

A RULE BASED APPROACH FOR DETECTING PHISHING ATTACKS

A DISSERTATION

*Submitted in partial fulfillment of the
requirements for the award of the degree*

of

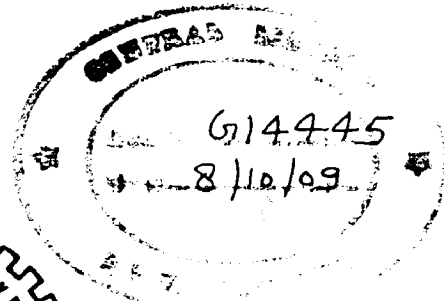
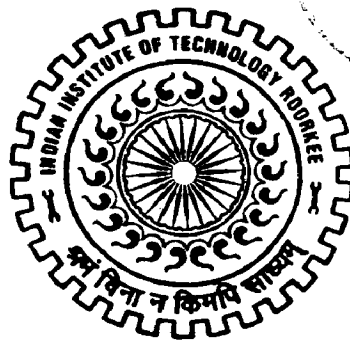
MASTER OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

RAVINDRA KUMAR SINGH



DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE-247 667 (INDIA)

JUNE, 2009

Candidate's Declaration

I hereby declare that the work being presented in the dissertation report titled "A Rule Based Approach for Detecting Phishing Attacks" in partial fulfillment of the requirement for the award of the degree of Master of Technology in Computer Science and Engineering, submitted in the Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, is an authentic record of my own work carried out under the guidance of Dr. Kuldip Singh, Professor, in Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee. I have not submitted the matter embodied in this dissertation report for the award of any other degree.

Dated: 30/06/2009

Place: IIT Roorkee



(Ravindra Kumar Singh)

Certificate

This is to certify that above statements made by the candidate are correct to the best of my knowledge and belief.

Dated: 30/06/2009

Place: IIT Roorkee.


Dr. Kuldip Singh,
Professor,
Department of Electronics
and Computer Engineering,
IIT Roorkee, Roorkee.
247667 (India).

ACKNOWLEDGEMENTS

I am thankful to Indian Institute of Technology Roorkee for giving me this opportunity. It is my privilege to express thanks and my profound gratitude to my supervisors Dr. Kuldeep Singh, Associate Professor for their invaluable guidance and constant encouragement throughout the dissertation. I was able to complete this dissertation in time due to the constant motivation and support received from them.

I am also grateful to Mr. Sandeep Sood, for helping me to understand some basic and important concepts explored in the dissertation work. I am grateful to Mr. Avtar Singh, Mr. Ingawale Abhijeet Digambar, and my colleagues, for being excellent peers and creating a congenial environment for work. I am also thankful to all my friends who helped me directly and indirectly in completing this dissertation. Most importantly, I would like to extend my deepest appreciation to my family for their love, encouragement and moral support.


(Ravindra Singh)

Abstract

Phishing attacks are one of the emerging serious threats against personal data security. These attacks are often performed by sending out emails that seem to originate from a trusted party. The objective is to deceive the recipient to release sensitive information such as usernames, passwords, banking details, or credentials. The aim of phishing is to steal a user's identity in order to make fraudulent transactions as if the Phisher were the user.

Though a large number of methods have been proposed and implemented for detecting Phishing attacks, a complete solution is missing. A great amount of research is being carried out to solve this problem using rule based method, browser based method and machine learning approaches but still there are insufficient methods that can be used against Phisher's novel attacks which they are able change time to time.

In this Dissertation entitled "A RULE BASED APPROACH FOR DETECTING PHISHING ATTACKS", a solution is proposed for organization wide solution; rule set has been proposed for this system to filter out phishing mails at the perimeter of the organization. A balanced rule set has been used to keep false positive and false negative low.

The proposed strategy has been simulated using C++. The accuracy of the rule set has been compared with existing approach and accuracy of the proposed rule set has been tested on publicly available phishing and non phishing mails. The relation between number of rules and false positive and false negative is also explored.

Table of Contents

Candidate's Declaration & Certificate.....	i
Acknowledgements.....	ii
Abstract.....	iii
Table of Contents.....	iv
List of Figures.....	vii
List of Tables.....	viii
1. Introduction and Statement of the Problem	1
1.1 Introduction.....	1
1.2 Motivation.....	2
1.3 Statement of the Problem.....	3
1.4 Organization of the Report.....	3
2. Background	
2.1 An Introduction to Phishing	4
2.2 Types of Phishing Attacks	4
2.3 Defense Mechanisms	7
2.4 Client-Side Solution	7
2.4.1 Desktop Protection Agents	8
2.4.2 Email Sophistication	10
2.4.3 Browser Capabilities	12
2.4.4 Digitally Signed Email	13
2.4.5 Customer Vigilance	15
2.5 Server-Side Solution.....	16
2.5.1 Customer Awareness	16
2.5.2 Validating Official Communications	18

3. Related Work	20
3.1 Literature Review	20
3.1.1 Spam Assassin	20
3.1.2 Spoof Guard	20
3.1.3 PILFER	21
3.1.4 HoneyTank	22
3.1.5 Phishwish	22
3.1.6 IDS Based Phishing Attack Detection	24
3.2 Research Gaps	24
4. Proposed System	26
4.1 Overview of the System	26
4.1.1 SMTP Server	26
4.1.2 E-mail Filter	30
4.1.3 Rule Base for Filter	32
4.1.4 Decision Making by Rule Set	34
4.1.5 Database.....	35
4.2 Data Set.....	35
4.3 Calculation for False Positive and False Negative	37
5. Simulation	38
5.1 Overview of Simulation	38
5.2 Routine for IP and Character matcher	39
5.3 Implementation Details of Different Rules	39
5.3.1 Implementation of Rule 1- IP Based URLs	39
5.3.2 Implementation of Rule 2 Newly registered domain names	40
5.3.3 Implementation of rule 3- Non matching URLs	41
5.3.4 Implementation of Rule 4 and Rule no 5	43
5.3.5 Implementation of rule 6- Number of dots in a url	43

6. Results and Analysis	43
6.1 Performance of Individual Rules	43
6.2 Overall Performance of the Rule Set	45
6.3 Positive vs Number of Rules	46
6.4 False Negative vs Number of Rules	47
6.5 Comparison with IDS Based Anti Phishing System	48
7. Conclusion and Future Work	49
7.1 Conclusion	49
7.2 Future Work	50
REFERENCES.....	51
APPENDIX	i

List of Figures

Figure 1.1	Numbers of Crimeware Websites in 2008	2
Figure 4.1	Design of the Proposed System	26
Figure 4.2	Example of a phishing e-mail message, including a deceptive URL address linking to a scam Web site	30
Figure 4.3	Example of masked URL address	31
Figure 4.4	A Flowchart for Filter Decision Making	35
Figure 5.2	An Example of IP Based Link	40
Figure 5.3	An Example e-mail with Masked Link	42
Figure 6.1	Corpus E-mails Containing Different Features	44
Figure 6.2	Overall Performances of Rules on Phishing and Non Phishing Mails	45
Figure 6.3	Effect of No. of Rules on False Positive	46
Figure 6.4	Effect of Number of Rules on False Negative	47

List of Tables

Table 6.1	Comparison with IDS Based Anti Phishing System	48
-----------	--	----

Chapter 1

Introduction and Statement of the Problem

1.1 Introduction

Phishing is the term used to describe massive e-mails that trick recipients into revealing their personal or company confidential information, such as social security and financial account numbers, account passwords and other identity or security information. These e-mails request the user's personal information as a client of a legitimate entity with a link to a website that looks like the legitimate entity's website or with a form contained in the body of the e-mail. The aim of phishing is to steal a user's identity in order to make fraudulent transactions as if the Phishers were the user. In a typical phishing attempt, you will receive an authentic-looking email message that appears to come from a legitimate business; e.g., bank, online shopping site. It will ask you to divulge or verify personal data such as an account number, password, credit card number or Social Security number. Often the language will be intimidating, e.g. "Your account will be closed or suspended if you don't follow these directions." Although legitimate online banking and e-commerce are very safe, one should always be careful about giving personal financial information over the Internet. It is also possible for one to be phished by mail, telephone or even in person. Security organizations and companies have done research and development on anti-phishing techniques and tools, which include basic changes in the E-mail infrastructure to help lessen Spam, more widespread deployment of anti-Spam, anti-Malware, personal firewall products, privacy protection software, and stronger authentication for electronic transactions, etc. Some of them have good effects on decreasing the number of phishing.

Unfortunately, phishing attacks are growing both in numbers and in complexity. Phishers are always refining their techniques such as using automated tools and Botnets to increase their catch. Phishing Emails are becoming increasingly sophisticated.

1.2 Motivation

The Anti-Phishing Working Group (APWG) website reports that number of crimeware-spreading sites infecting PCs with password-stealing crimeware reached an all time high of 31,173 in December, and 827% increase from January 2008[1]. These attacks are ever increasing and do not show any sign of slowing. These attacks come in the form of emails asking you to reveal your personal data such as login credentials, credit card number and ATM card number etc.

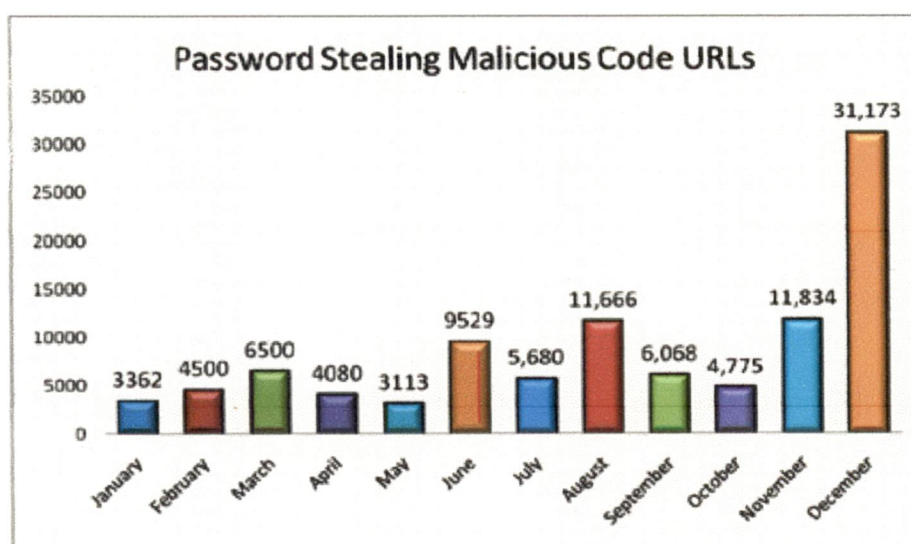


Figure 1.1 Numbers of Crimeware Websites in 2008

In addition to the above information some other information APWG reports

1. Unique phishing reports submitted to APWG recorded a yearly high of 34,758 in December.
2. The number of unique keyloggers and crimeware-oriented malicious applications reached an all-time high in July reaching 1,519 in July.
3. Rogue anti-malware began to rise in July, skyrocketing in December to 9,287.

Though current solutions use strong spam filters to filter out any malware mail still Phishers are quite successful in their business. A lot of research is going on in pursuit of betterment of the technology available; browser based plug-in, phishing detection using IDS are some of the available solution. But they are not perfect and there is need of improvement.

1.3 Problem Statement

In this Dissertation, a rule set has been proposed to classify phishing emails. The chief objective of the dissertation is to propose a minimal rule set so that classifier can filter out Phishing mail with better accuracy.

We have made an attempt to design a system to keep it free from problem such as

1. Dependence on individual user's settings
2. Dependence on web browser policies
3. Non evaluation of mails
4. Bypassing of Client-browser

We have made this attempt by proposing an organization wide server side solution based on a new rule set.

1.4 Organization of the Dissertation Report

This dissertation report comprises of six chapters including this chapter that introduces the topic and states the problem. The rest of the report is organized as follows.

Chapter 2 gives the background of Phishing and Types of Phishing attacks. Chapter discusses in details about solution for these attacks

Chapter 3 This chapter discusses related work in the field of email filtering with their feature and shortcomings; chapter concluded with research gaps.

Chapter 4 gives the design of the system and overview of rule set proposed also discusses rule set's decision making.

Chapter 5 discusses the performance metrics used and the accuracy of the rule set, it has been compared with existing.

Chapter 6 concludes the dissertation work and gives suggestions for future work.

Chapter 2

Background

2.1 An Introduction to Phishing

One of the emerging serious threats against personal data security is phishing. Phishing attacks are often performed by sending out emails that seem to originate from a trusted party. The objective is to deceive the recipient to release sensitive information such as usernames, passwords, banking details, or credentials [15].

Phishing is the term used to describe massive e-mails that trick recipients into revealing their personal or company confidential information, such as social security and financial account numbers, account passwords and other identity or security information. These e-mails request the user's personal information as a client of a legitimate entity with a link to a website that looks like the legitimate entity's website or with a form contained in the body of the e-mail. The aim of phishing is to steal a user's identity in order to make fraudulent transactions as if the Phisher were the user.

2.2 Types of Phishing Attacks

Numerous different types of phishing attacks have now been identified. Some of the more prevalent are listed below.

a) Deceptive Phishing

The term "phishing" originally referred to account theft using instant messaging but the most common broadcast method today is a deceptive email message. Messages about the need to verify account information, system failure requiring users to re-enter their information, fictitious account charges, undesirable account changes, new free services requiring quick action, and many other scams are broadcast to a wide group of recipients with the hope that the unwary will respond by clicking a link to or signing onto a bogus site where their confidential information can be collected.

b) Malware-Based Phishing

refers to scams that involve running malicious software on users' PCs. Malware can be introduced as an email attachment, as a downloadable file from a web site, or by exploiting known security vulnerabilities--a particular issue for small and medium businesses (SMBs) who are not always able to keep their software applications up to date.

c) Keyloggers and Screenloggers

These are particular varieties of malware that track keyboard input and send relevant information to the hacker via the Internet. Keyloggers are programs that install themselves either into a web browser or as a device driver, which monitor data being input and send relevant data to a phishing server [16].

d) Session Hijacking

Describes an attack where users' activities are monitored until they sign in to a target account or transaction and establish their bonafide credentials. At that point the malicious software takes over and can undertake unauthorized actions, such as transferring funds, without the user's knowledge.

e) Web Trojans

Pop up invisibly when users are attempting to log in. They collect the user's credentials locally and transmit them to the phisher.

f) Hosts File Poisoning

When a user types a URL to visit a website it must first be translated into an IP address before it's transmitted over the Internet. The majority of SMB users' PCs running a Microsoft Windows operating system first look up these "host names" in their "hosts" file

before undertaking a Domain Name System (DNS) lookup. By "poisoning" the hosts file, hackers have a bogus address transmitted, taking the user unwittingly to a fake "look alike" website where their information can be stolen.

g) System Reconfiguration

Attacks modify settings on a user's PC for malicious purposes. For example: URLs in a favorites file might be modified to direct users to look alike websites. For example: a bank website URL may be changed from "bankofabc.com" to "bancofab.com".

h) Data Theft

Unsecured PCs often contain subsets of sensitive information stored elsewhere on secured servers. Certainly PCs are used to access such servers and can be more easily compromised. Data theft is a widely used approach to business espionage. By stealing confidential communications, design documents, legal opinions, and employee related records etc., thieves profit from selling to those who may want to embarrass or cause economic damage or to competitors.

i) DNS-Based Phishing ("Pharming")

DNS-based phishing is used here to refer generally to any form of phishing that interferes with the integrity of the lookup process for a domain name. This includes hosts file poisoning, even though the hosts file is not properly part of the Domain Name System. Hosts file poisoning is discussed in the malware section since it involves changing a file on the user's computer [14].

j) Content-Injection Phishing

Describes the situation where hackers replace part of the content of a legitimate site with false content designed to mislead or misdirect the user into giving up their confidential information to the hacker. For example, hackers may insert malicious code to log user's credentials or an overlay which can secretly collect information and deliver it to the hacker's phishing server.

k) Man-in-the-Middle Phishing

This is harder to detect than many other forms of phishing. In these attacks hackers position themselves between the user and the legitimate website or system. They record the information being entered but continue to pass it on so that users' transactions are not affected. Later they can sell or use the information or credentials collected when the user is not active on the system.

l) Search Engine Phishing

This Phishing occurs when Phishers create websites with attractive (often too attractive) sounding offers and have them indexed legitimately with search engines. Users find the sites in the normal course of searching for products or services and are fooled into giving up their information. For example, scammers have set up false banking sites offering lower credit costs or better interest rates than other banks. Victims who use these sites to save or make more from interest charges are encouraged to transfer existing accounts and deceived into giving up their details.

2.3 Defense Mechanisms

As Phisher has a large number of methods at their disposal consequently there is no single solution capable of combating all these different attack vectors. However, it is possible to prevent current and future Phishing attacks by utilizing a mix of information security technologies and techniques. For best protection, these security technologies and techniques must be deployed at **two logical layers**

1. The Client-side – this includes the user's PC.
2. The Server-side – this includes the businesses Internet visible systems and custom applications.

2.4 Client-Side Solution

The client-side should be seen as representing the forefront of anti-phishing security. Given the distributed nature of home computing and the widely varying state of customer skill levels and awareness, client-side security is generally much poorer than a managed corporate workstation deployment. However, many solutions exist for use within both the home and corporate environments. At the client-side, protection against Phishing can be afforded by:

- Desktop protection technologies
- Utilization of appropriate less sophisticated communication settings
- User application-level monitoring solutions
- Locking-down browser capabilities
- Digital signing and validation of email

2.4.1 Desktop Protection Agents

Most users of desktop systems are familiar with locally installed protection software, typically in the form of a common anti-virus solution. Ideally, desktop systems should be configured to use multiple desktop protection agents (even if this functionality duplicates any corporate perimeter protection services), and be capable of performing the following services:

- Local Anti-Virus protection
- Personal Firewall
- Personal IDS
- Personal Anti-Spam
- Spyware Detection

Many desktop protection software providers (e.g. Symantec, McAfee, Microsoft, etc.) now provide solutions that are capable of fulfilling one or more of these functions. Specific to phishing attack vectors, these solutions (or a combination of) should provide the following functionality:

- The ability to detect and block “on the fly” attempts to install malicious software (such as Trojan horses, key-loggers, screen-grabbers and creating backdoors) through email attachments, file downloads, dynamic HTML and scripted content.
- The ability to identify common Spam delivery techniques and quarantine offending messages.
- The ability to pull down the latest anti-virus and anti-spam signatures and apply them to the intercepting protection software. Given the variety in spamming techniques, this process should be scheduled as a daily activity.
- The ability to detect and block unauthorised out-bound connections from installed software or active processes. For example, if the customers host has been previously compromised the protection solution must be able to query the authenticity of the out-bound connection and verify it with the user.
- The ability to detect anomalies in network traffic profiles (both inbound and outbound) and initiate appropriate counter-measures. For instance, detecting that an inbound HTTP connection has been made and substantial outbound SSL traffic begins on a non-standard port [20].
- The ability to block inbound connections to unassociated or restricted network ports and their services.
- The ability to identify common Spyware installations and the ability to prevent installation of the software and/or blocking outbound communications to known Spyware monitoring sites.
- Automatically block outbound delivery of sensitive information to suspected malicious parties. Sensitive information includes confidential financial details and contact information. Even if the customer cannot visually identify the true web-site that will receive the sensitive information, some off the shelf software solutions can.

Advantages

- Local Defence Awareness - Local installation of desktop protection agents is becoming an easier task, and most customers already appreciate the value of anti-

virus software. It is a simple conceptual process to extend this cover to other protection agents and get customers to “buy-in”.

- Protection Overlapping - Using a variety of desktop protection agents from various software manufacturers tends to cause overlaps in overall protection. This means that a failure or security lapse in one product may be detected and defended against by another.
- Defence-in-Depth - The independent nature of desktop protection agents means that they do not affect (or are affected by) security functionality of other externally hosted services – thereby contributing to the overall defence-in-depth posture of an organisation.

Disadvantages

- Purchasing Price - The purchasing price of desktop protection agents is not an insignificant investment for many customers. If multiple vendors’ solutions are required to provide coverage against all attack vectors, there can be a substantial multiplication of financial cost for very little extra security coverage.
- Subscription Renewals - Many of the current desktop protection agents rely on monthly or annual subscription payments to keep the users installation current. Unless appropriate notices are given, these renewals may not take place and the protection agents will be out of date.
- Complexity & Manageability - For corporate environments, desktop protection agents can be complex to deploy and manage – particularly at an enterprise level. Since these solutions require continual deployments of updates (sometimes on a daily schedule), there may be a requirement of an investment in additional man-power.

2.4.2 Email Sophistication

Many of the email applications corporate users and customers use to access Internet resources provide an ever increasing level of functionality and sophistication. While some of this functionality may be required for sophisticated corporate applications and systems – use of these technologies typically only applies to inter-company systems.

Most of this functionality is not required for day-to-day use – particularly for Internet communication services.

This unnecessary embedded (and often default) functionality is exploited by Phishing attacks (along with increasing the probability of other kinds of attacks). In general, most popular applications allow users to turn off the most dangerous functionality.

a) HTML-based Email

Many of the attacks outlined in Section 2 are successful due to HTML-based email functionality, in particular the ability to obfuscate the true destination of links, the ability to embed scripting elements and the automatic rendering of embedded (or linked) multimedia elements. HTML functionality must be disabled in all email client applications capable of accepting or sending Internet emails. Instead plain-text email representation should be used, and ideally the chosen font should be fixed-width such as Courier.

Emails will then be rendered in plain-text, preventing the most common attack vectors. However, users should be prepared to receive some emails that appear to be “gobbledygook” due to textual formatting issues and probable HTML code inclusions. Some popular email clients will automatically remove the HTML code. While the visual appeal of the received emails may be lessened, security is improved substantially.

Users should not use other email rendering options (such as Rich-text or Microsoft Word editors) as there are known security flaws with these formats which could also be exploited by Phishers.

b) Attachment Blocking

Email applications capable of blocking “dangerous” attachments and preventing users from quickly executing or viewing attached content should be used whenever possible. Some popular email applications (such as Microsoft Outlook) maintain a list of

“dangerous” attachment formats, and prevent users from opening them. While other applications force the user to save the file somewhere else before they can access it.

Ideally, users should not be able to directly access email attachments from within the email application. This applies to all attachment types (including Microsoft Word documents, multimedia files and binary files) as many of these file formats can contain malicious code capable of compromising the associated rendering application (e.g. the earlier example of a vulnerability in the RealPlayer .RM player). In addition, by saving the file locally, local anti-virus solutions are better able to inspect the file for viruses or other malicious content.

Advantages

- Overcomes HTML Obfuscation - Forcing all inbound emails into text-only format is sufficient to overcome standard HTML-based obfuscation techniques.
- Overcoming Attached Viruses - By blocking attachments, and/or forcing content to be saved elsewhere, it makes more difficult for automated attacks to be conducted and provides extra potential for standard anti-virus products to detect malicious content.

Disadvantages

- Readability - The rendering of HTML-based emails often means that HTML code elements make the message difficult to read and understand.
- Message Limitations - Users often find it difficult to include attachments (such as graphics) in TEXT-only emails having been used to drag-and-drop embedding of images into to HTML or Microsoft Word email editors.
- Onerous Blocking - The default blocking of “dangerous” attachments often results in technical users attempting to bypass these limitations in commercial environments that are used to attaching or receiving executable content.

2.4.3 Browser Capabilities

The common web browser may be used as a defense against phishing attacks – if it is configured securely. Similar to the problems with email applications, web browsers also offer extended functionality that may be abused (often to a higher degree than email

clients). For most users, their web browser is probably the most technically sophisticated application they use.

The most popular web browsers offer such a fantastic array of functionality – catering to all users in all environments – that they unintentionally provide gaping security flaws that expose the integrity of the host system to attack (it is almost a weekly occurrence that a new vulnerability is discovered that may be exploited remotely through a popular web browser). Much of the sophistication is devoted to being a “jack of all trades”, and no single user can be expected to require the use of all this functionality.

Customers and businesses must make a move to use a web browser that is appropriate for the task at hand. In particular, if the purpose of the web browser is to only browse Internet web services, a sophisticated web browser is not required.

To help prevent many Phishing attack vectors, web browser users should:

- Disable all window pop-up functionality
- Disable Java runtime support
- Disable ActiveX support
- Disable all multimedia and auto-play/auto-execute extensions
- Prevent the storage of non-secure cookies
- Ensure that any downloads cannot be automatically run from the browser, and must instead be downloaded into a directory for anti-virus inspection

2.4.4 Digitally Signed Email

It is possible to use Public Key cryptography systems to digitally sign an email. This signing can be used to verify the integrity of the messages content – thereby identifying whether the message content has been altered during transit. A signed message can be attributed to a specific users (or organizational) public key. Almost all popular email client applications support the signing and verification of signed email messages. It is recommended that users:

- Create a personal public/private key pair

- Upload their public key to respected key management servers so that other people who may receive emails from the user can verify the messages integrity
- Enable, by default, the automatic signing of emails
- Verify all signatures on received emails and be careful of unsigned or invalid signed messages – ideally verifying the true source of the email

A message signature is essentially a sophisticated one-way hash value that uses aspects of the sender's private key, message length, date and time. The email recipient uses the public key associated with the email sender's address to verify this hash value. The contents of the email should not be altered by any intermediary mail servers. It is important to note that, in general, there are no restrictions on creating a public/private key pair for any email address a person may choose and consequently uploading the public key to an Internet key management server. Therefore it is still possible for a Phisher to send forth an email with a spoofed address and digitally sign it with a key that they own.

S/MIME

An S/MIME digital signature allows an email recipient to verify that the "From:" address in a message has not been forged or 'spoofed', by checking two things [21]:

- The "From:" address is correct (e.g. 'visa.com' and not 'visa-security.com')
- The digital signature is valid, this is indicated by some type of visual cue

S/MIME, PGP/MIME, and OpenPGP signed mail

Although they offer similar services to email users, the two methods have very different formats. Further, and more important to corporate users, they have different formats for their certificates. This means that not only can users of one protocol not communicate with the users of the other; they also cannot share authentication certificates.

Key points for S/MIME and PGP:

- S/MIME was originally developed by RSA Data Security, Inc. It is based on the PKCS #7 data format for the messages, and the X.509v3 format for certificates. PKCS #7 is based on the ASN.1 DER format for data.
- PGP/MIME is based on PGP, which was developed by many individuals, some of whom have now joined together as PGP, Inc. The message and certificate formats were created from scratch and use simple binary encoding. OpenPGP is also based on PGP.
- S/MIME, PGP/MIME, and OpenPGP use MIME to structure their messages. They rely on the multipart/signed MIME type that is described in RFC 1847 for moving signed messages over the Internet.

2.4.5 Customer Vigilance

Customers may take a number of steps to avoid becoming a victim of a phishing attack that involve inspecting content that is presented to them and questioning its authenticity.

General vigilance includes:

- If you get an email that warns you, with little or no notice, that an account of yours will be shut down unless you reconfirm billing information, do not reply or click on the link in the email. Instead, contact the company cited in the email using a telephone number or Web site address you know to be genuine.
- Never respond to HTML email with embedded submission forms. Any information submitted via the email (even if it is legitimate) will be sent in clear text and could be observed.
- Avoid emailing personal and financial information. Before submitting financial information through a Web site, look for the "lock" icon on the browser's status bar. It signals that your information is secure during transmission.
- For sites that indicate they are secure, review the SSL certificate that has been received and ensure that it has been issued by a trusted certificate authority. SSL certificate information can be obtained by double-clicking on the "lock" icon at the bottom of the browser, or by right-clicking on a page and selecting properties.

- Review credit card and bank account statements as soon as you receive them to determine whether there are any unauthorized charges. If your statement is late by more than a couple of days, call your credit card company or bank to confirm your billing address and account balances.

2.5 Server-side Solution

By implementing intelligent anti-phishing techniques into the organizations web application security, developing internal processes to combat phishing vectors and educating customers – it is possible to take an active role in protecting customers from future attack. By carrying out this work from the server-side, organizations can take large steps in helping to protect against what is invariably a complex and insidious threat.

At the client-side, protection against Phishing can be afforded by:

- Improving customer awareness
- Providing validation information for official communications
- Ensuring that the Internet web application is securely developed and doesn't include easily exploitable attack vectors
- Using strong token-based authentication systems
- Keeping naming systems simple and understandable

2.5.1 Customer Awareness

It is important that organizations constantly inform their customers and other application users of the dangers from Phishing attacks and what preventative actions are available. In particular, information must be visible about how the organization communicates securely with their customers. For instance, a posting similar to the following will help customers identify phishing emails sent in the organizations name.

"MyBank will never initiate a request for sensitive information from you via email (i.e., Social Security Number, Personal ID, Password, PIN or account number). If you receive an email that requests this type of sensitive information, you should be suspicious of it. We strongly suggest that you do not share your Personal ID, Password, PIN or account number with anyone, under any circumstances. If you suspect that you have received a fraudulent email, or wish to validate an official email from MyBank, please visit our anti-

phishing page "<http://mybank.com/antiphishing.aspx>" Key steps in helping to ensure customer awareness and continued vigilance:

- Remind customers repeatedly. This can be achieved with small notifications on critical login pages about how the organization communicates with their customers. Customers reaching the page should be prompted to think about the legitimacy of the email (or other communication) that drove them to the page.
- Provide an easy method for customers to report phishing scams, or other possible fraudulent emails sent in the organizations name. This can be achieved by providing clear links on key authentication and help pages that enable customers to report a possible phishing scam – and also provide advice on recognising a scam. Importantly, the organization must invest in sufficient resources to review these submissions and be capable of working with law enforcement agencies and ISP's to stop an attack in progress.
- Provide advice on how to verify the integrity of the website they are using. This includes how to:
 - Check the security settings of their web browser
 - Check that their connection is secure over SSL
 - Review the "padlock" and certificate signature of the page
 - Decipher the URL line in their browser
- Establish corporate communication policies and enforce them. Create corporate policies for email content so that legitimate emails cannot be confused with phishing attacks. Ensure that the departments likely to communicate with customers clearly understand the policy and take steps to enforce them (e.g. perimeter content checking systems, review by QA teams, etc.). To be effective, organizations must ensure that they are sending a clear, concise and consistent message to their customers. For example, don't post announcements claiming to "never prompt users to fill in forms in an email" one day and then send out an email request for online bill payment the following day, which includes a login form in the email.
- Respond quickly and clearly about phishing scams that have been identified. It is important that customers understand that the threat is real and, importantly, how the

organization is working to protect them against attack. However, organizations must take care not to swamp customers with information.

2.5.2 Validating Official Communications

Steps may be taken by an organization to help validate official customer communications and provide a means for identifying potential phishing attacks. Tied closely with the customer awareness issues already discussed, there are a number of techniques an organization may apply to official communications, however care must be taken to only use techniques that are appropriate to the audience's technical ability and value of transactions.

a) Email Personalization

Emails sent to customers should be personalized for the specific recipient. This personalization may range from the use of the customer's name, or reference some other piece of unique information shared between the customer at the organization.

Examples include:

- “Dear Mr Smith” instead of “Dear Sir,” or “Our valued customer”
- Credit card account holder “**** *32 6722” (ensure that only parts of confidential information are used)
- Referencing the initiating personal contact such as “your account manager Mrs Jane Doe...” Organizations must ensure that they do not leak other confidential details about the customer (such as full address details, passwords, individual account details, etc.) within their communications. Previous Message Referral

It is possible to reference a previous email that was sent to the customer – therefore establishing a trail of trust in communications. This may be achieved through various means.

The most common methods are:

- Clearly referencing the subject and date of the previous email.
- Providing a sequential number to the email.

While these methods of email referral are valuable, they are also complex for the customer to validate. There are no guarantees that the customer still retains access to a previous email to verify the sequence – and is especially so if the organization sends the customer a high volume of emails, or frequent advertising-type messages.

b) Digital Signatures

The use of digital certificates to sign messages is recommended. However, care must be taken to educate customers on their use and understand how to validate signatures.

- **Web Application**
- **Validation Portals**

A successful method of providing reassurance to customers on the authenticity of a communication, and subsequently providing the ability to identify a new phishing attack, is to provide a portal on the corporate website. The web portal exists to allow customer to copy/paste their received message content to an interactive form, and for the application to clearly display the authenticity of the message.

If the message fails the authenticity checks, the message should be manually verified by the organization to evaluate whether the message contains a malicious phishing attack. Similarly, an interface should be provided in which customer can copy/paste suspicious URL's that they have received. The application then validates whether this is a legitimate URL relating to the organization.

Visual or Audio personalization of email

It is possible to embed personalized visual or audio data within an email. This material would have been supplied by the customer previously, or contain the equivalent of a shared secret.

However, this method is not recommended as it may be rendered ineffectual through the enforcement of non-HTML or attachment emails at the customer side.

Chapter 3



Related Work

3.1 Literature Review

Current solutions use strong spam filters to isolate phishing solicitations or capture phishing sites at the browser. To name a few are

- Spam Assassin
- SpoofGuard
- Pilfer
- HoneyTank
- Phishwish
- IDS based phishing attack detection

3.1.1 Spam Assassin

Spam Assassin is a tool that recognizes spam containing phishing email. Fette et al.[3] states that SpamAssassin has a false negative of 15% for spam e-mails, and performs worst when tested with 10 fold cross validation. SpamAssassin uses a wide range of heuristic tests on mail headers in order to identify spam, and can be An Intrusion Detection System for Detecting Phishing Attacks 185 customized. For algorithms, it uses text analysis, Bayesian filtering, DNS blocklists, a collaborative filtering database, and a Stochastic Gradient Descent method in training a neural network. This is used for its scoring based on perception that uses a single perception with a log sig activation function that maps the weights to Spam Assassin's score space. SpamAssassin does not delete email from mail boxes, but it can route classified e-mail to mail boxes or folders[3].

3.1.2 Spoof Guard

Chou et al. [4] proposes a browser-based plug-in, SpoofGuard, that monitors users Internet activities and warns if the tool classifies a visiting web-site as a phishing page. SpoofGuard uses the observation that a page is loaded from an e-mail message and whether the URL was visited before. The authors propose the use of the following properties: (1) Logos – use of images. (2) Suspicious URLs – urls that contains IP

address or higher length urls. (3) User Input – pages that has form input. (4) Short lived – the spoof sites are shut down within 2 – 3 days. (5) Copies – similar contents. (6) Sloppiness or lack of familiarity with English – misspellings and grammar errors.

SpoofGuard uses 3 methods to determine impersonation: (1) a stateless method that determines whether a downloaded page is suspicious, (2) a stateful method that evaluates a downloaded page in light of previous user activity, and (3) a method that evaluates outgoing post data. SpoofGuard uses a standard aggregate function to calculate the total spoof score (TSS) computed as: $TSS(\text{page}) = \sum_{i=1}^n w_i P_i + \sum_{i=1}^n \sum_{j=1}^n w_{i,j} P_i P_j + \sum_{i=1}^n \sum_{j=1}^n \sum_{k=1}^n w_{i,j,k} P_i P_j P_k \dots$

For a given downloaded web page and a browser state TSS produce a number P_i within $[0,1]$ where 1 indicates a page more likely to be a spoof page. The w_i 's are preset weight to minimize false positives. SpoofGuard has a configuration pop-up screen that requires a user defined spoof rating threshold. This allows setting independent weights and security levels for the domain name, url, link, password and image checks. The user interface alerts suspicious sites with a traffic light symbol lighting for the degree of the probable spoof activity. The information which was based for classifying is available for the user. Even though a link from an e-mail is a good method for phishing detection, a user clearing the browser history could result in many false positives. Sensitivity decreasing on this system would result in false negatives while increasing would result in false positives.

3.1.3 PILFER

It is a machine-learning based approach to classification [3]. PILFER decides whether some communication is deceptive, i.e. whether it is designed to trick the user into believing they are communicating with a trusted source, when in reality the communication is from an attacker. It makes this decision based on information from within the email or attack vector itself (an internal source), combined with information from external sources. This combination of information is then used as the input to a classifier, the result of which is a decision on whether the input contained data designed to deceive the user. With respect to email classification, it has two classes, namely the

class of phishing emails, and the class of good (“ham”) emails. It identifies some of the email as Phishing email based on some features. These features are

- (i) IP based URLs
- (ii) Age of linked-to domains
- (iii) Non matching URLs
- (iv) “Here” links to nonmodal domain

3.1.4 HoneyTank

Honey Tank collects Spam using a Honeynet[11,12] and automatically generates a pattern. The pattern is to be used by a network based intrusion detection systems. A HoneyTank is a workstation receiving TCP segments sent to unallocated IP addresses and replying to those segments to emulate real end systems that supports TCP services. They use Advanced Sequential Analyzer on Unix (ASAX) as the intrusion detection system. ASAX is a generic system that analyzes sequential files like security audits trails. It is composed with three parts which are analyzer, rule declarations, and format adaptor. The analyzer receives the input from the format adaptor and analyzes according to the declared rules [5].

3.1.5 Phishwish

Its primary goal is to minimize the complexity of the rule-base and configuration, and maximize the number of phishing emails detected while minimizing the number of false positives. Phishwish [7] is applicable to emails that instruct the recipient to log into a web site. It processes text based and HTML formatted emails, although some rules are only applicable to HTML. Each rule is assigned a configurable weight, W_i and a flag X_i . Phishwish sets X_i to 1 if the rule is applicable to the email and to 0 otherwise. Each rule produces a value, P_i , ranging from 0.0 - 1.0. If the rule is not applicable, $P_i = 0$. The final

score is $S = \frac{\sum W_i P_i}{\sum W_i X_i}$, with higher values of S indicating a greater probability of phishing.

When describing the rules, a positive result is indicative of phishing, in which case P_i is set to 1 except for rules 8 and 10 where it is set to a fraction. A negative result is indicative of a valid email, in which case P_i is set to 0. Business refers to the business

from which the email supposedly has been sent. LoginURL refers to the URL within the email that the recipient should use to access the business' login page. The rules fall into the following general categories:

- (1) Identification and analysis of the login URL in the email
- (2) Analysis of the email headers
- (3) Analysis across URLs and images in the email
- (4) Determining if the URL is accessible

These rules are:

Rule 1: If the email appears (based on search engine results) to not be directing the recipient to the actual login page for the business, the result is positive.

Rule 2: In HTML formatted emails, if a URL displayed to the recipient uses TLS, it is compared to the URL in the HREF tag. If the URL in the tag does not use TLS, the result is positive.

Rule 3: If the login URL is referenced as a raw IP address instead of a domain name, the result is positive.

Rule 4: If the business name appears in the login URL, but not in the domain portion, the result is positive.

Rule 5: In HTML formatted emails, if a URL is displayed to the recipient, it is compared to the URL in the HREF tag. If their domains do not match, the result is positive.

Rule 6: The chain of "Received" SMTP headers is checked to determine if the path includes a server or a mail user agent in the same DNS domain as the business. The rule is positive if such a Received header is not present.

Rule 7: Rules 7 and 9 perform a case-insensitive byte-wise comparison of the domain of all URLs in the email message with the domain of the login URL. Rule 7 analyzes non-image URLs for such inconsistencies in their domains. If inconsistencies are detected, the rule is positive. Rule 8: Rules 8 and 10 match the DNS registrant for the domain of each URL in the email with the DNS registrant for the domain in the login URL. Rule 8 analyzes non-image URLs for inconsistencies in their whois registrant information. P8 is set to the percentage of URLs whose information differs from that of the login URL.

Rule 9: This rule analyzes image URLs for inconsistencies in their domains. If inconsistencies are detected, the rule is positive.

Rule 10: This rule analyzes image URLs for inconsistencies in their whois registrant information. P10 is set in the same manner as P8 in Rule 8.

Rule 11: The rule is positive if the web page is inaccessible. The rule is considered not applicable otherwise.

3.1.6 IDS Based Phishing Attack Detection

This solution proposed by Hasika Pamunuwa et al. uses IDS to detect phishing attacks [3]. This System architecture has two parts. First part of the architecture seeks emails from the outside of the world and forwards it to the IDS which act as a filter. Once filtering is done; identified phishing emails are saved in database. Second part of the system is validation system it crawls backs the addresses of the suspected emails and validates whether the mail is a genuine phishing mail or not.

If we talk about its filtering system, it uses open source IDS Snort as a filter which on the following rules identifies emails as a phishing email:

1. HTML encoded in e-mail.
2. Any URL including IP addresses.
3. URLs that has been masked with HTML to a different address

3.2 Research Gaps

As discussed above most of these solutions are either client side solution a browser based plug-in or spam filter.

First solution Spam Assassin[8] is a tool for detecting email at the server side which has weakness that it has false negative of 15%.

Second solution Spoof Guard[4] is a browser based plug-in and can be bypassed by the attackers, it keeps track of browser history to be used in phishing detection, a user clearing the browser history could result in many false positives. As it takes user defined setting, sensitivity decreasing on this system would result in false negatives while increasing would result in false positives.

Fette et al. [3] proposes PILFER classify phishing email with a true positive rate of 92% and a false positive rate of 0.1%.

The solution proposed by Hansika et al. uses IDS to detect phishing attacks. Though it has the advantage that it is an organization wide solution but its filtering algorithm is very primitive.

This whole observation can be put into the following points

1. There should not be user dependence
2. All the traffic should be evaluated
3. False positive and false negative should be low
4. Filtering rule set should be accurate

Chapter 4

Proposed System

4.1 Overview of the System

In this part of our report we have discussed the overall architecture of the system. First component of our system is a SMTP server which seeks email from outside as shown in figure. Second part of it is email filter which is the most important part of the architecture, upon receiving the emails from the server email filter on the basis of some predefined rules identifies some of the mail as phishing mail at this point it forwards the suspicious mail to third part of the system the database.

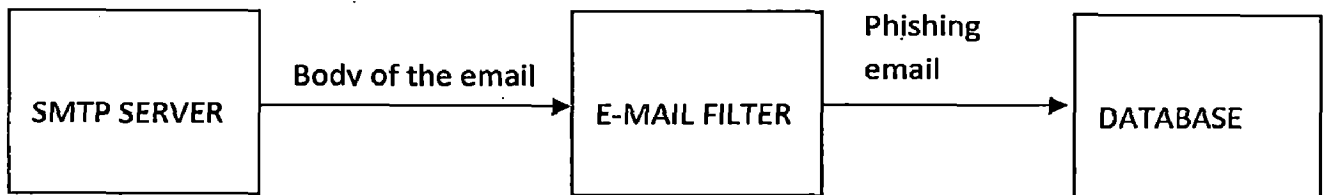


Figure 4.1 Design of the Proposed System

Each part of the system in more details:

4.1.1 SMTP Server

It is an Internet standard for electronic mail (e-mail) transmission across Internet Protocol (IP) networks. While electronic mail servers and other mail transfer agents use SMTP to send and receive mail messages, user-level client mail applications typically only use SMTP for sending messages to a mail server for relaying. For receiving messages, client applications usually use either the Post Office Protocol (POP) or the Internet Message Access Protocol (IMAP) to access their mail box accounts on a mail server.

SMTP is a relatively simple, text-based protocol, in which a mail sender communicates with a mail receiver by issuing simple command strings and supplying necessary data over a reliable ordered data stream channel, typically a Transmission Control Protocol (TCP) connection. An SMTP session consists of a series of commands, initiated by the SMTP client, and responses from the SMTP server through which the session is opened, operating parameters are exchanged, the recipients are specified, and possibly verified, and the message is transmitted, before the session is closed. The originating host is either an end-user's email client also known as mail user agent (MUA), or a relay server's mail transfer agent (MTA). SMTP was designed as an electronic mail transport and delivery protocol, and as such it is used between SMTP systems that are operational at all times. However, it has capabilities for use as a mail submission protocol for email clients (split user-agent) that do not have the capability to operate as MTA. Such agents are also called message submission agents (MSA)[6], sometimes also referred to as mail submission agents. They are typically end-user applications and send all messages through a smart relay server, often called the outgoing mail server, which is specified in the programs' configuration. A mail transfer agent, incorporated either in the e-mail client directly or in the relay server, typically determines the destination SMTP server by querying the Domain Name System for the mail exchanger (MX record) of each recipient's domain name. Conformant MTAs fall back to a simple address lookup (A record) of the domain name when no mail exchanger is available. In some cases an SMTP client, even a server, may also be configured to use a smart host for delivery. The SMTP client typically initiates a Transmission Control Protocol (TCP) connection to the SMTP server on the well-known port designated for SMTP, port number 25. SMTP is a delivery protocol only. It cannot pull messages from a remote server on demand. Other protocols, such as the Post Office Protocol (POP) and the Internet Message Access Protocol (IMAP) are specifically designed for retrieving messages and managing mail boxes. However, the SMTP protocol has a feature to initiate mail queue processing on a remote server so that the requesting system may receive any messages destined for it (cf. #Remote Message Queue Starting). POP and IMAP are preferred protocols when a user's personal computer is only intermittently powered up, or Internet connectivity is only transient and hosts cannot receive message during off-line periods.

SMTP transport example

A typical example of sending a message via SMTP to two mailboxes (alice and theboss) located in the same mail domain (example.com) is reproduced in the following session change.

```
S: 220 smtp.example.com ESMTP Postfix
C: HELO relay.example.org
S: 250 Hello relay.example.org, I am glad to meet you
C: MAIL FROM:<bob@example.org>
S: 250 Ok
C: RCPT TO:<alice@example.com>
S: 250 Ok
C: RCPT TO:<theboss@example.com>
S: 250 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
C: From: "Bob Example" <bob@example.org>
C: To: Alice Example <alice@example.com>
C: Cc: theboss@example.com
C: Date: Tue, 15 Jan 2008 16:02:43 -0500
C: Subject: Test message
C:
C: Hello Alice.
C: This is a test message with 5 header fields and 4 lines in the message
body.
C: Your friend,
C: Bob
C: .
S: 250 Ok: queued as 12345
C: QUIT
S: 221 Bye {The server closes the connection}
```

For illustration purposes here (not part of protocol), the protocol exchanges are prefixed for the server (S:) and the client (C:).

After the message sender (SMTP client) establishes a reliable communications channel to the message receiver (SMTP server), the session is opened with a greeting by the server, usually containing its fully qualified domain name, in this case smtp.example.com. The client initiates its dialog by responding with a HELO command identifying itself in the command's parameter.

The client notifies the receiver of the originating e-mail address of the message in a MAIL FROM command. In this example, the email message is sent to two mailboxes on the same SMTP server: one each for each recipient listed in the To and Cc header fields. The corresponding SMTP command is RCPT TO. Each successful reception and execution of a command is acknowledged by the server with a result code and response message (e.g., 250 Ok).

The transmission of the body of the mail message is initiated with a DATA command after which it is transmitted verbatim line by line and is terminated with a characteristic sequence of a new line (<CR><LF>) with just a single full stop (period) followed by another line indication (<CR><LF>). The QUIT command ends the session.

The information that the client sends in the HELO and MAIL FROM commands are added (not seen in example code) as additional header fields to the message by the receiving server. It adds a Received and Return-Path header field, respectively.

In our proposed system what SMTP server does is it takes up the data part of the email and forwards it to the filter. This can be done by using some open source SMTP server and manipulating its DATA command to forward data part of email to email filter.

4.1.2 E-mail Filter

The most important part of our system is E-mail filter around which everything revolves. This part of the architecture filters out some of the incoming emails from SMTP server as a phishing emails. It does so using some predefined rule set. These rule sets are basically some common feature which most of the emails hold. We now will discuss in details about these properties. Phishing mails possess some common traits to deceive people so that they can be convinced to reveal ones personal data. These personal data can be credit card numbers, passwords, account data, or other information.

These e-mail messages that appear to come from Web sites you trust, like your bank or credit card Company, and request that you provide personal information.

What does a phishing mail look like?

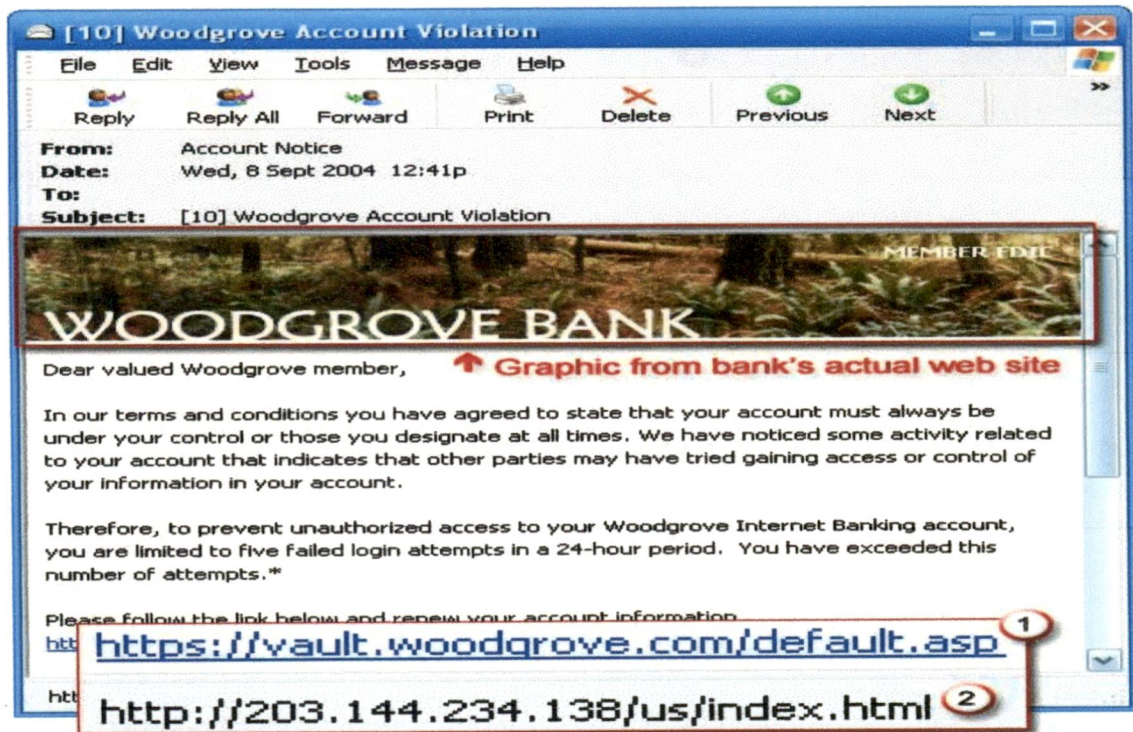


Figure 4.2 Example of a phishing e-mail message, including a deceptive URL address linking to a scam Web site

As Phishers become more sophisticated, so do their phishing e-mail messages and pop-up windows. They often include official-looking logos from real organizations and other identifying information taken directly from legitimate Web sites. The following is an example of what a phishing scam e-mail message might look like.

To make these phishing e-mail messages look even more legitimate, the Phishers may place a link in them that appears to go to the legitimate Web site (1), but it actually takes you to a fake scam site (2) or possibly a pop-up window that looks exactly like the official site as shown in figure 4.2.

These copycat sites are also called "spoofed" Web sites. Once you're at one of these spoofed sites, you might unwittingly send personal information to the con artists.

Here are a few phrases to look for if you think an e-mail message is a phishing scam.

1. "Verify your account"

Businesses should not ask you to send passwords, login names, Social Security numbers, or other personal information through e-mail. If you receive an e-mail from Microsoft asking you to update your credit card information, do not respond: this phishing scam

2. "If you don't respond within 48 hours, your account will be closed"

These messages convey a sense of urgency so that you'll respond immediately without thinking. Phishing e-mail might even claim that your response is required because your account might have been compromised.

3. "Dear Valued Customer"

Phishing e-mail messages are usually sent out in bulk and often do not contain your first or last name.

4. "Click the link below to gain access to your account."

HTML-formatted messages can contain links or forms that you can fill out just as you would fill out a form on a Web site. The links that you are urged to click may contain all or part of a real company's name and are usually "masked," meaning that the link you see does not take you to that address but somewhere different, usually a fake Web site. Notice in the following example Figure 4.3 that resting the mouse pointer on the link reveals the real Web address, as shown in the box with the yellow

background. The string of cryptic numbers looks nothing like the company's Web address, which is a suspicious sign.

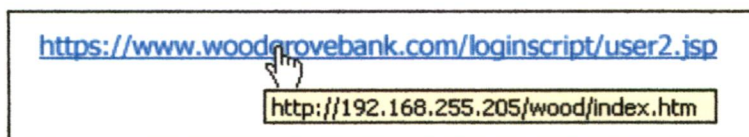


Figure 4.3 Example of masked URL address

As from the above discussion we can see how emails can be tailored to look authentic. Exploiting these common traits of the Phishing mails we have used some rule base; On the basis of those rules we will filter out Phishing mails.

4.1.3 Rule Base for Filter

Rules used to filter emails are as follows

a) IP based URLs

One way to obscure a server's identity is achieved through the use of an IP address. Use of an IP address makes it difficult for users to know exactly where they are being directed to when they click the link[]. These attacks are generated from compromised system. These machines generally do not possess DNS entries, and the simplest way to refer to them is by IP address. Companies never link to pages by an IP-address, and so such a link in an email is a potential indication of a phishing attack. As such, anytime we see a link in an email whose host is an IP-address (such as [http://192.168.0.1/paypal.cgi?fix account](http://192.168.0.1/paypal.cgi?fix+account)), we flag the email as having an IP-based URL. As phishing attacks are becoming more sophisticated, IP-based links are becoming less prevalent, with attackers purchasing domain names to point to the attack website instead. However, there are still a significant number of IP-based attacks, and therefore this is still a useful feature.

b) Newly Registered Hosts

Nowadays Phishers have become more sophisticated; they do not give away by using IP based URLs. Name-based attacks, in which a Phisher will register a similar

or otherwise legitimate-sounding domain name (such as paypal.com or paypal-update.com) are increasingly common. These domains often have a limited life, however. Since Phishers always have this fear to get caught. As many companies pay heed to this issue, for example Microsoft keeps track of domain name registrations involving any of their trademarks. We exploit this fact by performing WHOIS query for all those suspected URLs to find out whether they are newly registered domain if it is found to be less than 60 days we mark the email as Phishing email.

c) Nonmatching URLs

Phishers often exploit HTML emails, in which it is possible to display a link that says paypal.com but actually links to badsite.com. For this feature, all links are checked, and if the text of a link is a URL, and the HREF of the link is to a different host than the link in the text, the email is flagged with a “nonmatching URL” feature. Such a link looks like ` paypal.com`.

d) Contains Javascript

JavaScript is used for many things, from creating popup windows to changing the status bar of a web browser or email client. It can appear directly in the body of an email, or it can be embedded in something like a link[18]. Attackers can use JavaScript to hide information from the user, and potentially launch sophisticated attacks. An email is flagged with the “contains javascript” feature if the string “javascript” appears in the email, regardless of whether it is actually in a `<script>` or `<a>` tag

e) HTML emails

HTML-formatted emails are mainly used for phishing attacks, because plaintext emails do not provide for the scale of tricks afforded with HTML-formatted emails. Hyperlinks are active and clickable only in html formatted emails[17]. Most emails are sent as either plain text, HTML, or a combination of the two in what is known as a multipart/alternative format. The email is flagged with the HTML email feature if it

contains a section that is denoted with a MIME type of text/html. (This includes many multipart/alternative emails). While HTML email is not necessarily indicative of a phishing email, it does make many of the deceptions seen in phishing attacks possible. For a Phisher to launch an attack without using HTML is difficult, because in a plain text email there is virtually no way to disguise the URL to which the user is taken. Thus, the user still can be deceived by legitimate-sounding domain names, but many of the technical, deceptive attacks are not possible.

f) Urls Containing Large Numbers of Dots

There are many ways by which attacker can name a domain which are very elusive. Consider this domain name <http://www.ebay.update.data.com>, this sounds very much real that any naïve user may commit a mistake to visit the site. Other way is by using redirection script for example

<http://www.google.com/url?q=http://www.somebadsite.com> seems some legitimate site hosted at Google but actually redirecting you to some phishing site. Since there could be mails where they contain such urls even then they are not phishing emails hence this feature does not confirm identity of the mail as a phishing email. Hence we categorize these mails as suspicious mail and try to impose some other rules on it to clarify its identity as a phishing mail.

4.1.4 Decision Making by Rule Set

The flowchart shown in Figure 4.4 shows how the rule set decides that a given email is phishing email. After getting a mail it tests it with one or more rule before arriving to a conclusion. In rule set we have some rule which on being true for some mail make a direct decision that the mail is Phishing mail; on the other hand we have rules even if they are true for some emails it does not directly classify them as Phishing mail rather it goes for another test to confirm its identity as Phishing mail. Rule number 5 and Rule number 6 are such rule. Once a mail having feature of rule number 5&6 appear we treat them as a suspicious mail and collect all link from them to be tested with rule 2.

We apply all the rules to all the rules one by one until we get some match. Rule 1,2,3 and 4 are individually enough to categorize a mail as a Phishing mail.

A Flowchart of Filter

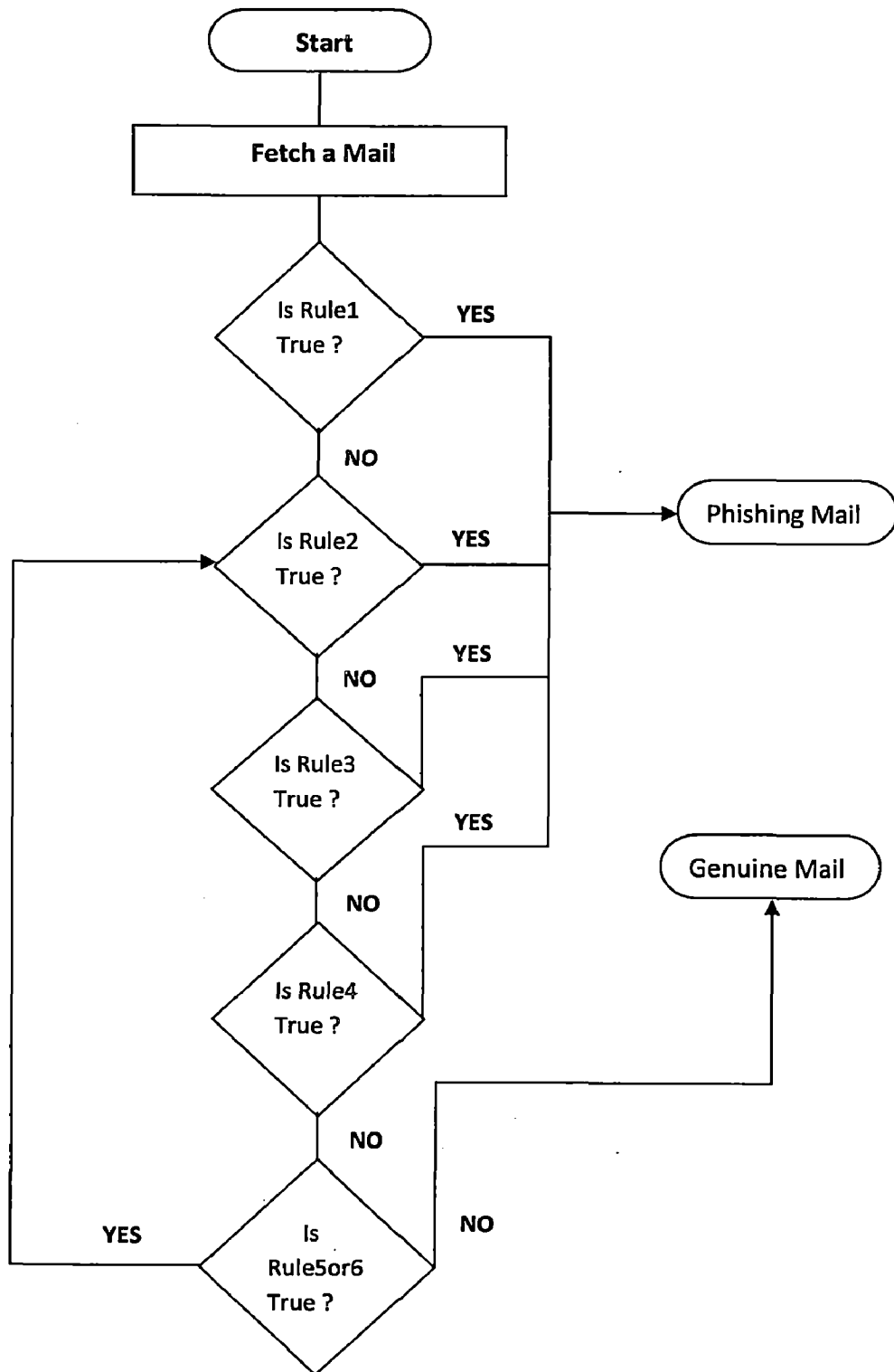


Figure 4.4 A Flowchart for Filter Decision Making

4.1.5 Database

After identifying the phishing mails at the filtering point we simply put them aside on a database. We can use any database such as Microsoft Access or MY SQL. Its purpose is to store emails with date and time.

We can use database such as MySQL to store information about potential phishing messages, where database entries consist of the e-mail contents, time of receipt, and the originating IP address. A daemon process that runs on the same machine scans all URLs listed on flagged e-mail, strips off their HTML tags and identifies if they have been observed earlier.

4.2 Dataset

Two publicly available datasets were used to test our implementation: the ham corpora from the SpamAssassin project [8]. The publicly available phishing corpus [9] (approximately 850 email messages).

We have use five different program modules to extract features discussed section 4.1.3. We label emails as being non-phishing if they come from the SpamAssassin ham corpora, and as phishing if they come from the phishing corpus. For these experiments we used the entire dataset and did not re-label any of its contents.

Many mail clients keep spam filter and they assign class to each mail as “ham” or “spam” we have intentionally removed spam rating from each of the mail so that rule base can be checked for the all the mails individually. Though rule base can achieve better accuracy by using spam filters label.

For non-phishing emails we have used ham corpora from the SpamAssassin project [8]. These are 3000 non-phishing and non spam mails easy and hard.

4.3 Calculation for False Positive and False Negative

Filter may misclassify a good mail as phishing mail or a phishing mail as a good mail and both are non- desirable and have different impact as well. So we report separately the rate of false positives and false negatives. The false positive rate corresponds to the proportion of ham emails classified as phishing emails, and false negative rate corresponds to the proportion of phishing emails classified as ham.

If F_p and F_n are false positive and false negative respectively

Then

$$F_p = \frac{\text{ham}(\text{phishing mail})}{\text{ham}(\text{phishing mail}) + \text{ham}(\text{non phishing})}$$

$$F_n = \frac{\text{corpus}(\text{non phishing})}{\text{corpus}(\text{non phishing}) + \text{corpus}(\text{phishing})}$$

Where

Ham (phishing mail) -- is good (ham) mails that has been detected phishing mails

Ham (non phishing) – is number of mails identified correctly

Corpus (non phishing) – phishing mails that are misclassified

Corpus (phishing) – phishing mails identified correctly

Chapter 5

Simulation

5.1 Overview of Simulation

To simulate the design proposed by us IDS can be used which can be implemented as a organization wide server. Open source IDS such as Snort are obvious choice because Snort[10] is an open source network intrusion prevention and detection system (IDS/IPS) developed by Sourcefire. Combining the benefits of signature, protocol and anomaly based inspection Snort is the most widely deployed IDS/IPS technology worldwide. With millions of downloads and over 225,000 registered users Snort has become the *de facto* standard for IPS.

Snort provides rule language which can be used to develop or modify rule base for traffic filtering. But Snort rule language has some limitations; first it can only understand TCP and UDP secondly rule language provided by it is not flexible enough to write complex rules for text based filtering. Although Snort provides facility to add plug-in so that the extra bit more flexibility can be attained but in our case since our main focus is on accuracy of the rule base; we have preferred C++ over Snort for the intended work.

Most of the simulation has been done using C++ language and a small portion of the simulation has been done using Shell script. We have done simulation for each rule independently so that the individual performance can be analyzed properly.

Our first step towards the simulation was to develop a generalized character and integer matcher.

The matcher is the base of the whole simulation as the whole simulation logic works using these matchers.

The proposed rule base has been simulated on windows XP environment and all the code has been developed using Dev C++ IDE 4.9.9.2. Each rule implementation is independent of other so each rule can be run independently and performance can be analyzed individually; but each rules simulation uses some common class that needs to be included accordingly.

We have chosen C++ for simulation; basic reason behind choosing C++ is its simplicity and familiarity. As all the work is text based matching scheme C++ became an obvious choice.

Main challenge of the simulation was different format of the dataset which we have collected from various sources. Beside their format one more challenge was that most of the phishing emails use HTML and we needed to cover all possibility by which they can be written so that they can be filtered out accordingly.

5.2 Routine for IP and Character matcher

This is the most important routine of our simulation as each rule will use it somehow. The main logic behind IP matcher routine is that we try to find out a match for a character string “xxx.xxx.xxx.xxx” where each ‘x’ represents a numeric character. While matching it against the array of characters we make sure that ‘x’ occurs at least once before and after each ‘.’ character. Character matcher will be a simple one which tries to find out one to one match; this matcher is not case sensitive since in our case we mostly search through HTML script which basically is not case sensitive.

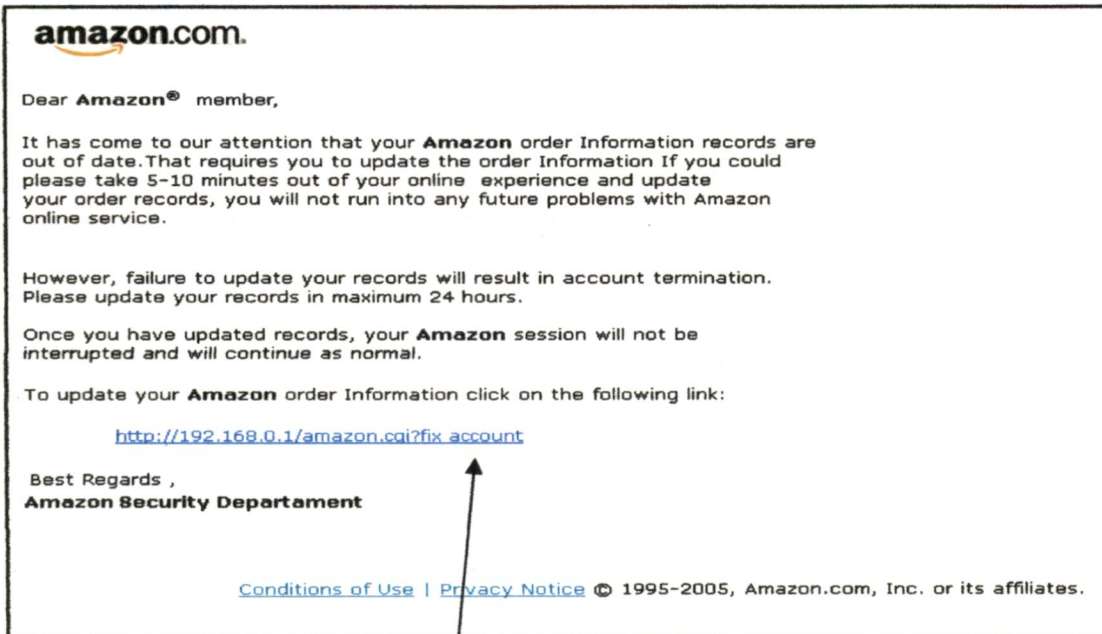
The routine for the IP matcher has been attached in appendix.

5.3 Implementation Details of Different Rules

5.3.1 Implementation of Rule 1- IP Based URLs

Basic component of it is an IP matcher routine which reads in a file line by line and checks whether it contains any IP address or not it goes till the end of the file and collects all those IPs.

Here is an example where a phishing mail has used IP address as a link:



IP Based Link

Figure 5.2 An Example of IP Based Link

5.3.2 Implementation of Rule 2 Newly registered domain names

Today IP based emails are less, Phishers register domain for phishing purpose so that email appears to have generated from authentic place. Generally Phishers register similar sounding names as that of the original domain for example playpal.com or paypal-update.com sounds like paypal.com. But they have fear to be caught so they tend to use the domain as soon as possible. We therefore perform a whois query on each suspected links. WHOIS is a query/response protocol which is widely used for querying an official database in order to determine the owner of a domain name, an IP address, or an autonomous system number on the Internet. WHOIS lookups were traditionally made using a command line interface, but a number of simplified web-based tools now exist for looking up domain ownership details from different databases. Web-based WHOIS clients still rely on the WHOIS protocol to connect to a WHOIS server and do lookups,

clients still rely on the WHOIS protocol to connect to a WHOIS server and do lookups, and command-line WHOIS clients are still quite widely used by system administrators. WHOIS normally runs on TCP port 43.

Simulation for this rule has been done in two parts

- i) Collection of suspected domain name
- ii) WHOIS query

i) Collection of domain name

It has been done on the basis of some other rules, we collect all those domain name if that has appeared in href field of the HTML script of a mail having a property that it does not match that target field.

For example<ahref=http://mail.pooding.com/src/resolution_center/index.php?cmd=LogIn
> <fontface = " Verdana " size = " 2" > http://www.paypal.com/cgi-bin/webscr?cmd=_login run

We also collect all those host name which have unnecessary numbers of dots in them.

For example: <http://www.all-design.com.tw>

ii) WHOIS Query

After collecting domain names we do a whois query to each of them which gives us who is the owner of the domain and on what date it has been registered. To do this task we have used a short simple Linux shell script, since windows by default does not provide whois command. Once we we get the response of the query we then compare the date of the email to the date of registration if the difference happens to be less than 60 days we say domain is fresh domain and mark tha email as a phishing email.

5.3.3 Implementation of rule 3- Non matching URLs

To make the email look authentic Phishers generally use masked link by using HTML script. This way they may be able to convince the user that email is coming from authentic place. To do so they use href an attribute for the anchor element <a>. HREF indicates the URL being linked to. HREF makes the anchor into a link.

This is a real life email example which uses this feature



Figure 5.3 An Example e-mail with Masked Link

As shown in Figure 5.3 above href link does not match the link you are being forwarded to.

To collect all these email we have used character matching routine which tries to find out href tag in an email once it is found. We then turn our head to compare this link to the other attribute of the href if it happens not to be same we say it is a phishing mail else otherwise. Once we get such link we include it in the list of suspected host list we also save these to be used in rule no. 2.

5.3.4 Implementation of Rule 4 and Rule no 5- HTML and Javascript emails

Both these rules are quite same as we just for some keywords to check whether they use HTML or javascript.

To check whether a mail uses HTML or not search for the tag HTML, if a mail contains HTML tag we say it is HTML email. A noteworthy point here is that being an HTML email does not necessarily indicates that is an Phishing email, but at the same time it is very hard for Phishers to deceive users without using HTML or any other sophisticated script. Hence once we find a HTML email we consider it as a suspicious email and try to impose other rules on it to confirm its status.

In a similar way we look for javascript emails, for this we search for the string “javascript” if we find such emails which have javascript tag we confirm it as a phishing email unlike in case of HTML email where it was just a suspicious email.

5.3.5 Implementation of rule 6- Number of dots in a url

As in our very first rule wherein if we get any email containing ip based link we straight away confirm its identity as phishing mail. But today Phishers do not give away any chance by using such a link anymore so they register legitimate sounding domain name to trick the users. Consider the following link: *http://www.playpal.update.data.com*
This link may be mistaken for an email from “paypal.com”. One of the attribute of this mail is that it has more of dots than usual.

To search all of these domain names we collect all the suspected domain names from our dataset. If these domains have more number of dot than usual then we call it as a suspected email. Another way of deception can be launched by using redirection script
Consider the following link: *http://www.google.com/url?q=http://www.somebadsite.com*
above link appears as if it is being directed to some site hosted at google.com but in reality it will take you to some phishing site. To identify this type of deception we search for link that contain redirection script. Both these feature does not confirm the phishing mail identity but they come under suspicious emails. These mails need to be check further for the confirmation as a phishing email.

Chapter 6

Results and Analysis

6.1 Performance of Individual Rules

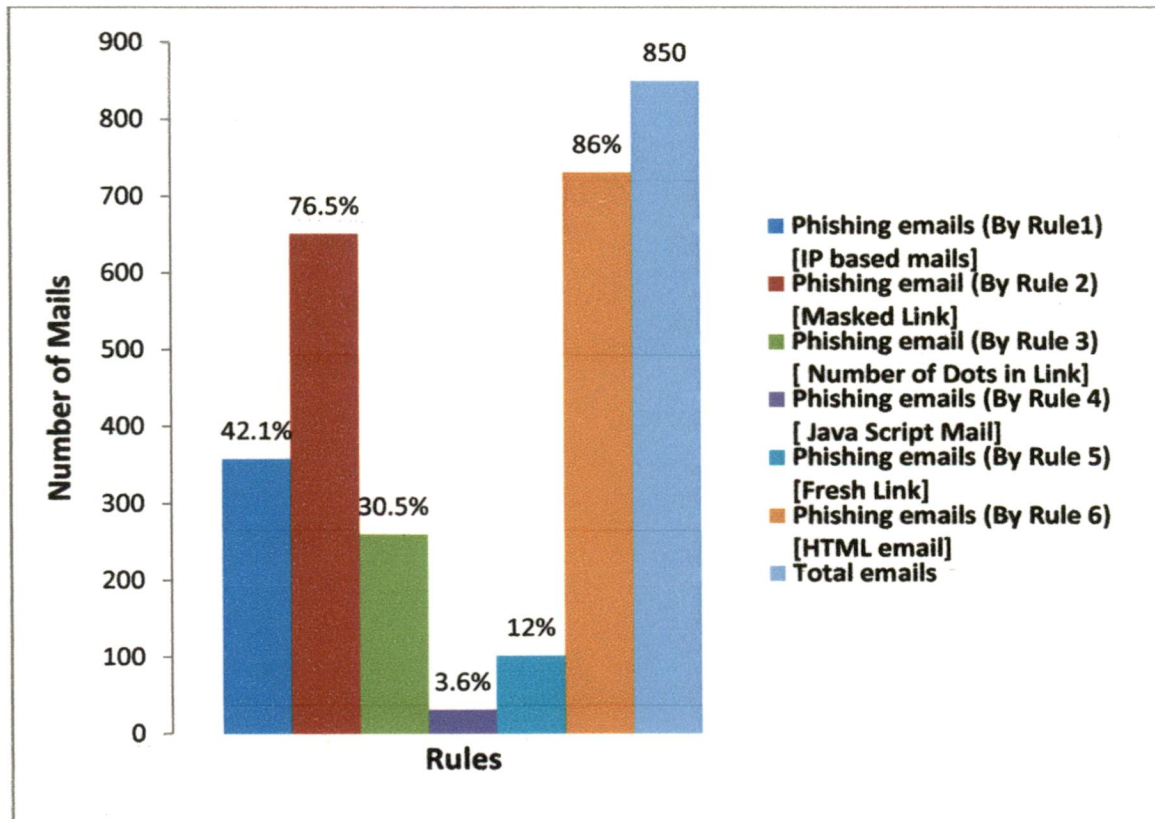


Figure 6.1 Corpus E-mails Containing Different Features

Figure 6.1 is performance of each individual rules on phishing emails dataset (corpus). Each bar corresponds to some rule and shows how many phishing mails (corpus) has that feature.

Analysis of results:

From the above graph we conclude that each of the rules selected by us for email filter has significance as none of them has gone with zero performance.

We have got majority of emails which use html (86%), thus to launch phishing attacks html is basic necessity for Phishers.

Most number of genuine attacks is launched by link masking (76%).

6.2 Overall Performance of the Rule Set

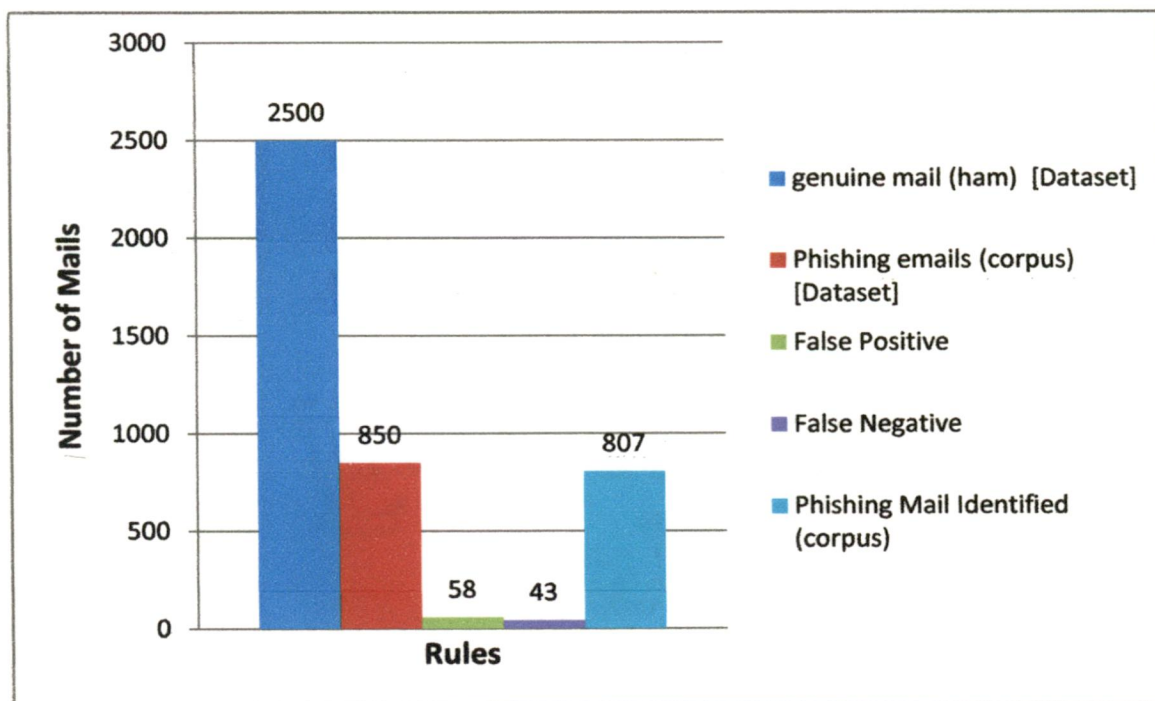


Figure 6.2 Overall Performances of Rules on Phishing and Non Phishing Mails

As shown in Figure 6.2 first two columns are dataset ham and corpus respectively. Ham is genuine mail whereas corpus is a collection of phishing mails. Our proposed rule base identifies 807 phishing mails correctly out of 850 phishing mails which has accuracy nearly 95%.

False positive of our rule base is 0.023, we have misclassified 58 good mails as phishing mails out of 2500.

False negative our rule base is 0.05, we have misclassified 43 phishing mails as good mails out of 850 mails.

Analysis of the result

High percentage of accuracy shows the reliability of the rule set, lesser false positive shows very less genuine emails uses these feature. Hence even if we want to improve on our rule set these rules are indispensable.

False negative is also low in our case which shows compactness of the rule set.

6.3 False positive vs Number of Rules



Figure 6.3 Effect of No. of Rules on False Positive

Figure 6.3 shows a graph between false positive and number of rules used; as shown initially when we have just one rule (rule no1) we have zero false positive which shows that none of the genuine mail contain IP based link. As the number of rules increases false positive also increases. Initially number of misclassified mail is 0 and after including all the rules it has final value 58.

Analysis of results

As we can false positive get increased with the number of rules, thus increasing number of rules does not have positive effect only it has negative effect too. So rule set should always be chosen optimally.

6.4 False Negative vs Number of Rules



Figure 6.4 Effect of Number of Rules on False Negative

False negative is the number of phishing mail that has been misclassified as a non phishing mail.

As shown in Figure 6.4 initially it has maximum value of $F_n=0.57$ and by the inclusion of the sixth rule it decreases to $F_n=0.05$.

Analysis of the results

As the number of rules increases false negative decreases; it is possible to attain zero false negative by increasing number of rules but that will come at the cost of increase in false positive (as discussed in previous section). Hence proper balance should be maintained between false positive and false negative by choosing appropriate number of rules.

6.5 Comparison with IDS Based Anti Phishing System

Table 6.1

<i>Method</i>	<i>No. of Rules</i>	<i>False +ve</i>	<i>False -ve</i>	<i>Accuracy</i>
<i>Proposed Rule Set</i>	Six	0.023	0.05	94.94%
<i>IDS Based Method</i>	Three	0.013	0.08	91%

As shown in Table 6.1 we have compared our rule base with the rule base of IDS based system discussed in section 3.1.6 on the basis of three parameter false positive, false negative and accuracy. We have got better results than the IDS based solution in terms of accuracy. IDS based system has better false negative rate.

Analysis of the results

From above we conclude as number of desirable rules increase accuracy gets increased; this is also the reason why false negative rate is better for our rule set.

Table 6.1 shows false positive (0.013) for IDS based system is better than our rule sets false positive (0.023) reason behind it is quite simple as in figure 6.3 we have shown as the number of rules increases false positive also increased since IDS based system has lesser number of rules its false positive rate is also low.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

In this dissertation we have proposed a rule set for phishing email filter for an organization wide phishing detection system. We have used a new rule base which uses more number of rules than its previous version; we have tested the rule set with the publicly available Phishing and non Phishing mails. We have compared the accuracy of our solution with the existing solution. We have also shown a relation between number of rules in the rule set and false positive and false negative. Simulation of each of the rule has been done individually to study the accuracy and significance of a rule individually.

Following conclusion can be made from our rule set's performance

1. We have compared our rule base with the rule base of IDS based system on the basis of three parameter false positive, false negative and accuracy. We have got better results than the IDS based solution in terms of accuracy. Our system has a accuracy of 94.94% as compared to IDS based system's 91
2. Features to filter out mails should be selected cautiously since false positive and false negative which are performance measurement quantity of rule set, has a direct relation to these features. Loose feature can result larger rate of false positive and false negative.
3. Number of rules to be used for the email filter is also as important as the features, though it appears that increasing number of rules reduces false negative rate but the fact that number of rules also increases false positive should be kept in mind. Hence a proper balanced rule set should be used.

7.2 Future Work

Though we have improved the rule base accuracy for an organization wide system but still there is a large scope of improvement. We have only employed the feature of an email message to figure out whether a mail is phishing mail or not. We have not used any external information to detect Phishing emails; we are mentioning some of the features which can be included to improve its performance

- Organizations such as APWG maintain a blacklist of confirmed phishing websites, which could be integrated with the rule base.
- Many mail clients already have a spam filter in place, these filters assign emails a class as Phishing mail or non Phishing mail we can leverage the advantage of this existing class to classify email.

REFERENCES

- [1] Anti-Phishing Working Group, <http://www.antiphishing.org/index.html>
- [2] Hasika Pamunuwa, Duminda Wijesekera and Csilla Farkas, “An Intrusion Detection System for Detecting Phishing Attacks”, W.Jonker and M.Petkovic (Eds.): SDM, LNCS 4721, pp. 182-192, 2007
- [3] Ian Fette, Norman Sadeh and Anthony Tomasic, “Learning to detect phishing emails” In Proceedings of the International world Wide Web Conference (WWW), pp. 649–656, 2007
- [4] Neil Chou, Robert Ledesma, Yuka Teraguchi, Dan Boneh, and John C. Mitchell, “Client-side defense against web-based identity theft” , In Proceeding of 11th Annual Network and Distributed System Security Symp. (NDSS '04) February 5–6 San Diego, CA Internet Society pp. 119–128, 2004
- [5] Nicolas Vanderavero, Xavier Brouckaert, Olivier Bonaventure and Baudouin Le Charlier, “ The HoneyTank.: A Scalable Approach to collect malicious Internet Traffic In Proceeding of international infrastruc survivability workshop (IISW'04) 2004, held in conjunction with the 25th IEEE International Real-time systems symposium (RTSS04), IEEE Computer Society Press, Los Alamitos pp. 185 – 205, 2004
- [6] R. Gellens, J. Klensin, “Message Submission for Mail”, RFC 4409, The Internet Society, April 2006
- [7] Debra L. Cook, Vijay K. Gurbani, and Michael Daniluk, “Phishwish: A Stateless Phishing Filter using Minimal Rules ”, In Proceeding of 12th International Conference of Financial Cryptography and Data Security, FC 2008, Cozumel, Mexico, pp. 182 - 186, 2008

- [8] The Apache SpamAssassin Project. available at: <http://spamassassin.apache.org/>, site last visited on 10 june 2009
- [9] John Nazario, phishingcorpus homepage, 2006
- [10] <http://monkey.org/%7Ejose/wiki/doku.php?id=PhishingCorpus>, site last visited on 09 Jun 2009
- [11] Snort: The open source network intrusion detection system, <http://www.snort.org/>, site last visited on 10 june 2009
- [12] Niels Provos, "A Virtual Honeytrap Framework", In the Proceedings of the 13th USENIX Security Symposium, pp. 1-1 San Diego, CA 2003.
- [13] The HoneyNet Project & Research Alliance, "Know your Enemy: Phishing-- Behind the Scenes of Phishing Attacks", <http://www.honeynet.org/papers/phishing/2005>.
- [14] Sujata Doshi, Neils provos, Monica chew and Aviel D. Rubin, "A framework for detection and measurement of phishing attacks" Proceedings of the ACM workshop on Recurring malware Alexandria, Virginia, Pages: 1 - 8 USA 2007.
- [15] Aaron Emigh, "Online Identity Theft: Phishing Technology, Chokepoints and Countermeasures", ITTC Report on Online Identity Theft Technology and Countermeasures 2005
- [16] Yue Zhang, Jason Hong, Lorrie Cranor, "CANTINA: A Content Based Approach to Detecting Phishing Sites" In Proceedings of the 16th international conference on World Wide Web, pp. 639 - 648 2007

- [17] Anti-Phishing Working Group ,“The Crimeware Landscape: Malware, Phishing, Identity Theft and Beyond”,
www.antiphishing.org/reports/APWG_CrimewareReport.pdf ,2006
- [18] Ram Basnet, Srinivas Mukkamala, and Andrew H. Sung, “Detection of Phishing Attacks: A Machine Learning Approach”, *STUDFUZZ 226*, pp. 373–383, 2008
- [19] M. Dolores del Castillo, Angel Iglesias, and J. Ignacio Serrano, “Detecting Phishing E-mails by Heterogeneous Classification ”, *International Conference on Intelligent Data Engineering and Automated Learning(IDEAL 2007)*; pp. 234 - 250; Birmingham(GB) 2007
- [20] Anil Sagar, “Phishing Attacks and Counter Measures”, *CERT-In White Paper (CIPW) 2005*, <http://www.scribd.com/doc/.../Phishing-Attack-Countermeasures/>, site last visited on 09 Jun 2009
- [21] Chik How Tan and Joseph Chee Ming Teo, “Protection Against Web-based Password Phishing”, In the proceedings of *International Conference on Information Technology (ITNG'07)*, pp. 754-759 Las Vegas, Nevada, USA
- [22] Anti-Phishing Working Group: Report Phishing,
<http://www.antiphishing.org/smim-dig-sig.html>, site last visited on 09 Jun 2009


```

        if(ch=='\n')
            lineno++;
        if(ch!=oword[j])
            {match=0;break;}
    }
    //len=0;
}
if(match==1)
    { //telgnow=i.tellg();
    count++;
    }

}

}
os.seekp(0);
lineno=0;
}
cout<<"\ncount " <<count<<endl;

```

```

//-----second routine-----
-----//

```

```

int match(char *oword,char* str)
    { long int x=1;

    char line[100];
    int lineno=0;
    int count=0;
    char fname[10];
    char ch;//oword[10];
    int telgprev=0,telgnow=0;

    {
        ifstream i;

        os << m << ".txt";
        const char* fptr=( os.str().c_str() );

        strcpy(fname, str);

        i.open(fname, ios::binary);
    }
}

```



```

        i.seekg(0);

int match, len;
while(1)
{
    i.get(ch);

    if(ch=='\n')
        lineno++;
    if(i.eof())
        break;
    if(ch!=oword[0])
        {continue;}

    else
    {
        match=1;
        len=1;
        for(int j=1; j<strlen(oword); j++)
        {
            len++;
            i.get(ch);
            if(ch=='\n')
                lineno++;
            if(ch!=oword[j])
                {match=0; break;}
        }
        //len=0;
    }
    if(match==1)
    {
        count++;
    }

}

lineno=0;
}
cout<<"\ncount"<<count<<endl;

}

```

```
//-----THIRD ROUTINE-----//
```

```
int match(int getptr, char *oword, char* str)
    { long int x=1;

    char line[100];
    int lineno=0;
    int count=0;
    char fname[10];
    char ch;//oword[10];
    int telgprev=0, telgnow=0;

    {
        ifstream i;

        os << m << ".txt";
        const char* fptr=( os.str().c_str() );

        strcpy(fname, str);

        i.open(fname, ios::binary);

        i.seekg(getptr, ios::beg);

    int match, len;
    int getvalue;
    while(1)
    {
        i.get(ch);
        if(lineno > 3)
            return -2;
        if(ch=='\n')
            lineno++;
        if(i.eof())
            break;
        if(ch!=oword[0])
            {
                continue;
            }

        else
        {
            match=1;
            len=1;
            for(int j=1; j<strlen(oword); j++)
                {
                    len++;
                }
        }
    }
}
```

```

        i.get(ch);
        if(ch=='\n')
            lineno++;
        if(ch!=oword[j])
            {match=0;break;}
    }
    //len=0;
}
if(match==1)
{
    count++;
    int getvalue=i.tellg();
    return getvalue;
}
}
}
}

```

//-----ROUTINE FOR IP MATCHING-----//

```

int ipmatcher(char *ip)
{
    long int x=1;
    int count=0;

    char line[100];
    int lineno=0;
    char ch;

    //int m=10;
    char fname[10];

    char* ptr="xxx.xxx.xxx.xxx";
    int telgprev=0,telgnow=0;

    int len;

    ifstream i;
    ofstream of;
    ostringstream os;

    lineno=0;
    of.open("ipfile.txt",ios::app); //write
down buffer in a file and compare it for possible ip address

```

```

cout<<ip;
of<<ip;of<<"\n";
of.close();
i.open("ipfile.txt",ios::binary);

i.seekg(0,ios::end);
i.clear();
int telgback=i.tellg();
telgback=telgback-(strlen(ip))-2;

i.seekg(telgback,ios::beg);

while(i)
{
con:      int dot=0;
          int d=0;
          int flag=0;
          int match=0;
          i.get(ch);
          // cout<<ch;
          if(ch=='\n')
            lineno++;
          if(!(ch>=48&&ch<=57))
            { //matchon=1;
              continue;
            }
          else
            { // cout<<ch<<"\n";

              char* p=ptr;
              p++;
              for(int j=1;j<strlen(ptr);j++)
                { if(!flag)
                    {
                      i.get(ch);
                    }
                  if(ch=='\n')
                    lineno++;

                  if(*p=='x')
                    {
                      if(ch=='.')
                        { flag=1;
                          }
                    }
                }
          if(d==1&&!(ch>=48&&ch<=57))//immediately
            //after '.' there must be numeric
            //following it

```

```

    {
        match=0; break; cout<<ch<<" 1\n";
    }
    match=1;
    d=0;
    p++;
}
else if(ch=='.' && (*p)=='.')
{
    flag=0;
    d=1;
    dot++;
    p++;
}
else
{
    flag=0;
    if(dot>=3)
    {
        cout<<" 1Found a match\n";
        // cout<<"DOT " <<dot<<"\n";
        match=0; d=0; return 1;
    }
    break;
    cout<<dot<<" " <<ch<<" " <<lineno<<" 2\n";
}

```

```

if(ch!='.' && !(ch>=48 && ch<=57) && ch!='\0' && ch!='\n' && ch!=13 && ch!=9
&& dot<3)

```

```

    {
        cout<<"i visit here1 " <<int(ch) <<" \n";
        match=0;
        goto con;
    }

```

```

if((ch>=65 && ch<=90) || (ch>=97 && ch<=122))

```

```

    {
        cout<<"i visit here2 " <<ch<<" \n";
        match=0;
        goto con;
    }
} //end of for loop

```

```

} //end of else
if(match==1)
{ if(dot<3)
    return 0;
}

```

```
        //cout<<" dot "<<dot;
        cout<<"2Found a match \n";
        count++;
        return 1;
    }

} //end of while
os.seekp(0);
lineno=0;

return 0;
} //end of function
```

```
};
```