An Email Application with Active Spoof Monitoring and Control

T.P. Fowdur, Member IEEE and L.Veerasoo
p.fowdur@uom.ac.mu lutchmee.veerasoo@umail.uom.ac.mu
Department of Electrical and Electronic Engineering
University of Mauritius
Mauritius

Abstract- **Spoofing is a serious security issue for email applications. Although several anti-email spoofing techniques have been developed, most of them do not provide users with sufficient control and information on spoof attacks. In this paper a web-based client oriented anti-spoofing email application is proposed which actively detects, monitors and controls email spoofing attacks. When the application detects a spoofed message, it triggers an alert message and sends the spoofed message into a spoof filter. Moreover, the user who has received the spoofed message is given the option of notifying the real sender of the spoofing attack. In this way an active spoof control is achieved. The application is hosted using the HTTPS protocol and uses notification messages that are sent in parallel with email messages via a channel that has been secured by the Secure Socket Layer (SSL) protocol. The notification messages allows the application to detect spoofed emails. The application has been developed in PHP using the Netbeans IDE and provides a user-friendly interface which is easily deployable over the web.**

Key words: Anti-Spoofing, Application, Email Security

I. INTRODUCTION

Electronic mail (email) has become an indispensable means of communication for social, business and educational purposes. According to an email statistics report published for the period 2015-2019 by the Radicati Group [1], the number of email users is expected to increase from over 2.6 billion in 2015 to over 2.9 billion in 2019 and more than 205 billion emails were exchanged daily in 2015. These statistics clearly demonstrates the increasing adoption of email as a means of communication [1]. However, emails entail several massive security threats for both individuals and corporations. They are susceptible to threats such as virus attacks, email spoofing, phishing, scam and spamming. The Radicati Group reports that 90% of the 2.8 million emails that are sent every second and some 90 trillion emails that are sent yearly are but spam and viruses [2]. Another major threat is email spoofing which happens when a person (the spoofer) sends an e-mail to a recipient with a forged sender address. The spoofer can thus send malicious emails to specific recipients or extract sensitive information from them by falsely claiming that the email is actually from the trusted source [3,4,5]. There are several conventional measures that can be used to counteract email spoofing for example Sender Policy Framework (SPF) [6], SenderID [7] and DKIM (Domain Keys Identified Mail) [8]. Nonetheless, these measures are not flawless and it is still possible to spoof emails easily using certain programs [9,10]. Consequently, several researchers have developed more sophisticated means to counteract email spoofing. An overview of some recent anti-spoofing mechanisms is now presented

In [11], the authors proposed an anti-spoofing scheme for IP packets which provides an extended inter-domain packet filter architecture along with an algorithm for filter placement. A security key is first placed in the identification field of the IP header and a border router checks the key on the source packet. If this key corresponds to the key of the target packet, the packet is considered valid, else it is flagged as a spoofed packet. A Packet Resonance Strategy (PRS) which detects different types of spoofing attacks that use up the resources of the server or commit data theft at a datacenter was proposed in [12]. PRS used a two-level detection mechanism which allowed a client to access the datacenter only after undergoing a two-level authentication. The two levels are called Packet Bouncer and Packet Transmit respectively and a spoofing attack could be detected at both levels [12]. A new approach to detect E-mail address spoofing was proposed in [13] where fingerprint recognition was used for user authentication and ensuring secure email sign in and message viewing. The user's fingerprint and record were used for user validation, signature for email messages and verification of other users' emails. This mechanism has been employed as an email client referred as the Secure Email with Fingerprint Recognition (SEFR) [13]. In [14], SMTP servers were used to enforce a policy to compare data fields in email headers against a threshold for detecting date and time spoofing. An algorithm to compute the threshold was developed. The algorithm incorporated forensic analysis of the E-mail header to detect date and time spoofing [14]. Finally, in [15], a client-oriented SSL-based anti-spoofing application was proposed. The application allowed secure emails to be sent and received emails to be authenticated using the SSL protocol. The application successfully authenticated valid email messages and detected spoofed ones [15].

This paper extends the anti-spoofing application of [15] by incorporating several new features. First, it combines the HTTPS protocol with SSL so as to make the application become globally deployable. In [15] the application could be used only within a subnet. Moreover, in [15] the notification message as well as the spoofed email was all channeled into the user's inbox. However, in the proposed application, when a spoofed email is received, the application generates an alert message on a dashboard and a spoof filter is used to direct the spoofed email to a separate folder. These measures prevent clogging of the user's inbox. If the email comes from a

978-1-4673-6680-9/16/$31.00 ©2016 IEEE

genuine source, a notification message is also generated in the dashboard display window. Finally, the proposed application provides the user with an option of notifying the true sender that his/her email has been spoofed. The application has been developed in PHP using the Netbeans IDE and provides a user-friendly interface which is easily deployable over the web.

This paper is organized in the following way: Section II gives an overview of email servers, spoofing attacks and security measures to counteract spoofing. Section III describes the implementation details of the new application. Section IV deals with system testing and Section V concludes the paper.

II. BACKGROUND

Figure 1 shows a generic email transmission system [16]. The main components of this system are the Mail User Agent (MUA), Mail Transfer Agent (MTA) and the Mail Delivery Agent (MDA). The MUA allows a user to create new email messages and read incoming email messages. In Figure 1, the user creates a source email using the MUA. The email is then directed by the MDA which is responsible for managing email transmission between the MUA and MTA. The MDA forwards the email to the MTA where it enters a company network. The MTA is responsible for delivering email messages to their destinations.

The message is delivered by the Simple Mail Transfer Protocol (SMTP) which uses port 25 [16, 17, 18]. If the destination address of the email is different from that of the sender, the MTA has to look for the Mail Exchanger (MX) information of the destination mail server. The MX information allows the current MTA to know which mail server it has to establish a connection with to forward the email. This MX information is stored in the Resource Record of a Domain Name System (DNS).

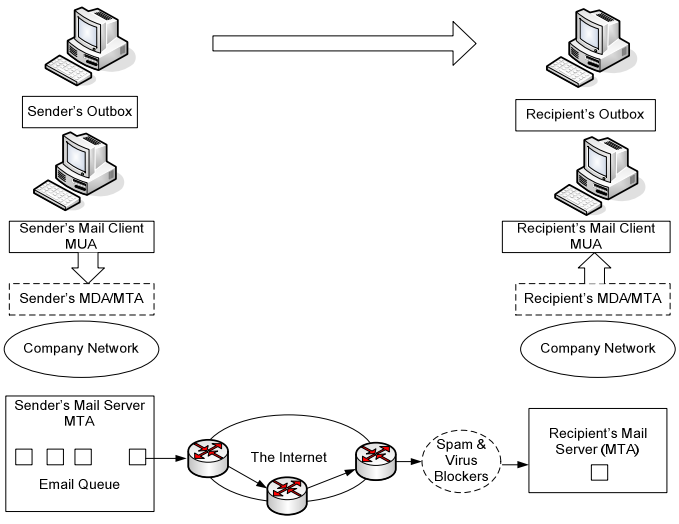

Figure1. Email Transmission System [16].

Once the MX information is retrieved, the MTA forwards the email to the MTA of that mail server. The receiving mail server runs a series of anti-virus and anti-spam checks to ensure that the email is not infected with viruses and spam contents. If the checks are successfully carried out, the email message is then forwarded to the MTA of the destination email server [17, 18]. The MTA then sends the mail to the recipient's MDA to deliver the email to the correct recipient's mailbox. This message can then be read by using a proper MUA. If authentication is successful, the MUA connects to either a POP3 (Post Office Protocol) server or an IMAP (Internet Mail Access Protocol) server to retrieve the email [17, 18].

Most email servers are equipped with ubiquitous security mechanisms such as the Domain Keys Identified Mail (DKIM) and Sender Policy Framework (SPF) to fight threats such as spoofing and spamming [6,7,8]. Figure 2 illustrates a simple spoofing attack scenario. User A and User B are trusted parties. User C wants to send an email to User A by pretending to be User B. Consequently, User C spoofs the email address of User B and sends a spoofed email to User A. In the absence of adequate security and anti-spoofing measures, User A will believe that the email sent to it by User C is from User B.

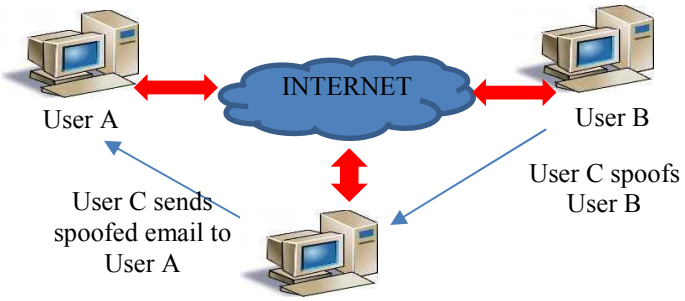

Figure 2. A Simple Spoofing Scenario.

There are several anti-spoofing email mechanisms that have been developed. Since this paper is based on the work of [15], an overview of the application proposed in [15] is given next. The architecture for the application developed in [15] is illustrated in Figure 3. The application has a Graphical User Interface (GUI) and it is deployed on all devices in the same subnet. Two such users using the application are illustrated in Figure 3 The primary functions of the application are sending emails and sending SSL-based confirmation messages to email recipients once an email is sent.

Suppose an email has to be sent from a gmail account. The application first connects to the internet and accesses the SMTP server of gmail, i.e. *smtp.gmail.com.* After that, the email address and the password of the user are authenticated and the user can send emails to any destination via the application from his/her gmail account. [15].

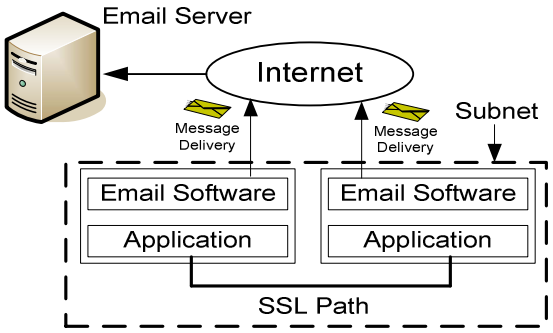

Figure 3. Architecture of anti-spoofing email application of [15].

A database of all Internet Protocol (IP) addresses and user names of all users on the subnet is created by the application and displayed in a table. If a user has to send an email to a different user on the same subnet, s/he can select the user's name from the table. When the name is selected, the IP address of that user is saved by default so as to allow an authenticated message to be sent by creating a secure Transport Control Protocol (TCP) connection with SSL. When the recipient receives the email it also receives the authentication message which confirms that the email comes from the trusted source. If a spoofed email is received, no authentication message is obtained and hence the identity of the source is not confirmed [15].

One major limitation of the application of [15] is that its operation is restricted only to devices in the same subnet. Moreover, both the authentication messages and spoofed emails are sent to the users' inbox. Finally, as is the case with most other anti-spoofing email applications, there are no automated means to inform users whose email addresses have been spoofed of the spoofing attacks. These issues are resolved in the new application which is proposed in this work and described in the next section.

III. PROPOSED EMAIL APPLICATION

This section describes the architectural and implementation details of the proposed email application which provides active spoof control and monitoring. The system is implemented in PHP using the Netbeans IDE. MySQL has been used to design the database part and the apache web server has been used to process requests via HTTPS. A block diagram of the application's architecture is shown in Figure 4. In order to send emails using the application, the users connect to the website where the application is hosted. In this case it is an experimental website used to test the application: https://spoofemaildetector.website.

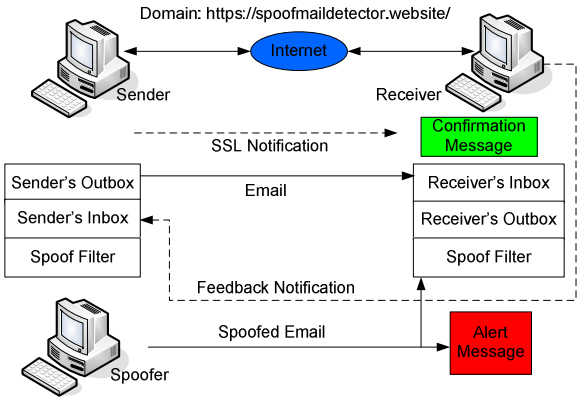

Figure 4. Architecture of the proposed system.

The users must first register themselves in the application before they can access their respective mail servers via the application for sending and receiving emails. When the sender sends an email, an SSL notification is sent in parallel to confirm its authenticity. The receiver receives the sender's email in his /her inbox and the secure SSL based notification on a dashboard as an authentication message. If a spoofer sends a spoofed email to the receiver, the receiver will receive an alert message. The spoofer's mail will be filtered by a spoof filter and the receiver will also be given the option of sending a feedback to sender, informing him that his email has been spoofed. The use of notification messages, alert messages, feedback messages and spoof filtering, altogether provide an effective means of monitoring and controlling spoofing. The application was implemented using several PHP methods. The complete structure of the methods is given in Figure 5.

The Login.php method has been used to implement the system Login form. Users input their usernames and passwords via this form to log into the system. This method also validates the username or password input by the user. The main GUI form of the system is Index.php. It comprises of a dashboard which consists of the inbox, the emails sent, the form for registering users and the send email form. Additionally, it gives a basic description of the system.

When the program is launched on the internet, the user is prompted to log into the system via the login form. Registration is a mandatory step which a user has to complete to be able to log into the system. A registered user is given a username and password which s/he uses to log into the system. This username and password is different from the email address and password of the user's email account. If the username or password is invalid, the user will be prompted by the system to re-enter them. Upon successful validation of the username and password, the system grants access to the user.

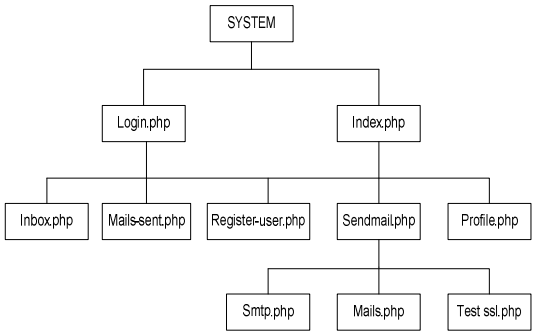

Figure 5. The PHP methods structure of the application.

An email can be sent by first selecting the send email form and filling up all the required fields. The appropriate SMTP server must also be chosen. The data input by the user from the send email form is stored by the application as soon as the "send" button is pressed. The data are also mapped onto corresponding variables to startup an email transmission session with a valid destination SMTP server. The email sent is authenticated, and the receiver receives a secure email notification. If a spoofed email was sent to the receiver, the application detects the spoofed mail and generates a spoof alert. Moreover, the recipient can notify the sender that his/her email address has been spoofed via a feedback notification. A detailed demonstration of this process is given in the next Section.

The Profile.php method implements a drop down menu for users, giving them the option to either view their profile or log out of the system. Moreover, register-user.php is the registration form for users in the system and Sendmail.php is for sending emails. The Sendmail.php form contains three sub-methods namely: smtp.php, mails.php and test-ssl.php. The smtp.php method sends emails via the SMTP port number 587.

IV. TESTING AND ANALYSIS

In this section the operation of the application is illustrated and tested. Two PCs connected by a wireless network as shown in Figure 6 are used for the testing.

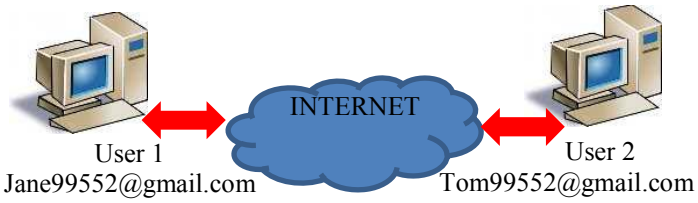

Figure 6. Experimental set-up for application testing.

In order for users to be able to use the application, they must first be registered on the system. Registration is performed by accessing the link https://spoofemaildetector.website/register-user.php . The users will then be directed to a form for filling up their details. They will have to select a user name and password to access the system and then input the details of their email account. A typical form for User 1 is shown in Figure 7.

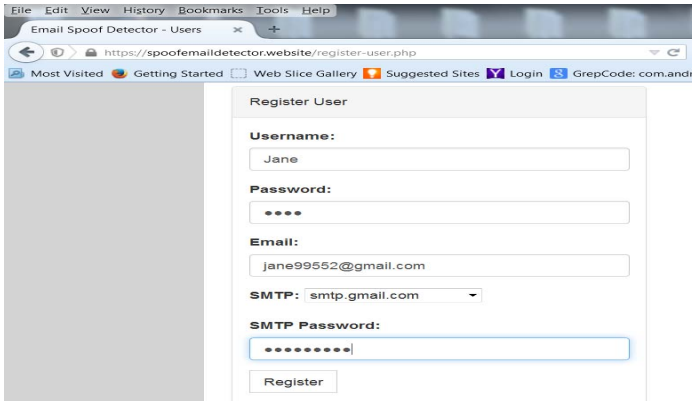

Figure 7. Registration Process for User 1.

Once the users have been registered, they can access the system on the address https://spoofemaildetector.website/login.php, where they will first have to login as illustrated in Figure 8. For the case of User 1, the login step is illustrated in Figure 8. After successful login, to send an email, the email form must be chosen from the dashboard.

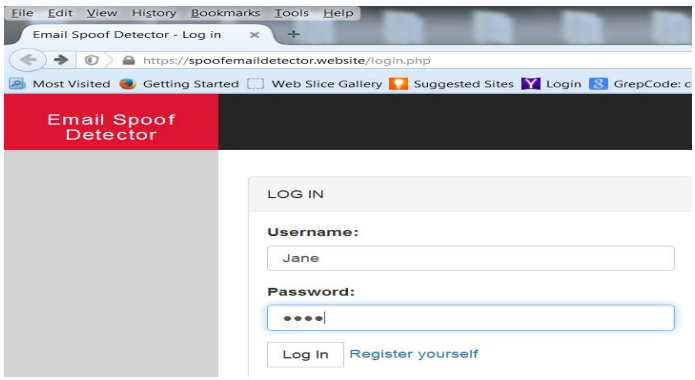

Figure.8. Login process of User 1.

The destination email addresses of the users are obtained directly from the set of registered users in the system as shown in Figure 9.

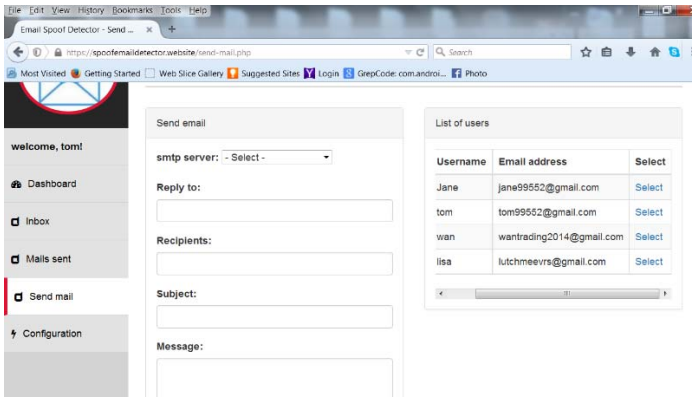

Figure 9. The Send email form.

As an example, suppose the user Tom wants to send an email to user Jane. For that purpose, Tom selects Jane from the list as well as the appropriate SMTP server which in this case is gmail as shown in Figure 10. It is to be noted that emails can be sent only to registered users.

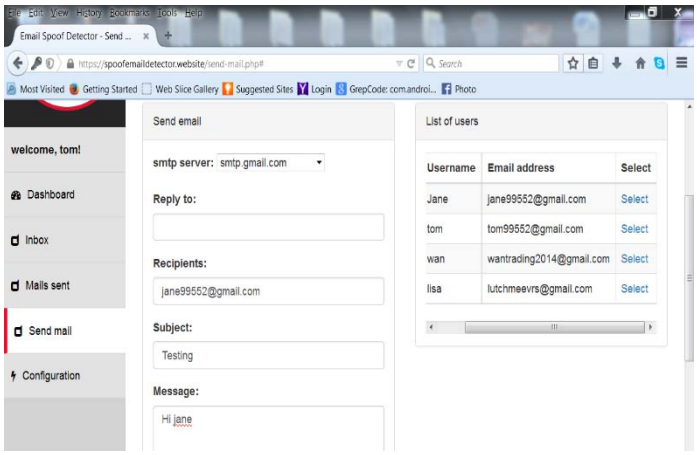

Figure.10. Sending email example.

When Jane receives the email, the system can also confirm that the email has actually been sent from Tom. This is because when Tom sent the email, an SSL authenticated message was also sent in parallel to Jane. The message will appear in the application window when Jane opens the email as shown in Figure 11. If ever someone had spoofed the email address of Jane and had sent a spoofed email to Tom using Jane's email address, the application would generate an alert message. This is illustrated in Figure 12. At the same time the application would allow Tom to send a notification to Jane to inform her that her email has been spoofed. As such if Tom clicks the 'notify the real user' option from Figure 12, Jane would receive a spoof notification message as shown in Figure 13. The application therefore not only allows secure emails to be sent but also provides an efficient monitoring of spoofed emails via alert messages by allowing users to give feedback on the detection of spoofed emails.

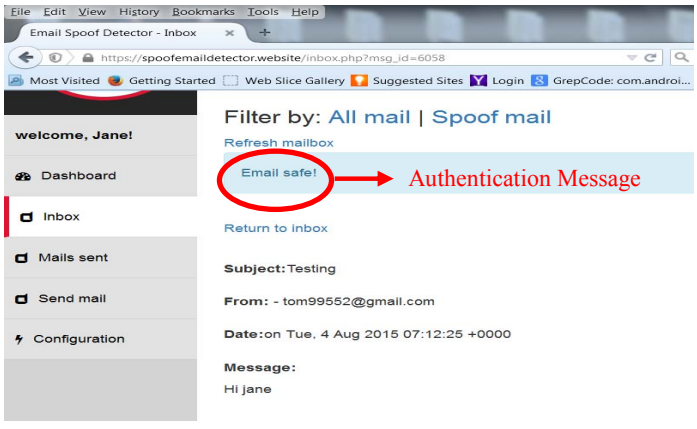

Figure 11. Email authentication message.

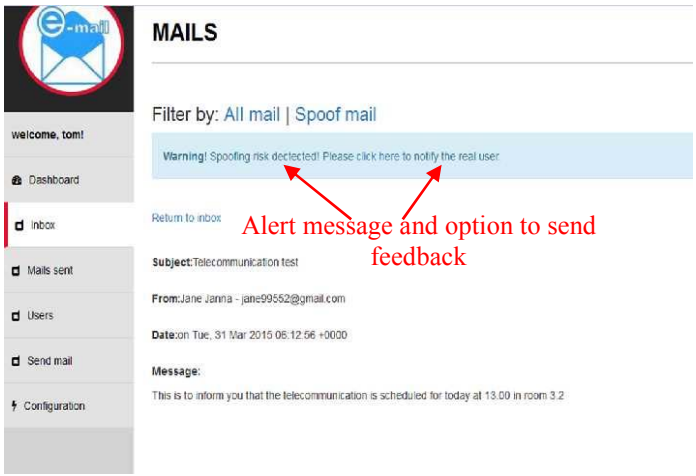

Figure 12. Spoof email detection.

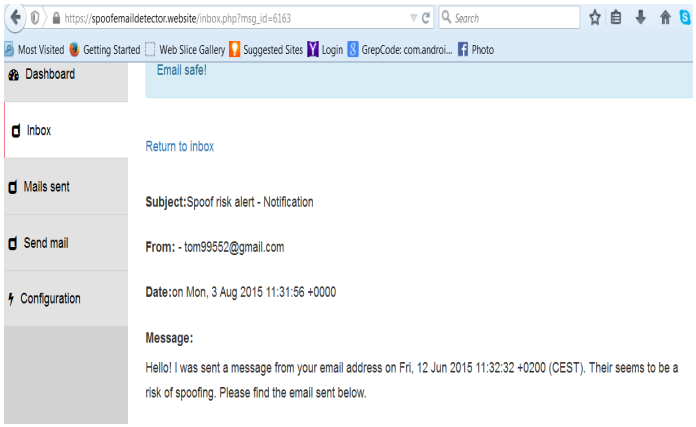

Figure 13. Spoof email notification feedback.

Finally, the application also has an option to filter only the spoofed mails so as to separate them from legitimate emails. Figure 14 shows a list of all the spoofed emails received in the spoof filter.

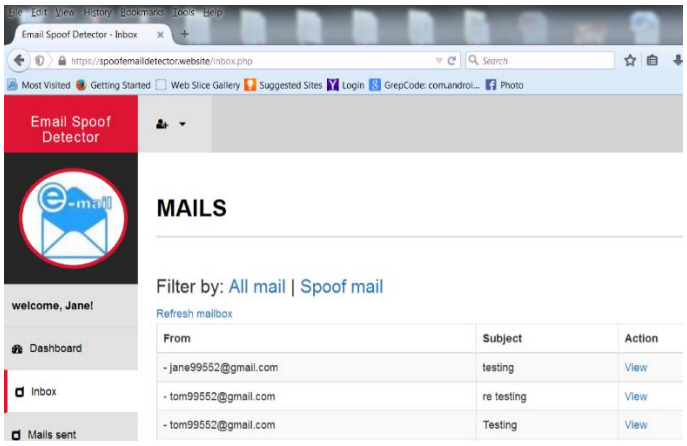

Figure.14. Filtering spoofed emails.

V. CONCLUSION

Email spoofing is a rising security risk which requires sophisticated mechanisms to counteract it so as to protect email users from its disastrous consequences. However, most well established anti- spoofing techniques are server oriented and hence do not provide users with explicit control on spoof attacks. In [15] a client oriented anti-spoofing application was proposed but its use was limited to a single subnet. This paper has built upon the work of [15] to propose a web-based client oriented anti-spoofing application that is globally deployable. Moreover, the application provides a more efficient handling of alert messages and spoofed emails which do not clog the user's inbox as was the case in [15]. Additionally, receivers can send explicit feedback notification messages to senders who have been spoofed. Hence the proposed application is not only easily deployable but provides a straightforward solution to monitor and control email spoofing. An interesting future work would be to enhance the application's interface or convert it into a plugin that could integrate conventional email applications such as gmail and yahoomail.

ACKNOWLEDGMENT

The authors would like to acknowledge the financial support of the University of Mauritius.

REFERENCES

[1] The Radicati Group Inc, Email statistic report, 2015-2019. Available: www.radicati.com/wp/wp-content/uploads/2015/02/Email-Statistics-Report-2015-2019- Executive-Summary.pdf [Accessed : January 2015].

[2] Radicati Group Inc, Emails Statistics Report 2013-2017, Available: http://finance.yahoo.com/news/radicati-group-releases-email-statistics-110000374.html [Accessed: January 2015].

[3] T. Bradley,"Minimize Your Exposure to Email Spoofing", 2012. Available from: http://www.pcworld.com/article/253305/minimize_your_exposure_to_email_spoofing.html [Accessed February 2015].

[4] P. Gil,"What Is an 'Email Spoof'? Is It a Type of Phishing Attack?", Available from: http://netforbeginners.about.com/od/p/f/email-spoof-phishing-attack.htm [Accessed February 2015].

[5] M. Rouse, "email spoofing", Search Security, 2007. Available from : http://searchsecurity.techtarget.com/definition/email-spoofing [Accessed February 2015].

[6] J.Mehnle, "Sender Policy Framework.Introduction", 2010. Available from: http://www.openspf.org/ [Accessed February 2013].

[7] Microsoft Corporation,"Sender ID Framework Verification System-Aims to Reduce Spam and Increase Safety Online", 2007. Available from: http://www.microsoft.com/mscorp/safety/technologies/senderid/default.mspx [Accessed February 2013].

[8] D. Crocker,T. Hansen and P. Hallam-Baker, "DomainKeys Identified Mail (DKIM) Service Overview", 2009. Available from: http://www.dkim.org/specs/rfc5585.pdf [Accessed January 2015].

[9] G. Clule,"Phishing attack attempts to steal Google passwords via Red Cross website", 2013. Available: http://nakedsecurity.sophos.com/2013/01/18/phishing-attack-google-passwords-red-cross/ [Accessed February 2015].

[10] J. Strickland,"10 Worst Computer Viruses of All Time", Available from: http://computer.howstuffworks.com/worst-computer-viruses.htm#page=10 [Accessed February 2015].

[11] G. Velmayil and S. Pannirselvam, "Detection and Removal of IP Spoofing Through Extended-Inter Domain Packet Filter Architecture" in Int Journal of Computer Applications, Vol. 49- No.17, July 2012.

[12] N. Jeyanthi and N.Ch.S.N. Iyengar, "Packet Resonance Strategy: A Spoof Attack Detection and Prevention Mechanism in Cloud Computing Environment" IJCNIS, Vol. 4, No. 3, December 2012.

[13] A.S. Zadgaonkar, Suresh Kashyap and Murari Chandra Patel, "Developing a Model to Detect E-mail Address Spoofing using Biometrics Technique"IJISME, ISSN: 2319-6386, Volume-1, Issue-6, May 2013.

[14] Nivedita Joshi, Dev Baloni and Sanjay Kumar, "Counter-Measures To Prevent Spoof E-Mail Tracking" in (IJATER) and 1st Intl Conf on Research in Science, Engineering & Management. (IOCRSEM 2014).

[15] D.Mooloo and T.P.Fowdur, "An SSL-Based Client-Oriented Anti-Spoofing Email Application", Proceedings of IEEE Africon 2013, Mauritius, 9-12 September 2013, pp.523-527. DOI: 10.1109/AFRCON.2013.6757757

[16] Kavi Groups,2008. How Email Really Works. Available:http://www.niso.org/khelp/kmlm/user_help/html/how_email_works.html [Accessed February 2015].

[17] Yatri Trivedi,2011. HTG Explains: How Does Email Work ? Available: http://www.howtogeek.com/56002/htg-explains-how-does-email-work/ [Accessed 21 February 2015].

[18] Marshall Brain and Tim Crosby,2011. How E-mail Works. Available: http://computer.howstuffworks.com/e-mail-messaging/email6.htm [Accessed February 2015].