www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 4 Issue 3 March 2015, Page No. 10914-10917

Security Suite

Tanmay Ghawate[1], Chaitanya Patel[2], Rushikesh Bargaje[3], Kamlakar Kadam[4], Prof. Harmeet Khanuja[5].

[1] Pune University, Department of Computer Engineering
MMCOE, Pune-411052, India.
tanmayghawate.comp@mmcoe.edu.in

[2] Pune University, Department of Computer Engineering
MMCOE, Pune-411052, India.
chaitanyapatel.comp@mmcoe.edu.in

[3] Pune University, Department of Computer Engineering
MMCOE, Pune-411052, India.
rushikeshbargaje.comp@mmcoe.edu.in

[4] Pune University, Department of Computer Engineering
MMCOE, Pune-411052, India.
kamlakarkadam.comp@mmcoe.edu.in

[5] Pune University, Department of Computer Engineering
MMCOE, Pune-411052, India.
harmeetkaurkhanuja@mmcoe.edu.in

Abstract: Now a days information security is become a basic need in this digital world. It is very important to protect the information from various intruders. The system on which information is stored should not be vulnerable to E-mail attacks such as phishing, E-mail date and time spoofing E-mail address spoofing, as well as various malware attacks. This project presents basic protection that can be provided to the system .It illustrates E-mail forensics to detect E-mail Date and Time spoofing as well as E-mail address spoofing. We have created dataset of spoofed and legitimate E-mails. We propose a technique to perform the analysis of E-mail, by reading the header information and analyzing the fields related to date and time as well as sender and receiver address. We have also given an introduction to various malware detection techniques. We have described hash based malware detection technique. Here we have compare the hash values of input file with the database of bad hash values to detect virus.

Keywords: SMTP; E-mail Spoofing; E-mail Forensics; E-mail headers ,Hash ,MD5,SHA ;

1. INTRODUCTION

Security suit is an application that is used to provide basic security features to the user. It consists of two basic modules one is for spoof email detection and other is for malware detection. Email spoof detection module provide basic technique for detection of Email address spoofing and Email date and time spoofing by analysing the header file of the Email . In Email address spoofing we check the senders host and sender's address. In Email-date and time spoofing we compare send-date and receive-date.

Second module provides malware detection. Techniques used in malware detection can be signature based, behaviour based or heuristic. Hash based detection technique is simple technique to detect malwares. We can use various hashing algorithms such as MD5, SHA etc. Database of bad hash values to be maintained and should be scanned efficiently. We have to use efficient string searching algorithms to scan the database. We can directly use some antivirus engines that can be integrated with your security software to provide extra layer of security. Security suite can be enhanced by adding some more functionalities such as anti-phishing, malicious web site scanner.

2. E-MAIL DATE AND TIME SPOOFING

.

E-mail date spoofing emerged as an e-mail spoofing trick, wherein a Spammer sends spam e-mails that contain Forged send date to recipients. It keeps e-mails listed on top in recipient mailbox, thereby maximizing the chances of immediate attention by the recipient. The Date header field in a date spoofed e-mail may contain a date which is ahead or before the actual date it was sent. Accordingly, a date spoofed e-mail may be either a predated or a post-dated message. Almost all e-mail servers accept e-mail messages

spoofed in date. On webmail systems which use send date for sorting, post-dated e-mail messages remain on top in the inbox of the recipients. Send date sorting and short date formats in some webmail systems make even learnt recipients difficult to suspect date spoof mail, and date-spoofed e-mails are not identified by spam filters. Simple mail transfer protocol (SMTP) [1] which is the most widely deployed and primary protocol for E-mail Transfer does not define any security and privacy policy Date spoofing has been reported by Banday et al in [2], who have conducted a detailed study of handling of such e-mail messages by some commercial and corporate e-mail servers.

2.1. Technique for Detection of Date Spoofing:

In our proposed technique we calculate the margin which is the average time required to receive mail from the sender. By comparing this margin with the difference between send date and receive date we can recognise the date spoofed mail.

2.2. Technique for calculating margin

We consider some number of legitimate E-mail datasets and find out the delay in the delivery of each E-mail. The maximum delay is calculated and taken as the margin. This value should be updated as frequently as possible by considering more legitimate datasets for accurate threshold.

Algorithm for margin calculation [3]:

```
set margin=0, diff=0
set the margin value from the previous run
(margin is set to 0 in first run)
while (mail headers are available)
{
        Read date / time / offset of first Received: and
        Date:
        Convert into UTC
        Calculate difference between Lastser_Time and
        Sending_Time
        diff =  Lastser_time - Sending_time
        if (diff>margin)
         margin = diff
}

Write margin to the margin file
```

Algorithm for identifying date spoofed mail [3]:

```
if(sending_date and sending_time is not according to the
usual semantics)
then date error notification
elseif (sending_date = = lastser_date)
{
        if (lastser_time – sending_time < margin)
        the mail as legitimate
        else
        the mail as time spoofed
}
elseif (sending_date = = firstser_date)
        then mail as legitimate
else
        then mail as date spoofed
```

This algorithm requires conversion of time from time zones, which obeys the following relationship: Time in Time Zone A – UTC Offset from Zone A = Time in Zone B – UTC Offset for Zone B.
Thus, we can rearrange the above relationship to get Time in Zone B. Time in Zone B = Time in Zone A – UTC Offset for Zone A + UTC Offset for Zone B.

3. E-MAIL ADDRESS SPOOFING:

E-mail address spoofing is sending mail which Camouflages itself to come from someone else. It is done by modifying the E-Mail header [4] Form: field.

A Sample E-mail header contains the following fields: From, X-Apparently-To, Return–Path, Received–SPF, X-Originating-IP, Authentication-Results, Received-from, Received-by, DKIM-Signature, Date, From, To, Message-ID, Subject, MIME-Version, Content-Type, Content–Transfer-Encoding, Content-Length etc. The E-Mail Sender Spoofing can be detected by comparing the From, Return-Path, and the X-Received by headers. Any dissimilarity in the any of the above fields indicates modified E-Mail headers and confirms E-Mail Sender Spoofing.

3.1. Proposed Technique to detect E-Mail Sender Spoofing:

```
If (From = Return-Path)
{
        If (From = X – Received-By)
        {
                // legitimate Mail
        }
        Else
        {
                //Spoofed Mail
        }
}
Else
{
        // Spoofed Mail
}
```

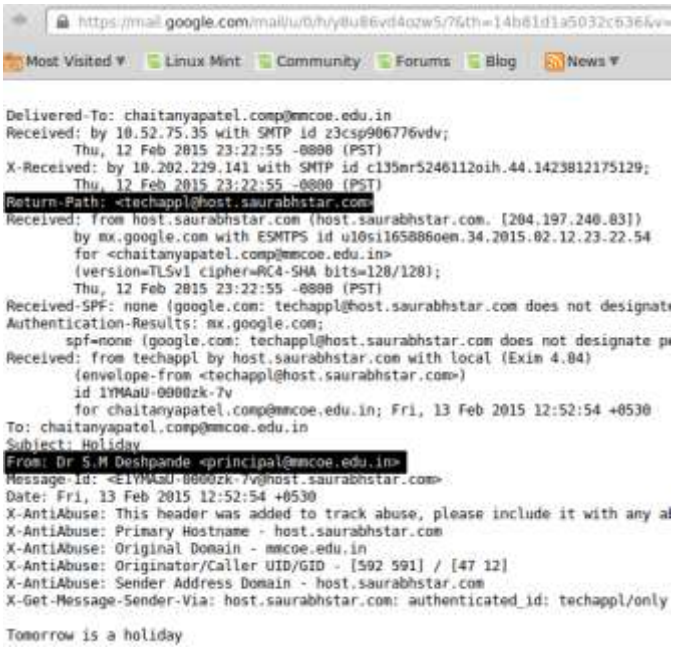

Figure 1: Email Address spoofing

As shown in Fig.1 the host name of the email address specified in the Return-Path field is "host.saurabhstar.com", whereas the host name of the email address in the From field is "mmcoe.edu.in". This clearly indicates that these two host names are different which means that one who has send mail is another mailer who is pretending to be the authenticated one. Hence from the headers of the mail we can conclude that the email is address spoofed.

4. MALWARE DETECTOIN TECHNIQUE:

Malware is malicious software that leads to malfunctioning of the infected system. There are various types of malwares such as viruses, worms, Trojan horse, adware, spyware, rootkits etc.

There are various malware detection techniques. These techniques are commonly classified as follows [5]:

4.1. Signature-Based Malware Detection Techniques:

Commercial antivirus scanners look for signatures which are typically a sequence of bytes within the malware code to declare that the program scanned is malicious in nature.

But there are some issues related to this technique:

- Signature extraction and distribution is a complex task.
- The signature generation involves manual intervention and requires strict code analysis.
- The signatures can be easily bypassed as and when new signatures are created.
- The size of signature repository keeps on growing at high rate according to time.

4.2. Specification-based Detection

Specification-based detection is the derivate of anomaly based detection. Instead of approximating the implementation of a system or application, specification-based detection approximates the requirements of application or system. In specification-based system .There exists a training phase which attempts to learn the all valid behaviour of a program or system which needs to inspect. The main limitation of specification based system is that it if very difficult to accurately specify the behaviour the system or program. One such tool is Panorama which captures the system wide information flow of the program under inspection over a system, and checks the behaviour against a valid set of rule to detect malicious activity [9].

4.3. Behaviour -based Detection

Behaviour based detection differs from the surface scanning method in that it identifies the action performed malware rather than the binary pattern. The programs with dissimilar syntax's but having same behaviour are collected, thus this single behaviour signature can identify various samples of malware. This types of detection mechanisms helps in detecting the malwares which keeps on generating new mutants since they will always use the system resources and services in the similar manner.

4.4. Proposed technique for malware detection:

As we discuss some malware detection techniques above, the simplest and commonly used technique is signature based detection. Signature is byte code present in virus file which uniquely differentiate it from legitimate files. There are millions of viruses present and growing every day. As much as viruses increase we have to expand our virus signature database .This will be difficult to maintain database as number of virus increases. We can use hash code [10] of files to compare the input files with our database to detect the malware. We have to maintain bad hash value database to detect malware.

We can design simple malware scanner in two basic steps:

Step1: Calculation of hash value:

There are various hashing algorithms used to calculate hash value of input file. They are as follows:

Table 1: hashing algorithms [6]:

Algorithm	M.D size	Message Size	Block Size	Word Size	No. of rounds
MD5	128	< 2^ 64	512	32	64
SHA-1	160	< 2^64	512	32	80
SHA-256	256	< 2^64	512	32	64
SHA-512	512	< 2^128	1024	64	80

From the above hashing algorithms you can use any suitable algorithms to calculate hash value of file. Here

we are going to use MD5 algorithm. MD5 seems to be less CPU intensive.

Step2: Comparing Hash value with database:

For comparing calculated hash values with the huge database of bad hash values we must use faster algorithm for string compare [7]. There are various string searching algorithms that can be used. They are as follows [8]:

1. Naive string matching algorithm:

It is also known as Brute Force method**.** It has no pre-processing phase, needs constant extra space. It always shifts the window by exactly one position to the right. It requires '2n'. Expected text characters comparisons: The most intuitive way is to slide a window of length *m* (pattern) over the text (of length *n*) from left to right one letter at a time.

2. Knuth–Morris–Pratt algorithm:

It compares the pattern with the text from left to right. In case of a mismatch or whole match it uses the notion border of the string. It decreases the time of searching compared to the Brute Force algorithm. This algorithm uses automata to find all the occurrences of a pattern in a text.

3. Boyer-Moore String Matching Algorithm:

It is a particularly efficient string searching algorithm, and it has been the standard benchmark for the practical string searching. This algorithm holds a window containing pattern over text, much as the naïve search does. This window moves from left to right, however, its improved performance is based around two ideas:

1. Inspect the window from right to left.
2. Recognize the possibility of large shifts in the window without missing a match.

4. Rabin Karp String Matching Algorithm:

It uses the naïve search method (i.e. sliding window) and substantially speeds up the testing of equality of the pattern to the substrings in the text by using *hashing***.** It is used for multiple pattern matching (in addition to single pattern matching), because it has the unique advantage of being able to find any one of *k* strings in $O(n)$ time on average, regardless of the magnitude of *k*. The key to performance is the efficient computation of hash values of the successive substrings of the text**.** A hash function converts every string into a numerical value, called its hash value (code, sum), using for instance the ASCII value of characters.

5. Aho-Corasick String Matching Algorithm:

The algorithm consists of two parts: The first part is the building of the tree from keywords or patterns you want to search for, and the second part is searching the text for the keywords using the previously built tree (finite state machine). FSM is a deterministic model of behaviour composed of a finite number of states and transitions between those states. During the second phase, the BFS (breadth first search) algorithm is used for traversing through all the nodes.

At each stage, the node to be expanded is indicated by a marker. In general all the nodes are expanded at a given depths before any nodes at the next level are expanded.

Table 2: Comparison of String Searching Algorithms [8]:

Algorithm	Preprocessing time	Matching time
Naïve string search algorithm (brute force)	0 (no preprocessing)	average $O(n+m)$, worst $O(n\ m)$
Knuth-Morris-Pratt algorithm	$O(m)$	$O(n)$
Boyer-Moore algorithm	$O(m + \|\Sigma\|)$	$O(n/m)$, $O(n)$
Rabin-Karp algorithm	$O(m)$	average $O(n+m)$, worst $O(n\ m)$
Aho-Corasick algorithm (suffix trees)	$O(n)$	$O(m+z)$
		z = number of matches

5. Conclusion:

To provide well-rounded protection, a security suite for personal computers or smart phones should include a collection of tools blending various capabilities that operate in synergic fashion. This article provided an overview of some of the most relevant approaches that can be used to provide basic level of protection. E-mail is an important application that needs to be subjected to a number of security measures. We have seen that how to check the E-mail header contents to identify the date spoofing and address spoofing. We have also seen the various malware detection techniques, their merits and demerits. Here we also described hash based malware detection technique. Here we have compare the hash values of input file with the database of bad hash values to detect virus.

6. References:

[1] J. Klensin, "Simple Mail Transfer Protocol," RFC 2821, April 2001.

[2] Banday MT, et al. 2010. Analysing Internet e-mail date spoofing, Digital Investigation, doi:10.1016/j.diin.2010.11.001.

[3] Preeti Mishra, Emmanuel S. Pilli and R. C. Joshi," Forensic Analysis of E-mail Date and Time Spoofing" 2012 Third International Conference on Computer and Communication Technology.

[4] Email header: "How to read and analyze email header fields and information about SPF, DKIM,SpamAssassn" https://www.arclab.com/en/amlc/how-to-read-and-analyze-the-email-header-fields-spf-dkim.html

[5] Nwokedi Idika, Aditya P. Mathur," A Survey of Malware Detection Techniques", February 2, 2007
[6]William Stallings, "Cryptography and Network security:Principle and Practice", ISBN-13 9788131761663.
[7] http://en.wikipedia.org/wiki/String_matching.
[8] Akhtar Rasool,Amrita Tiwari, Gunjan Singla,Nilay Khare," String Matching Methodologies:A Comparative Analysis", (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 3 (2), 2012.
[9] Heng Yin, Dawn Song, Manuel Egele,Christopher Krugel , and Engin Kirda , "Panorama: Capturing System – wide Information Flow for Malware Detection and Analysis", in Proc CCS'07, October 29 – November 2, 2007, Alexandria, Virginia, USA,ACM Press
[10]https://msdn.microsoft.com/enus/library/f9ax34y5%28v=vs.110%29.aspx