# Phish-IDetector: Message-Id Based Automatic Phishing Detection

Rakesh Verma and Nirmala Rai
*Department of Computer Science, University of Houston, 4800 Calhoun Road, Houston, Texas, U.S.A.*

Keywords: Phishing, Message-ID, N-gram, Machine Learning.

Abstract: Phishing attacks are a well known problem in our age of electronic communication. Sensitive information like credit card details, login credentials for account, etc. are targeted by phishers. Emails are the most common channel for launching phishing attacks. They are made to resemble genuine ones as much as possible to fool recipients into divulging private and sensitive data, causing huge monetary losses every year. This paper presents a novel approach to detect phishing emails, which is simple and effective. It leverages the unique characteristics of the Message-ID field of an email header for successful detection and differentiation of phishing emails from legitimate ones. Using machine learning classifiers on n-gram features extracted from Message-IDs, we obtain over 99% detection rate with low false positives.



## 1 INTRODUCTION

With an overwhelming increase in the number of Internet users, the incidents of cyber-crimes are also increasing exponentially. Every year money, time and productivity is lost, and valuable information and private details are compromised through various cyber-attacks. One of the most popular techniques used to steal sensitive user information and credentials is phishing. This form of attack targets individual Internet users as well as small or large businesses. Typically, carefully designed electronic mails are employed to lure victims into revealing sensitive data. Since emails are the most common means of launching a phishing attack, our work concentrates on detecting these phishing emails and distinguishing them from legitimate ones.

Every email consists of mainly two parts: the header and the body. The header consists of several pre-formatted fields such as From, Delivered-To, Subject, Message-ID, etc. The body consists of the main content of the email, usually in text/HTML format. The phishers make it very difficult to detect the phishing emails by meticulously constructing them to closely resemble legitimate ones. This makes the process of distinction non-trivial, which has been observed by other researchers (Irani et al., 2008) also. The email body is completely under the control of the sender while the header follows a relatively stricter format and is not entirely controlled by the sender. So we focus in this paper on detection based on email headers. In particular, based on looking at a few (less than 10) legitimate emails and the same number of phishing emails, our attention was drawn to the Message-ID field. This field is a string following a certain basic format described below. It also contains information designed to make the email globally unique. It cannot be altered easily and it provides important information about the email which includes it.

Our work centers on these useful properties of Message-IDs and exploits it further by applying n-gram analysis to the Message-IDs. Various machine learning algorithms including an on-line confidence weighted learning algorithm were employed using 10-fold cross validation on different data sets and they produced detection rates of above 99%. To our knowledge, this is the first time Message-IDs have been used with n-gram analysis to detect phishing emails.

### 1.1 Background

Electronic mail or email proliferated during the 1990s and has evolved to become an indispensable part of our current social fabric. Essentially, an email has two parts: the header and the body. The email body contains the actual message being sent from the sender to the receiver and is completely under the control of the sender. Whereas, email header consists of several fields, some mandatory and some optional, which carry information regarding the source, destination, routing details, timestamps etc. (Resnick, 2001). Thus, the header cannot be completely ma-

M. Verma R. and Rai N..
Phish-IDetector: Message-Id Based Automatic Phishing Detection.
DOI: 10.5220/0005574304270434
In *Proceedings of the 12th International Conference on Security and Cryptography* (SECRYPT-2015), pages 427-434
ISBN: 978-989-758-117-5

nipulated by the sender. One such email header field is Message-ID. It is used by all mail transfer agents (MTAs) to uniquely identify an email. RF2822 recommends using it even though it is an optional field.

In our experimental dataset 100% of the legitimate emails had Message-IDs. We also conducted an independent experiment with 10 volunteers to determine how often this field is present in legitimate emails. We found that almost 99% of legitimate emails have the Message-ID field. This gives us statistical confidence to state that an absence of the Message-ID field could be considered a red flag and a phisher would have to include it to avoid attracting suspicion.

Sendmail, one of the MTAs uses Message-ID for tracing emails and for logging process ids (Costales et al., 2007). Sendmail specification recommends including Message-ID in emails and also the setting of relevant macros in its configuration file in order to implement compulsory checking of Message-IDs (Costales et al., 2007). "Unlike spoofing other fields in the header, spoofing Message-ID needs special knowledge. Only technical savvy spammers can spoof the Message-ID cleverly" (Pasupatheeswaran, 2008). So, deep analysis on Message-IDs may reveal some sort of information that could open a window to trace the source of an email.

Based on this hypothesis, we delved deeper into Message-ID using n-gram analysis up to 10-grams and found the optimum detection rates at around 5- or 6-grams. For both higher and lower order n-grams the rates usually deteriorate. We applied several machine learning classifiers using stable version 3.6 of Weka (Hall et al., 2009) and an on-line confidence weighted learning algorithm of (Mejer and Crammer, 2010).

We provide an explanation of some terms we will use frequently throughout the paper.

**Message-ID.** RFC 2822 states that each email must have a globally unique identifier called Message-ID. If this is included it must be in the email header. RFC 2822 also defines the syntax of Message-ID. It should be like a legitimate email address and it must be included within a pair of angle brackets. A typical Message-ID looks like the following:<20020923025816.8E7A34A8@mercea.net>. According to RFC 2822, Message-ID can appear in three header fields. They are Message-ID header, in-reply-to header and references header. But Message-ID of the present email must be included against the Message-ID header (Pasupatheeswaran, 2008). It has a fixed format of the form <LHS@RHS>where the left hand side (LHS) is a representation of information including current time stamp, queue id, etc. coded in different formats according to the Sendmail version. The right hand side (RHS) represents the *fully qualified domain name* (FQDN). This part starts with local host name followed by a dot and other parts of domain information (Costales et al., 2007).

**N-gram.** The concept of N-gram is related to natural language processing. It is a sequence of n characters in a string or text. For e.g. if the text is abc123 the 1-grams would consist of one character sequence e.g. a, c, 2, etc. Similarly, 2-grams would be overlapping sequence of 2 characters like ab, bc, c1, 12, 23. This idea can be further extended to higher order n-grams in a similar fashion.

# 2 THE OVERALL APPROACH

The complete process can be summarized in the following sequence of steps.

## 2.1 Message-ID Extraction

For our study we decided to choose Message-ID as the distinguishing property because of its content, uniqueness and fixed format. All the Message-IDs from the emails of different datasets are extracted using grep command and stored in a file. Since each Message-ID is of the format <LHS@RHS>, we get rid of the <, @ and >symbols common to all Message-IDs as a pre-processing step. After this step we get two attributes for each Message-ID. We have named them LHS and RHS to denote the left hand side and the right hand side of the Message-ID.

## 2.2 Input File Creation

Depending on which dataset the email belonged to, we labeled each instance as belonging to either "phishing" or "legit" (legitimate) class. We created a csv file with three columns: class label, LHS and RHS. The RHS part being of a more consistent format rather than LHS, we tried the classification based on only RHS as well. In that case, there are only two columns: class label and RHS.

## 2.3 N-gram Analysis

Further, we decided to do n-gram analysis of the collected Message-IDs so that we could represent the data in numeric format acceptable to most classifiers in Weka. This analysis generates unique n-gram features represented as Unicode code point of the characters. For e.g. the Unicode code point for "a" is 97

so the 1-gram "a" will be represented as 97. The feature extracted is the frequency of the n-gram in the attribute LHS or RHS.

### 2.4 Classification

Once we obtained and represented the data in arff format, we ran the following six classifiers on the arff file using Weka 3.6: RandomForest, J48, Bagging, AdaboostM1, SMO and NaiveBayes. We also ran four other classifiers but their performance was not at par with the 6 classifiers mentioned, so we omit their results. These are: ClassificationViaClustering, ComplementNaiveBayes, ZeroR and BayesNet. The arff files were also converted to .svm format, the input format for the confidence weighted algorithm (Mejer and Crammer, 2010), using a python script.

## 3 DATA SETS AND CLASSIFIERS

We have used two publicly available datasets. Email Message-IDs were collected from 4,550 public phishing emails from (Nazario, 2004) and from 9,706 legitimate emails from SpamAssassin public corpus datasets at (SpamAssassin, 2006). The phishing corpus and even the SpamAssassin ham corpus we used has been used previously by (Fette et al., 2007), (Toolan and Carthy, 2010), (Hamid and Abawajy, 2011).

SpamAssassin corpus segregates the emails into different subsets which we named as follows:

1. easy_ham consisting of 5051 emails
2. easy_ham_1 consisting of 2500 emails
3. easy_ham_2 consisting of 1400 emails
4. hard_ham consisting of 500 emails
5. hard_ham_1 consisting of 250 emails

All these emails had Message-IDs. We ran experiments on the phishing emails combined with each of the above mentioned subsets of legitimate emails. All the Message-IDs obtained from the phishing emails were added to the set of Message-IDs extracted from each of the above mentioned ham sets. These datasets are hence named according to the ham set involved in creating the dataset since the phishing set of Message-IDs is common to all of them. Additionally, all the experiments were performed once taking only RHS into account and once taking both the RHS and LHS into account, and the dataset names have been prefixed with RHS and SplitMsgId respectively. The names of the datasets are as follows:

1. RHSEasyHam and SplitMsgIdEasyHam
2. RHSEasyHam1 and SplitMsgIdEasyHam1
3. RHSEasyHam2 and SplitMsgIdEasyHam2
4. RHSHardHam and SplitMsgIdHardHam
5. RHSHardHam1 and SplitMsgIdHardHam1

We used Weka version 3.6 which is basically a collection of machine learning algorithms for data mining tasks. It was chosen because of its wide acceptability, popularity and its ease of use. It has previously been used for phishing detection by (Hamid and Abawajy, 2011) and (Chen et al., 2014). Weka provided us an easy method of comparing the performance of several classifiers on our datasets and choosing the best among them. We ran the experiments with around 10 classifiers and chose the best 6 among them. Each of them is explained here in brief.

**Random Forest classifier** (Breiman, 2001) consists of several decision tree classifiers. Each tree has a random set of features out of the total feature collection and this algorithm returns the maximum frequency class among all of the individual decision trees. It performed the best quite consistently in our experiments. For our experiments we used the default implementation of Weka 3.6 for the Random Forest classifier.

**J48** is a Java implementation of the decision tree formed by classifier C4.5 (Quinlan, 2014).

**SMO** is an implementation of sequential minimal optimization algorithm devised by John Platt for training a support vector classifier. All attributes are normalized by default in this algorithm. A more detailed explanation can be found at (Platt et al., 1999)

Bootstrap Aggregating or **Bagging** is a method for generating multiple versions of a predictor and using these to get an aggregated predictor. The aggregation averages over the versions when predicting a numerical outcome and does a plurality vote when predicting a class. It is explained in (Breiman, 1996).

**AdaBoostM1** is an implementation of the boosting algorithm by (Freund and Schapire, 1996). It is known to improve performance of a weak learner using a boosting algorithm. We have used the default base classifier for Weka 3.6 in this case.

**NaiveBayes** is the Weka implementation of the Naive Bayes classifier, which is a simple classifier that applies Bayes' theorem. It strictly assumes conditional independence and hence called 'naive.' More information can be found at (John and Langley, 1995).

## 4 INDEPENDENT EXPERIMENT ON MESSAGE-IDS

Due to privacy issues legitimate emails used in the field of phishing emails detection are usually not recent ones. To prove the viability of our method with current data without compromising the privacy aspect, we performed an independent experiment involving 10 anonymous volunteers. Each of them was given instructions along with a script that would collect some statistics from each mail box. We collected only two numbers from each of them, no. of emails(Email Count) and number of Message-IDs(Message-ID Count) not revealing any private data in their emails. The process involved configuring each volunteer's gmail account in their local UNIX machines using postfix and fetchmail. The script then separated the mailbox created for each volunteer into individual messages using procmail. And finally grep command was used to get the email count and the Message-ID count. We had to be careful not to over count the Message-IDs as sometimes a mail can have more than one Message-ID. To avoid such a mistake we used grep with the option of counting only the first occurrence of Message-ID in each email.

The data collected from the volunteers is shown in Table 1. It reveals that nearly 99% of the emails have Message-ID field and proves our hypothesis that in spite of being an optional field it would have to be included in the emails by a phisher to avoid raising any red flags.

Table 1: Email and Message-ID count from the independent experiment. Nearly 99% emails have Message-Ids.

| Message-ID Experiment | | |
|---|---|---|
| Volunteers | EmailCount | Message-ID Count |
| Volunteer 1 | 1959 | 1928 |
| Volunteer 2 | 1613 | 1594 |
| Volunteer 3 | 798 | 787 |
| Volunteer 4 | 719 | 712 |
| Volunteer 5 | 364 | 361 |
| Volunteer 6 | 352 | 352 |
| Volunteer 7 | 325 | 325 |
| Volunteer 8 | 277 | 263 |
| Volunteer 9 | 252 | 252 |
| Volunteer 10 | 118 | 118 |
| Total | 6777 | 6692 |
| Percentage Emails With Message-IDs | | |
| 98.75 | | |

## 5 CONTRIBUTIONS/RESULTS

Major contributions of this paper are: (i) the demonstration that Message-ID field is effective in phishing email detection for the first time and (ii) our approach of applying n-gram analysis technique with a rich variety of classifiers to the Message-ID field of the emails. To the best of our knowledge, n-gram analysis has not been used for phishing detection in a manner similar to ours. Moreover, this is the first paper where Message-IDs have played a pivotal role in phishing detection. Although the confidence weighted learning method has been used for classification of phishing emails, for instance by (Basnet and Sung, 2010), it has been applied to the email text unlike our method where it has been applied to the header information, specifically the Message-ID.

We have also checked for robustness of our method by using different data sets and generating fairly consistent detection rates. The detection and classification of phishing emails based on n-gram analysis of Message-IDs proves to be an excellent technique. We now explain in details about the n-gram analysis and the various classifiers used and the difficulties we faced during our experimental phase.

We wrote a Python script to extract all character n-grams from the Message-IDs and represent them as unique features in the form of Unicode point of the characters. For e.g. the Unicode code point for "a" is 97 so the 1-gram "a" will be represented as 97. The feature extracted is the frequency of the n-gram in the attribute LHS or RHS.

This transforms the string or text in the Message-ID field into set of numeric features and represents them in the form of a sparse arff file which can be processed in Weka. Once the features were in a readable form as arff files for Weka, we ran several classifiers with 10-fold cross-validation on each of the files for n-grams up to 3-grams. Since the file-size increased exponentially for each subsequent n-gram, Weka would crash for any n-gram higher than 3. Also, we could run only two classifiers for the 3-grams files due to the issue of large-sized files. For both 1- and 2-grams files we ran as many as 10 classifiers and found Random Forest to be the most effective of them all, obtaining highest True Positive rates and the lowest False Positive Rates.

For classification based on higher order n-gram analysis we used the faster on-line confidence weighted learning algorithm of (Mejer and Crammer, 2010). We obtained a collection of most confidence weighted learning algorithm into a library written in Java from (Crammer, 2009). Again, we selected the 10-fold cross-validation test option for maintaining

uniformity. With this algorithm we were able to perform the classification for all the files up to 10-grams. Looking at both the TPR and FPR values of these experiments, it was revealed that with an increase in order of n-gram, the classification improves but it starts deteriorating after a certain n-gram value. For most of the experiments this optimum value was obtained at the threshold of around 5- or 6-grams.

We present our results of all classifiers for the best among all the datasets, i.e. Hard Ham. Also, to give an idea of the performance across all datasets we include the results of our best classifiers i.e. Random Forest and J48 for all the datasets.

Table 2: True-Positive and False-Positive Rates for Weka Classifiers on SplitMsgIdHardHam Dataset.

| 1-gram for SplitMsgIdHardHam | | |
|---|---|---|
| Classifiers | TPR | FPR |
| RandomForest | 99.5 | 4.9 |
| J48 | 96.6 | 18 |
| Bagging | 96.7 | 27.5 |
| SMO | 94.3 | 46.8 |
| AdaboostM1 | 94.9 | 37.3 |
| NaiveBayes | 87.2 | 29.7 |

Table 3: True-Positive and False-Positive Rates for Weka Classifiers on SplitMsgIdHardHam Dataset.

| 2-gram for SplitMsgIdHardHam | | |
|---|---|---|
| Classifiers | TPR | FPR |
| RandomForest | 99.4 | 4.9 |
| J48 | 97 | 18.4 |
| Bagging | 97.2 | 23.2 |
| SMO | 97.6 | 8.8 |
| AdaboostM1 | 95 | 37 |
| NaiveBayes | 92 | 29.1 |

Table 4: True-Positive and False-Positive Rates for Weka Classifiers on SplitMsgIdHardHam Dataset.

| 3-gram for SplitMsgIdHardHam | | |
|---|---|---|
| Classifiers | TPR | FPR |
| RandomForest | 99.3 | 5.2 |
| J48 | 98.7 | 8 |

Tables 2–4 summarize the TPR and FPR values of the experiments on dataset SplitMsgIdHardHam. Results show a constant increase in TPR and decrease in FPR for higher order n-grams. So, the 3-grams results are the best in terms of both TPR and FPR.
Random Forest classifier even succeeds in getting 99.5% of the phishing emails detected with a small number of false positives, i.e. legitimate emails classified as phishing.Also, we find that the classifiers that perform the best classification are tree classifiers Random Forest and J48.

Table 5: True-Positive and False-Positive Rates for Weka Classifiers on RHSHardHam Dataset.

| 1-gram for RHSHardHam | | |
|---|---|---|
| Classifiers | TPR | FPR |
| RandomForest | 99.4 | 5 |
| J48 | 96.5 | 17.6 |
| Bagging | 96.7 | 27.4 |
| SMO | 94.3 | 46.8 |
| AdaboostM1 | 93 | 59.4 |
| NaiveBayes | 88.1 | 45.4 |

Table 6: True-Positive and False-Positive Rates for Weka Classifiers on RHSHardHam Dataset.

| 2-gram for RHSHardHam | | |
|---|---|---|
| Classifiers | TPR | FPR |
| RandomForest | 99.3 | 5.2 |
| J48 | 98 | 10.5 |
| Bagging | 97.7 | 18.6 |
| SMO | 98.8 | 5.5 |
| AdaboostM1 | 93.9 | 54.1 |
| NaiveBayes | 92.4 | 35.8 |

Table 7: True-Positive Rate and False-Positive Rate for Weka Classifiers on RHSHardHam Dataset.

| 3-gram for RHSHardHam | | |
|---|---|---|
| Classifiers | TPR | FPR |
| RandomForest | 99.4 | 5 |
| J48 | 97.4 | 16.8 |

Tables 5–7 summarize the TPR and FPR values of the experiments on dataset RHSHardHam. Similar to the SplitMsgIdHardHam dataset results there is a constant increase in TPR and decrease in FPR for higher order n-grams. So, the 3-grams results are the best in terms of both TPR and FPR.

Note that the SplitMsgIdHardHam dataset gives better results as compared to the RHSHardHam dataset. We hypothesize that many phishers lack adequate knowledge of LHS structure or do not spend time on it. Both RandomForest and J48 perform almost consistently well for both the datasets at almost any n-gram value.

Tables 8–11 summarize the TPR and FPR values of the RandomForest and J48 classifiers for the experiments across all datasets. These two classifiers performed the best and we present these tables to compare their results for each of the datasets we used. We find that the results are fairly consistent across

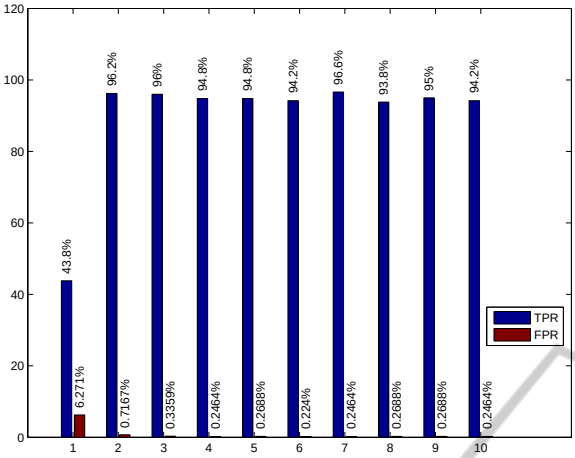


Figure 1: True-Positive and False-Positive Rates for Confidence Weighted Classifier on SplitMsgIdHardHam Dataset.

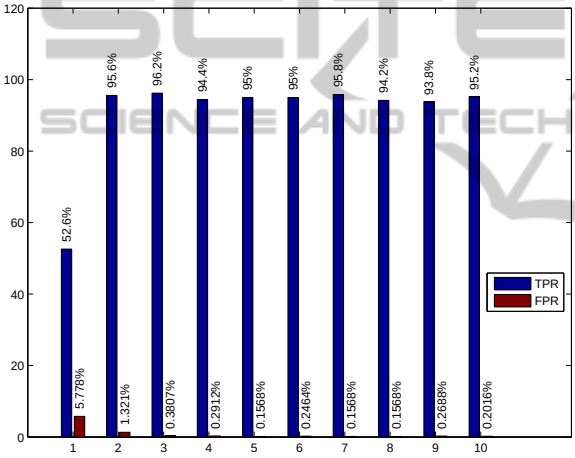


Figure 2: True-Positive Rate and False-Positive Rate for Confidence Weighted Classifier on RHSHardHam Dataset.

Table 8: True-Positive and False-Positive Rates for RandomForest and J48 across all SplitMsgId Datasets.

| 1-gram | | | | |
|---|---|---|---|---|
| DataSet | RandomForests | | J48 | |
| (Split) | TPR | FPR | TPR | FPR |
| EasyHam | 93.7 | 10.1 | 90.2 | 12.7 |
| EasyHam1 | 95.7 | 4.6 | 91.3 | 9.2 |
| EasyHam2 | 95.4 | 13.2 | 91 | 16.4 |
| HardHam | 99.5 | 4.9 | 96.6 | 18 |
| HardHam1 | 98.5 | 26.5 | 97.3 | 36.4 |

datasets and there is a gradual improvement of results with the increase in the order of n-grams.

We present the results of Confidence Weighted Classifier for all 10-gram datasets in figures 1 and 2. The advantage of Confidence Weighted algorithm was that it could easily run on all the 10-gram files and that it had quite low false positive rate consistently as

Table 9: True-Positive and False-Positive Rates for RandomForest and J48 across all SplitMsgId Datasets.

| 2-gram | | | | |
|---|---|---|---|---|
| DataSet | RandomForests | | J48 | |
| (Split) | TPR | FPR | TPR | FPR |
| EasyHam | 93.9 | 9.9 | 91 | 12.1 |
| EasyHam1 | 96.1 | 4.2 | 92.3 | 8.2 |
| EasyHam2 | 95.9 | 12.7 | 92.9 | 15 |
| HardHam | 99.4 | 4.9 | 97 | 18.4 |
| HardHam1 | 98.5 | 26.1 | 97.8 | 33 |

Table 10: True-Positive and False-Positive Rates for RandomForest and J48 across all RHS Data Sets.

| 1-gram | | | | |
|---|---|---|---|---|
| DataSet | RandomForests | | J48 | |
| (RHS) | TPR | FPR | TPR | FPR |
| EasyHam | 95.6 | 4.7 | 91.4 | 9.1 |
| EasyHam1 | 93.7 | 10.1 | 90.2 | 12.7 |
| EasyHam2 | 95.3 | 13.3 | 91 | 16.4 |
| HardHam | 99.4 | 5 | 96.5 | 17.6 |
| HardHam1 | 98.5 | 26.1 | 97.3 | 36.8 |

Table 11: True-Positive Rate and False-Positive Rate for RandomForest and J48 across all RHS Data Sets.

| 2-gram | | | | |
|---|---|---|---|---|
| DataSet | RandomForests | | J48 | |
| (RHS) | TPR | FPR | TPR | FPR |
| EasyHam | 94.8 | 8.8 | 95.9 | 5.3 |
| EasyHam1 | 98.5 | 1.5 | 96.6 | 3.4 |
| EasyHam2 | 95 | 15.1 | 96.1 | 7.5 |
| HardHam | 99.3 | 5.2 | 98 | 10.5 |
| HardHam1 | 97.9 | 36.4 | 98.4 | 22 |

compared to the Weka machine learning classifiers. Though the detection rates are not as high as Random Forest and J48 classifiers, the false positive rates are much lower.

## 6 INFORMATION GAIN

After the first set of experiments, we were curious to know which features were performing the best among all of the 1-gram, 2-gram and 3-gram attributes. A widely accepted method to find out the most effective features in a multi-feature classifier is calculating the information gain for the attributes. It is a measure of the difference in entropy values and is generally defined as follows for a decision tree:

```
IG(T,a) = H(T) - H(T|a)
```

where IG stands for Information gain, the function H represents entropy, T stands for the class and a is

the feature. We present the top 10 features along with their information gain values for each of the 1-gram, 2-gram and 3-gram features. From these IG values we find that for the RHSHardHam dataset, the hyphen '-' symbol is quite dominant as an attribute.

Table 12: Information gain values of Top 10 attributes represented as 'Att' in the table for RHSHardHam Data Set.

| RHSHardHam | | | | | |
|---|---|---|---|---|---|
| 1-gram | | 2-gram | | 3-gram | |
| Att | IG | Att | IG | Att | IG |
| - | 0.110599 | -a | 0.125108 | -sf | 0.125992 |
| a | 0.103357 | bv | 0.118968 | -a | 0.125108 |
| e | 0.073969 | sf | 0.118461 | -ac | 0.122093 |
| . | 0.063256 | v- | 0.117757 | fo1 | 0.122093 |
| s | 0.060871 | 1- | 0.116663 | -ag | 0.122093 |
| f | 0.059647 | o1 | 0.1156 | abv | 0.122093 |
| t | 0.058247 | c- | 0.11354 | bv- | 0.122093 |
| g | 0.055149 | - | 0.110599 | o1- | 0.122093 |
| n | 0.049335 | -s | 0.108526 | 1-a | 0.122093 |
| b | 0.045831 | a | 0.103357 | sfo | 0.122093 |

Table 13: Information gain values of Top 10 attributes represented as 'Att' for SplitMsgIdHardHam Dataset

| SplitMsgIdHardHam | | | | | |
|---|---|---|---|---|---|
| 1-gram | | 2-gram | | 3-gram | |
| Att | IG | Att | IG | Att | IG |
| . | 0.2007 | . | 0.200703 | . | 0.200703 |
| a | 0.16614 | a | 0.16614 | a | 0.16614 |
| o | 0.15855 | o | 0.158549 | o | 0.158549 |
| t | 0.10909 | l. | 0.146136 | il. | 0.146136 |
| r | 0.10193 | .J | 0.14211 | l. | 0.146136 |
| l | 0.09794 | Ma | 0.140517 | .10 | 0.144005 |
| i | 0.09426 | aM | 0.139745 | .J | 0.14211 |
| v | 0.0927 | av | 0.139732 | Ma | 0.140517 |
| M | 0.06281 | ot | 0.139432 | Mai | 0.139745 |
| - | 0.05612 | va | 0.138928 | vaM | 0.139745 |

## 7 SECURITY ANALYSIS

Our method relies on the Message-ID field which, though important and recommended, is optional. Without it, our method would not work. However, note that almost all legitimate emails include this field, and since a phisher tries to fool the user into believing that a phishing email is legitimate, omitting this field could serve as a red flag. In our experimental data set, 100% of the legitimate emails had Message-IDs. Our recent experiment with 10 volunteers reveals that the Message-ID field is present in nearly 99% of the legitimate emails. As mentioned, the exponentially increasing file size for higher order n-grams makes it difficult to run different classifiers on them without using specialized big data approaches. We currently ran only the confidence weighted algorithm on higher order n-gram files, which has proven itself to be competitive in other scenarios, but not guaranteed to be the ideal choice for best results.

Spoofing Message-ID field requires a technically savvy phisher, who is willing to go the extra mile to avoid detection. For example, either this field would have to be deleted, which would raise a red flag in light of our experiment with 10 volunteers, or the phisher would: (i) either fake the FQDN or (ii) copy the entire Message-ID field from a legal message sent earlier, and the phisher would have to turn off any checking in the mail program. For such sophisticated phishers, we recommend combining our classifiers with other classifiers or features from the header, the links and the body text in the email, as, for example, in (Verma et al., 2012).

## 8 RELATED WORK

Since it is a much employed security threat, automatic detection of phishing emails has attracted significant attention of researchers. For lack of space, we focus on prior work involving header analysis and refer to (Verma et al., 2012; Verma and Hossain, 2014) for more comprehensive descriptions.

A hybrid feature selection approach based on combination of content-based and behavior-based features was proposed in (Hamid and Abawajy, 2011). It utilized Message-ID tags of emails to capture attacker behavior. The authors use a binary feature called Domain_sender, which represents the similarity of domain name extracted from email sender with domain in Message-ID. If it is similar, the email is considered legitimate and the value is set to 0, otherwise 1. The hybrid method was able to achieve 96% accuracy with 4% false positives. For the phishing emails, they used the same dataset as ours, but for the legitimate ones, they only used the 2364 easy ham emails of the SpamAssassin corpus.

Another paper that investigates email header is (Pasupatheeswaran, 2008). In this, the Message-ID field and Message-ID generation are discussed in detail and the uniqueness of this field is established. The author shows spoofing of this field is not possible without sound technical knowledge suggesting that Message-ID could be used to find the source of the email, which could be useful in forensic analysis.

A comprehensive approach based on email header, links and body was proposed in (Verma et al., 2012). For header analysis they look at the From, Delivered-to and Received-From fields. For the links and body analysis, we refer to (Verma et al., 2012). Their

method was able to correctly classify 98% of phishing emails and 99.3% of 1000 legitimate emails from authors' inboxes.

## 9 CONCLUSION

We have presented a novel approach that is simple yet effective in detection and classification of phishing emails. We have shown how the unique characteristics of Message-IDs can be exploited with n-gram analysis to produce features that can distinguish between phishing and legitimate emails. Our approach studies the performance of different classifiers on different order of n-gram features and several datasets. It is also the first method that has applied confidence weighted learning algorithm on an email header field instead of the body. The results we obtained are very promising considering the minimal information required by our technique. If combined with existing methods of phishing detection based on header or body analysis, it might even reach higher detection rates with low false positives, and such combinations would be even more robust and harder to attack than individual methods.

## ACKNOWLEDGEMENTS

This research is supported in part by NSF grants CNS 1319212 and DUE 1241772.

## REFERENCES

Basnet, R. B. and Sung, A. H. (2010). Classifying phishing emails using confidence-weighted linear classifiers. In *International Conference on Information Security and Artificial Intelligence (ISAI)*, pages 108–112.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Chen, T.-C., Stepan, T., Dick, S., and Miller, J. (2014). An anti-phishing system employing diffused information. *ACM Transactions on Information and System Security (TISSEC)*, 16(4):16.

Costales, B., Janse, G., Abmann, C., and Shapiro, G. N. (2007). Sendmail (4th ed.). In *Sendmail (4th ed.)*. O'Reilly.

Crammer, K. (2009). Confidence weighted learning library. http://webee.technion.ac.il/people/koby/code-index.html.

Fette, I., Sadeh, N., and Tomasic, A. (2007). Learning to detect phishing emails. In *Proceedings of the 16th international conference on World Wide Web*, pages 649–656. ACM.

Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Thirteenth International Conference on Machine Learning*, pages 148–156, San Francisco. Morgan Kaufmann.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18.

Hamid, I. R. A. and Abawajy, J. (2011). Hybrid feature selection for phishing email detection. In *Algorithms and Architectures for Parallel Processing*, pages 266–275. Springer.

Irani, D., Webb, S., Giffin, J., and Pu, C. (2008). Evolutionary study of phishing. In *eCrime Researchers Summit, 2008*, pages 1–10. IEEE.

John, G. H. and Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In *Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 338–345, San Mateo. Morgan Kaufmann.

Mejer, A. and Crammer, K. (2010). Confidence in structured-prediction using confidence-weighted models. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 971–981. Association for Computational Linguistics.

Nazario, J. (2004). The online phishing corpus. http://monkey.org/ jose/wiki/doku.php.

Pasupatheeswaran, S. (2008). Email 'message-ids' helpful for forensic analysis?

Platt, J. et al. (1999). Fast training of support vector machines using sequential minimal optimization. *Advances in kernel methods-support vector learning*, 3.

Quinlan, J. R. (2014). *C4. 5: programs for machine learning*. Elsevier.

Resnick, P. (2001). Internet message format. http://www.ietf.org/rfc/rfc2822.txt.

SpamAssassin, A. (2006). Spamassassin public mail corpus. https://spamassassin.apache.org/publiccorpus/.

Toolan, F. and Carthy, J. (2010). Feature selection for spam and phishing detection. In *eCrime Researchers Summit (eCrime), 2010*, pages 1–12. IEEE.

Verma, R. and Hossain, N. (2014). Semantic feature selection for text with application to phishing email detection. In *Information Security and Cryptology–ICISC 2013*, pages 455–468. Springer.

Verma, R., Shashidhar, N., and Hossain, N. (2012). Detecting phishing emails the natural language way. In *ESORICS*, pages 824–841.