet may results in deviations, we tolerate one second error. Thus, if delay time of a sender is regular, we regard the URL that the sender transmits as a malicious link because the sender is likely to be a periodic bot.
3. **Regular Response Time of Sender:** Response time of a sender means the time difference that a sender responds to a receiver. According to our observation, a responder bot send message based on the action of a receiver and its response time is regular. In the same way, we check whether the difference of any two successive response time is zero or one second. If response time of a sender is regular, the sender may be a responder bot and then labels the URL he sends as a malicious URL.
4. **Fresh Domain:** A fresh domain means a registered domain is created for a few days. We use a WHOIS query to check the creation date of a domain. Hackers usually register a lot of new domains to set up their malware links or phishing websites. Moreover, the lifetimes of these malicious links are short. According to statistics of [2] from the APWG, in the first half of 2008, the average and median uptimes were 49.5 hours and 19.5 hours, respectively. Thus, we check if the difference between the domain creation date and the date of receiving the URL message is less than or equal to 49.5 hours, about 2 days. We directly regard a URL that has a fresh domain as malicious.
5. **IM Username in URL + Low Reputation of Domain:** To make a URL looks like a legitimate URL, hackers frequently embed an IM username of a sender or a receiver in a URL. That is, a username may be

used as a subdomain of the registered domain or become a path part of a URL. For instance, <http://username.prime-party-pics.com>, <http://host.piclooks.com/?username>. However, some legitimate websites that provide personal services (e.g., blog, album, video) also establish unique URLs for members by using their account name. For instance, <http://username.blogspot.com>, <http://www.youtube.com/user/username>. Thus, we additionally utilize the Google search to query the reputation of a domain. For instance, if <http://www.google.com>, we feed “google.com” to the Google search. If this domain can be found in the domain of search results, it represents that the domain is reputable. Because most malicious URLs have short lifetimes and lack links pointing to them (i.e., result in low PageRank), their domain usually do not appear in the domain of search results. We will simply check domains of the first two results. Therefore, when we receive a URL that contains a IM username, we will feed the domain to the Google search. If the top two domain are not the domain of received URL, we consider the received URL to be a malicious URL.

6. **E-mail Address in URL:** According to our observations, hackers may embed an e-mail address of a sender or a receiver in the URL. For example, <http://username@hotmail.com.fddcol.com>, <http://mainalbum.yoyohost.com/image.php?username@hotmail.com>. This situation is uncommon for a legitimate URL. Because an IM account maybe a e-mail address, hackers attempt to make a receiver to think this URL is legitimate by embedding a e-mail address in the URL. However, above the first example, it actually causes the string to the left of the @ symbol to be disregarded in the browser, while the string on the right is the actual hostname. This behavior belongs malicious, and we have not find yet that legitimate URLs that contain a e-mail address. Thus, we can regard a URL embedded a e-mail address as malicious.
7. **Hostname is Encoded:** Some abnormal URLs have a hostname that be percent-encoded, such as <http://www.%64isc%72%65%74%2done-%6ei%67h%74.%63o%6d>. Obviously the URL have malicious behavior because a legitimate URL does not encode its hostname to make users confused. Thus, when receiving a URL whose hostname is percent-encoded, we can deem it as malicious.
8. **IP is Encoded:** Some hackers replace the hostname with an IP address to make it difficult for users to know where the link is directed to. Even they alter the IP address to a informal type, such as hexadecimal or big integer representations. Although a legitimate URL may be an IP-based URL, it does not change the normal notation (i.e., dot-decimal notation, e.g., 210.218.213.134). Thus, we check whether the IP address is represented in hexadecimal (e.g., <http://0x42.0x1D.0x25.0xC2/>) or big integer (e.g., <http://1037729794/cache>). Then we regard these URLs with a informal notation as a malicious URL.

### 3.2 Features Used to Score

When a received URL message does not match one of malicious behavior patterns, we still can not think that it is legitimate. We will use a scoring mechanism with 11 features to check a received URL message is malicious or not. Next we will describe which features we used and the scoring algorithm is followed.

- **Feature 1 - IM Username in Text ( $F_1$ ):** This feature checks whether a URL message contains an IM username in the text. If any IM username exists in the text,  $F_1 = 1$ . If no username appears in the text, or a URL message has no text part,  $F_1 = 0$ .
- **Feature 2 - First URL Message ( $F_2$ ):** This feature checks if a received URL message is the first message of a day. If a URL message we received is the first message the other party sent,  $F_2 = 1$ ; otherwise,  $F_2 = 0$ .
- **Feature 3 - IM Username in URL ( $F_3$ ):** This feature checks if a URL contains any IM username, and denotes  $F_3 = 1$ ; otherwise,  $F_3 = 0$ .
- **Feature 4 - IP-based URL ( $F_4$ ):** This feature checks whether the hostname of a URL is an IP address. Some PCs used by hackers may not have hostnames, and the direct way is by an IP address. The hostname of a URL is an IP address which makes users unknown where this URL links to, so a legitimate website usually has a hostname. This feature is also used in several papers about phishing detection [3, 9, 10]. Therefore, if a hostname is an IP address despite its encoding,  $F_4 = 1$ ; otherwise,  $F_4 = 0$ .
- **Feature 5 - Confused URL ( $F_5$ ):** Some malicious URLs contain more “http” patterns, these look like URLs including redirection (e.g., `http://3104.mnu4urye.info?http://c43n34.com?35u3b`). But it actually not redirect to other site. In IM environment, a normal user rarely send a URL that contains redirection because IM users usually share a URL link based on they have browsed the URL. This is also a form of phishing attacks, and the path usually contains a URL of a known organization (e.g., `http://202.127.109.148/http://www.paypal.com/us/Confirm.htm`). Another confused situation is that the path contains a valid domain name without the pattern “http” (e.g., `http://sparkleyourcake.com/www.paypal.fr/`). Thus, this feature checks whether the path or the query of a URL contains the pattern of “http(s):” or “www.”, and denotes  $F_5 = 1$ ; otherwise,  $F_5 = 0$ .
- **Feature 6 - Domain in Google Search Result ( $F_6$ ):** This feature checks whether the domain of a received URL matches any domains of the top two search results. If both domains of the first two search results do not appear the domain of a URL we received,  $F_6 = 0$ ; otherwise,  $F_6 = 1$ .

- **Feature 7 - Entropy of Delay Time ( $F_7$ ):** This feature calculates the entropy of delay time of a sender. According to [1], it mentioned that a low entropy indicates the regular behavior of a likely chat bot, while a high entropy indicates the irregular behavior of a possible human. Therefore, we use entropy measures of delay time and response time to estimate the behavior of a sender that transmits a URL message. The Shannon entropy we applied [11],  $H(X)$  for a random variable  $X$  with  $n$  outcomes  $\{x_i : i = 1, \dots, n\}$ , is defined as

$$H(X) = - \sum_{i=1}^n p(x_i) \log_2 p(x_i),$$

where  $p(x_i)$  is the probability mass function of outcome  $x_i$ . In this paper, the random variable  $X$  represents delay time or response time of a sender. A lower entropy of delay time implies more regular delay time of a sender or less messages before a URL message sent. Thus,  $F_7$  is the entropy of delay time of a sender which is a positive value or zero. If a sender has no delay time, that is, he does not send any messages before sending a URL message,  $F_7 = -1$ .

- **Feature 8 - Entropy of Response Time ( $F_8$ ):** Similarly  $F_7$ , this feature calculates the entropy of response time of a sender. A lower entropy of response time implies more regular response time of a sender or less interactions before sending a URL message. If a sender has no response time,  $F_8 = -1$ ; otherwise,  $F_8$  is the entropy of response time of a sender.
- **Feature 9 - Age of Domain ( $F_9$ ):** This feature checks how many days has the domain been registered. We calculate the difference between the date a URL message received and the creation date of the domain, and denote by  $F_9$ . If the WHOIS database can not find the creation date of a domain,  $F_9 = -1$ . The technique of the WHOIS query is also used in some phishing detection research [3, 6, 9, 10].
- **Feature 10 - Dashes in Hostname ( $F_{10}$ ):** This feature checks the number of dashes in the hostname of a URL. We observed some malicious URLs whose hostname is likely a long sentence, which the words concatenated by dashes, such as `http://yj4yb6hmb3.boy-cant-get-you-out-of-my-head.cn/yj4yb6hmb3/Oraliao_show_23Y`. Mostly legitimate sites rarely use dashes in the hostname. So, we count the number of dashes in the hostname of a URL and denote by  $F_{10}$ .
- **Feature 11 - Longest Domain Label ( $F_{11}$ ):** Domain label means that those strings which separated by dots [12]. This feature checks the number of characters of the longest domain label. We found some malicious URLs have relatively long label in the hostname. However, legitimate sites do not use a too long label as its domain so that it is hard to remember the URL for users. For example, a malicious URL, `http://31837`.

hzaseruijintunhfeugandeikisn.com/5/54878/, its longest label has 28 characters (i.e., “hzaseruijintunhfeugandeikisn”). We denote this feature by  $F_{11}$ .

### 3.3 Scoring Algorithm

The scoring algorithm aims to determine a scoring model for predicting a URL message is malicious or not. The scoring model depends on a set of training data, which include  $N_b$  benign data and  $N_m$  malicious data. We set that  $N_b = N_m$ . Using 11 features described in Section 3.2, we calculate the corresponding scores for different feature values. We check these 11 features of a tested URL message, and give the corresponding scores. Finally, total score  $S$  is the sum of the score of each feature value. If  $S \leq 0$ , the URL regard as malicious; otherwise, the URL regard as benign.

In the scoring algorithm, we first construct the frequency distribution of each feature  $F_i, i = 1, \dots, 11$ , in two class of different data.  $FD_{ib}$  and  $FD_{im}$  denote the frequency distribution of feature values that  $F_i$  appears in the benign and malicious training data respectively. Because the number of some feature values that appears in the training data is too large, we will divide those feature values into some groups. Then the calculation of the score is based on grouped feature values. Therefore, we decide a set  $R_i$  which request to score, and it maybe contains one value or multiple values or some range of values. Those feature values that do not exist in the  $R_i$  represent that corresponding scores are zero.

The scoring algorithm  $Score(FD_{ib}, FD_{im}, R_i)$  takes  $FD_{ib}$ ,  $FD_{im}$  and  $R_i$  as inputs, and the output is the scoring tabulation of feature values within the  $R_i$ . For each element in the  $R_i$ , we recursively calculate the corresponding score as follows:

1. Calculating the number of benign and malicious data that have the feature value  $R_{ij}$  respectively,  $n_b$  and  $n_m$ . Where  $R_{ij}$  is a element in  $R_i$  which is a value or a range of values.
2. Calculating the score of  $R_{ij}$ :

$$S_{ij} = \frac{n_b - n_m}{N_b}.$$

Where  $n_b - n_m$  means the significance of this feature value, a absolute value that tends to zero implies this feature value is relatively unable to distinguish two class of data properly. On the other hand, the best effectiveness of a feature value represents one class data having the feature value and the other class data having no the feature value. If  $n_m > n_b$ , implies the significance for malicious data, we set a negative value. Therefore, we use two class data with the same amount for training. In this case, the best effect of a feature value is  $N_b (= N_m)$ . We then calculate  $S_{ij}$ , the proportion of the effect of a feature value relative to a best effect. So  $S_{ij} = +1$  or  $-1$  represents a feature value has a best effect.

After  $j$  rounds, the number of element within  $R_i$ , a scoring model  $(R_{ij}, S_{ij})$  of the feature  $F_i$  is produced. If a URL message has the feature value  $R_{ij}$ , it gets the corresponding score  $S_{ij}$ . Then the total score is

$$S = \sum_{i=1}^{11} S_{ij},$$

and if  $S > 0$ , the URL is labeled as a legitimate URL; if  $S \leq 0$ , the URL is labeled as a malicious URL.

**Table 1.** Example: Frequency distribution of  $F_2$ .

$FD_{2b}$	
$F_2$	Freq.
0	43
1	7

$FD_{2m}$	
$F_2$	Freq.
0	12
1	38

For example, Table. 1 shows the value and the corresponding frequency of feature 2 (First URL Message) appeared in the benign and malicious training data respectively. In this example, we have  $N_b = N_m = 50$  training data. Now, if we only desire to calculate the score of  $F_2 = 1$ , we set  $R_2 = \{1\}$ . Therefore,  $n_b - n_m = 7 - 38 = -31$ , and  $S_{ij} = -31/50 = -0.62$ . When a tested URL message has feature 2, it can get the score of  $-0.62$ . After the score of 11 feature values is decided, we check whether the total score is less than or equal to threshold, zero. If  $S \leq 0$ , we regard this URL as malicious.

### 3.4 Strategies for Score

In this section, we describe the set  $R_i$  we request to score. We decide a proper strategy for scoring and summarize in Table 2.

Note that features  $F_1$  to  $F_5$  are binary, and belong suspicious features. That is, one URL message having these features maybe malicious, but one URL message without these features maybe benign or malicious. It means that when these features is zero, its significance is small. Thus, we set  $R_1$  to  $R_5$  is  $\{1\}$ . We expect that the score is a negative value. As for another binary feature  $F_6$ , if the feature value is 0, it represents a URL is suspicious. So, we only consider  $R_6$  is  $\{0\}$ .

Because continuous features  $F_7$  and  $F_8$  have too much values, we divide all values appeared in training data into several intervals. We take the average value of the feature values of benign and malicious training data to be two partition points. In our data, the average of benign data  $Avg_b$  is greater than the average of malicious data  $Avg_m$ , we conduct three groups  $\{0 \sim Avg_m\}$ ,  $\{Avg_m \sim Avg_b\}$ ,  $\{Avg_b \sim\}$ . About  $F_9$ , this feature grouped by the interval of 30 days, and be one group if  $F_9 > 360$ . The last two features  $F_{10}$  and  $F_{11}$  are also continuous. Because their feature values do not too much, we set  $R_i$  is all appeared values in the training data.

**Table 2.** Feature values used to score.

Features	Selected $R_i$
$F_1$ : IM Username in Text	$R_1 = \{1\}$
$F_2$ : First URL Message	$R_2 = \{1\}$
$F_3$ : IM Username in URL	$R_3 = \{1\}$
$F_4$ : IP-based URL	$R_4 = \{1\}$
$F_5$ : Confused URL	$R_5 = \{1\}$
$F_6$ : Domain in Google Search Result	$R_6 = \{0\}$
$F_7$ : Entropy of Delay Time	$R_7 = \{\{0 \sim Avg_m\}, \{Avg_m \sim Avg_b\}, \{Avg_b \sim\}\}$
$F_8$ : Entropy of Response Time	$R_8 = \{\{0 \sim Avg_m\}, \{Avg_m \sim Avg_b\}, \{Avg_b \sim\}\}$
$F_9$ : Age of Domain	$R_9 = \{\{0 \sim 30\}, \{30 \sim 60\}, \dots, \{330 \sim 360\}, \{360 \sim\}\}$
$F_{10}$ : Dashes in Hostname	$R_{10} = \{\text{All appeared values}\}$
$F_{11}$ : Longest Domain Label	$R_{11} = \{\text{All appeared values}\}$

## 4 Experimental Evaluation

In this section, we evaluate the performance of our proposed approach. We collected IM conversation logs that contain URL messages from the network administrator of an anonymous corporation and our own logs. We check whether each URL message is benign or not by actually clicking URL links and observing conversation contexts. In order to collect more malicious URL messages, we created several IM fake accounts, and executed IM malware files or typed username and password to IM phishing sites. Totally we have 202 chat logs with benign URLs, and 222 chat logs with malicious URLs. Each log is conversation contents of two IM users all day, and may contains many URL messages. We then extract each URL message and its corresponding features to store in the database system. There are 356 benign URL messages and 315 malicious URL messages in our database, and these are our experimental data.

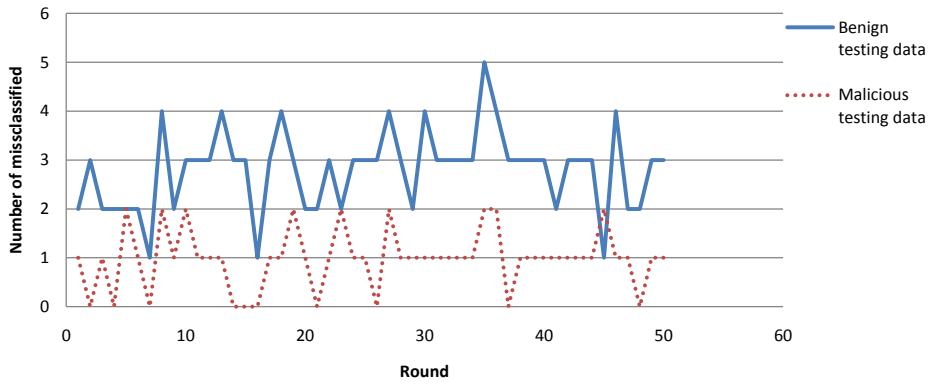
To study the effectiveness of our approach, we conduct the experiment with different training dataset. Each training dataset is randomly drawn from our database, and consists of 50 benign URL messages and 50 malicious URL messages. Remaining data are composed the testing dataset, which contains 306 benign URL messages and 265 malicious URL messages. We run experiment 50 times for each distinct training dataset and have the mean value for these 50 experiments. Then, we use two metrics to measure the performance of our

approach:

$$\text{False Positive Rate (FP)} = \frac{N_{ben \rightarrow mal}}{N_{ben}},$$

$$\text{False Negative Rate (FN)} = \frac{N_{mal \rightarrow ben}}{N_{mal}}.$$

Where  $N_{ben}$  is the number of benign URL messages in the testing dataset;  $N_{ben \rightarrow mal}$  is the number of benign URL messages that are misclassified as malicious URL messages;  $N_{mal}$  is the number of malicious URL messages in the testing dataset;  $N_{mal \rightarrow ben}$  is the number of malicious URL messages that are misclassified as benign URL messages.



**Fig. 3.** Number of false positives and false negatives for each experimental round.

The experimental result shows the false positive rate is 0.93% and the false negative rate is 0.37%. Both values achieve an acceptable result. We also display the number of false positives and false negatives for each experimental round in Fig. 3. The number of false positives  $N_{ben \rightarrow mal}$  is from 1 to 5. The number of false negatives  $N_{mal \rightarrow ben}$  is from 0 to 2. In the false positives, we found that one URL messages just match the malicious behavior pattern “Regular Delay Time of Sender”. Because his two successive delay time are equal. However, its probability should be very small in normal circumstance. And, some benign URL messages are first message in a day, while the URL is IP-based URL or the URL contains IM username. Such situations may get more negative scores to make them false positives. In the false negatives, we found that the creation date of some domain are not found by the whois query. Moreover, these domain could match the domain of google search result. If these URL messages also have no other suspicious features, this will result in our false negatives. For example, “contact-analyser.tk”, “contact-grabber.tk”, “contact-list-analyser.tk”, they are domain name in our false negatives. These web pages claim can tell you who blocked you in their MSN, but actually steal your IM account and password at

the same time. When a malicious website hosted in a domain provided by a free webspace service, we also may missclassify a malicious URL to a benign URL.

## 5 Conclusions

In this paper, we presented an anomaly based approach for detecting instant messaging malicious URLs in real-time. This approach is based on the anomalies of URL messages and sender's behavior in client side. We defined 8 malicious behavior patterns to identify known malicious behavior. We also design a scoring model by 11 features to predict a URL message that does not match any malicious behavior patterns. In our experimental evaluations, the results show that our approach achieves 0.93% false positive rate and 0.37% false negative rate on average.

In the future, we plan on collecting more URL message samples to make our experiments more accurate and convincing. About IM useful bots which provide some useful information by interactions, we will collect more related samples and test the effectiveness of our approach to IM useful bots.

## References

1. Gianvecchio, S., Xie, M., ZhenyuWu, Wang, H.: Measurement and Classification of Humans and Bots in Internet Chat. In: 17th USENIX Security Symposium (2008)
2. Anti-Phishing Working Group: Global Phishing Survey: Domain Name Use and Trends in 1H2008, [http://apwg.org/reports/APWG\\_GlobalPhishingSurvey1H2008.pdf](http://apwg.org/reports/APWG_GlobalPhishingSurvey1H2008.pdf)
3. Basnet, R., Mukkamala, S., Sung, A.H.: Detection of Phishing Attacks: A Machine Learning Approach. *Soft Computing Applications in Industry* (2008)
4. Garera, S., Provos, N., Chew, M., Rubin, A.D.: A Framework for Detection and Measurement of Phishing Attacks. In: Proceedings of the 2007 ACM workshop on Recurring malcode, 1–8 (2007)
5. Mannan, M., van Oorschot, P.C.: On Instant Messaging Worms, Analysis and Countermeasures. In: Proceedings of the 2005 ACM workshop on Rapid malcode, 2–11 (2005)
6. Pan, Y., Ding, X.: Anomaly Based Web Phishing Page Detection. In: Proceedings of the 22nd Annual Computer Security Applications Conference on Annual Computer Security Applications Conference, 381–392 (2006)
7. Williamson, M.M., Parry, A., Hyde, A.: Virus Throttling for Instant Messaging. In: Virus Bulletin Conference, 38–48 (2004)
8. Xie, M., Wu, Z., Wang, H.: HoneyIM: Fast Detection and Suppression of Instant Messaging Malware in Enterprise-like Networks. In: Proceedings of the 2007 Annual Computer Security Applications Conference (ACSAC '07) (2007)
9. Zhang, Y., Hong, J., Cranor, L.: CANTINA: A Content-Based Approach to Detecting Phishing Web Sites. In: Proceedings of the 16th international conference on World Wide Web (WWW '07) (2007)
10. Fette, I., Sadeh, N., Tomasic, A.: Learning to Detect Phishing Emails. In: Proceedings of the 16th international conference on World Wide Web (WWW '07) (2007)

11. Wikipedia: Entropy (information theory), [http://en.wikipedia.org/wiki/Information\\_entropy](http://en.wikipedia.org/wiki/Information_entropy)
12. Wikipedia: Domain Name System, [http://en.wikipedia.org/wiki/Domain\\_Name\\_System](http://en.wikipedia.org/wiki/Domain_Name_System)