New Technique for Cross Platform Availability Control over Electronic Messages

Ward Ahmed  
Department of Computer Science,  
Middle East University  
Amman, Jordan  
ward.it_se@yahoo.com

Dr Hebah H. O. Nasereddin  
Department of Computer Science,  
Middle East University,  
Amman, Jordan  
hnasereddi@meu.edu.jo

Abstract

In this research, a new technique is proposed to grant email senders control over sent messages even after they arrive at the receiver’s mailbox; the method suggests that the email gets encrypted by AES using a randomly generated key, the key is then stored on the sender server and an identifier is added to the header of the email, if the receiver wishes to read the email the receiver’s server must request the key from the sender server by sending the identifier, the sender can control the availability of the key and thus the ability of the receiver to read the email.

The experimental results suggest that the process of encryption has little impact on the overall time required to send the email, however, the process of requesting the key and decrypting the email required slightly more time than the normal process.

1. Introduction

One of the most widespread applications of computers nowadays that is used for person to person communication is Electronic Mail (Email). According to a survey done by Radicati Group in 2010, there were about 1.9 billion email users worldwide [1]. Emails are processed through the use of specific programs called Agents; on which each agent has specific tasks to do in the email processing paradigm. The Mail user Agents (MUA) is the one the user interacts with to send and read emails, the email is submitted from MUA to Mail Submission Agent (MSA) which temporarily queues outgoing emails, the email messages are then sent to the Mail Transfer Agent (MTA) whose function is to transfer the message to the target host by requesting the Mail eXchange (MX) record from the Domain Name Server (DNS), then it connects to the exchange server as Simple Mail Transfer Protocol (SMTP) client, the exchange server receives the email message from MTA and delivers it to Mail Delivery Agent (MDA) which has the responsibility of delivering the message to the mailbox of the receiver [2]. Emails are transferred from source to destination using SMTP which was designed to transfer email messages reliably through TCP on port 25 [3].

Advance Encryption Standard (AES) is a modern standard used for the encryption of digital data; it encrypts a 128-bit block from plaintext to cipher text and vice versa. AES supports three key lengths of 128-bit, 192-bit, 256-bit and requires the sender and the receiver to share the same key [4].

In this paper, section 2 will discuss some related researches, section 3 will explain the proposed technique in more detail. The results of the experiments are discussed in section 4.

2. Literature Review

[5] proposed a protocol for certified email, the protocol guaranteed the delivery of messages by relying on an online Trust Third Party (TTP) which has two pairs of keys, one pair for encryption and one pair for signing messages and verifying those signatures, the protocol allowed the email’s sender four authentication options to choose from.

[6] claimed that relying on web services for internet messaging provides the same functions of ordinary email protocols but with additional security functions and more flexibility by designing it as a family of web services, called WSEmail, on which the messages are Simple Object Access Protocol (SOAP) that use WSEmail features for message transmissions. The authors then concluded that WSEmail faces major problems with standardization and interoperability.
with SMTP, and suggested to mitigate that by writing more plugins to the SMTP.

[7] presented a novel approach based on the authorized Diffie-Hellman key agreement protocol to propose a certified email protocol with an offline TTP. Unlike other certified email protocols with an offline TTP, the proposed protocol encrypts the message with a key shared between the sender and the TTP without involving the TTP during the exchange.

[8] proposed a new model that aims at providing deniable authentication to protect personal privacy of the message’s sender. The authors assumed that each client has two pairs of keys; one pair is used for message encryption while the other is used for message authentication. The purpose of the design is to enable a specific message receiver to authenticate the message and to allow the message sender to deny generation of the message in order to protect his personal privacy.

[9] aimed to standardize a protocol to encrypt and digitally sign email correspondence, the authors discussed end-to-end security, reviewed some certified email protocols and proposed some improvements to enhance security, the paper demonstrated a prototype of the proposed protocol and the prototype was implemented using Component Object Model (COM) and web services.

Microsoft provided their email users with a function called “Message Recall” that aimed to retrieve an email message after being delivered to the destination. This function would not work if the recipient is not using Outlook, or is not logged on to the mail service provider, or is using Cached Exchange Mode and is working offline, or if the original message is moved from the recipient inbox, or if the opened first and marked as read [10].

[11] proposed a method for securing email communication exchange on mobile devices running on Android platform by using a combination of symmetric encryption (AES 128 bit), asymmetric encryption (RSA 1024 bit) and hash function (SHA-1). The study aimed at meeting the aspects of information security consisting of confidentiality, data integrity, authentication and nonrepudiation between two clients and claimed that the proposed model was successful in meeting these aspects; the authors also suggested making the system interoperable with other platforms on mobile devices and on desktop computers in the future.

[12] proposed a secure Email system that relied on Identity-Based Encryption (IBE), DNS and Proxy Service. The proposed model gave the sender control over his emails by ensuring that only intended recipients are allowed to view the encrypted email. The encryption/decryption process is performed by the proxy server. [13] proposed a protocol called Undo Sent Email (USE) which aims at retrieving an email after it has been sent to the receiver if the sender manages to pull the email first from the POP3/IMAP server. The proposed protocol has major two shortcomings: if the receiver has read the email first then the email cannot be retrieved successfully, the proposed protocol suffers when the domain names of the sender and the receiver are not the same, which means the protocol cannot function between cross-platform email service providers.

3. Proposed Methodology

The proposed model aims at granting control to the sender of an email message over its availability to the receiver, the sender should be able to deter the receiver from reading the message as wished, this can be achieved by encrypting the content of the message using AES algorithm by a secret key before sending it, the secret key is automatically generated by the server using Universally Unique Identifier Version 4 (V4 UUID) generation algorithm and is stored at the sender’s server, this key can be requested by the receiver at the time of opening the message through an API located at the sender’s server. The sender’s server will provide the secret key located in the server’s database needed to decrypt the message for the receiver in order to become readable. However, the availability of the secret key that is needed to decrypt the message can be modified by the sender which in turn controls its availability to the receiver server and thus to the receiver, the sender can make the key unavailable at any time whenever he wishes to, which is going to deter the receiver from decrypting and reading the message.

The author proposes a customized header called X-Ekey to indicate the location of the API that should be accessed in order to retrieve the secret key needed to decrypt the message; the MUA is going to be responsible of encrypting and submitting the key to the server.

3.1 The Proposed Sender Side Model

Figure 1 shows the proposed model from the sender’s side perspective...
Figure 1. The Proposed sender side model

The steps of the flowchart are explained in more details next.

1. The user writes the message.
2. The user enters the recipient’s email address.
3. The server generates a random key for message encryption by using auto generated UUID as an encryption key whose length is 128 bits.
4. The server encrypts the message using the key that was generated in step 3 and adds a new header “X-Ekey” along with sender server’s address to remove the possibility of duplicate identifiers, this header indicates that the message is encrypted and the receiver’s server needs to ask the sender’s server for the key in order to decrypt it. By doing so, the receiver will only be able to read the message as long as the sender keeps the key available.
5. When the encryption is done, the message is sent and the key is stored at the sender’s server.

3.2 The Proposed Receiver Side Model:

Figure 4 shows the proposed model from the sender’s side perspective.

1. After the receiver requests the message, the MUA will look for the API location.
2. The receiver MUA will post a request to the API along with a message identifier to receive the decryption key.
3. The sender server will return the requested key needed for decryption if its status is set to available, however if the sender changed the status of the key, the server will send an error code indicating that the key is unavailable.
4. If key is received successfully, the MUA will decrypt the message; however, if the MUA received an error code, the message cannot be read because it is encrypted.
4. Experimental Results

Several experiments were conducted to evaluate the performance of the proposed technique and its effect on the quality of the email exchange process. The efficiency of the technique is measured by four parameters which are:

- Time to send (TTS): the measure of the time required to generate an identifier and encryption key, encrypt the email and send it. TTS is measured in milliseconds by using the formula: \( TTS = T_1 - T_2 \) where: \( T_1 \) is the time when send command is received by the server and \( T_2 \) is the time when the message took off from the server.

- Time to read (TTR): the measure of the time required to read the email headers, request the decryption key located on a specific server and decrypt the email. TTR is measured in milliseconds by using the formula: \( TTR = T_1 - T_2 \) where: \( T_1 \) is the time when email headers are read and \( T_2 \) is the time when the decryption of message is completed. Note that the process of requesting and receiving the key falls within \( T_1 \) and \( T_2 \).

- Original email size (OES): is a measure of the email size including its headers before the encryption process and the addition of new headers, it is measured in Bytes.

- Prepared email size (PES): PES is the measure of the email size including its headers after the encryption process and the addition of new headers, it is measured in Bytes.

16 experiments were done on a shared host with 2.0 GB of dedicated RAM and were conducted on text based emails only to measure the normal process of sending an email against the proposed technique using the same random email characters for each one experiment.

4 examined results of 4 experiments when sending 1000 random characters as an email are being averaged out in Table 1 to help show differences in parameters among the tested techniques.

<table>
<thead>
<tr>
<th>Table 1. Average performance for 1000 character length experiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed Technique</td>
</tr>
<tr>
<td>OES</td>
</tr>
<tr>
<td>TTS</td>
</tr>
<tr>
<td>PES</td>
</tr>
<tr>
<td>TTR</td>
</tr>
</tbody>
</table>

4 examined results of 4 experiments when sending 5000 random characters as an email are being averaged out in Table 2 to help show differences in parameters among the tested techniques.

<table>
<thead>
<tr>
<th>Table 2. Average performance for 5000 character length experiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed Technique</td>
</tr>
<tr>
<td>OES</td>
</tr>
<tr>
<td>TTS</td>
</tr>
<tr>
<td>PES</td>
</tr>
<tr>
<td>TTR</td>
</tr>
</tbody>
</table>

4 examined results of 4 experiments when sending 10000 random characters as an email are being averaged out in Table 3 to help show differences in parameters among the tested techniques.

<table>
<thead>
<tr>
<th>Table 3. Average performance for 10000 character length experiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed Technique</td>
</tr>
<tr>
<td>OES</td>
</tr>
<tr>
<td>TTS</td>
</tr>
<tr>
<td>PES</td>
</tr>
<tr>
<td>TTR</td>
</tr>
</tbody>
</table>

4 examined results of 4 experiments when sending 20000 random characters as an email are being averaged out in Table 4 to help show differences in parameters among the tested techniques.

<table>
<thead>
<tr>
<th>Table 4. Average performance for 20000 character length experiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed Technique</td>
</tr>
<tr>
<td>OES</td>
</tr>
<tr>
<td>TTS</td>
</tr>
<tr>
<td>PES</td>
</tr>
<tr>
<td>TTR</td>
</tr>
</tbody>
</table>

4 examined results of 4 experiments when sending 10000 random characters as an email are being averaged out in Table 5 to help show differences in parameters among the tested techniques.

5. Conclusion

In conclusion, the proposed technique was successful in achieving the goals it aimed to deliver. The proposed processes were compared against the standard methodology of email processing and there have been short delays (in milliseconds) in the time the receiver’s server takes to read emails, this is due to the added processes of encryption key requesting and the decryption of the...
message, while there was a relatively little impact on the time it takes the sender’s server to send emails and changes in the size of an email when being prepared for sending, those conclusions were based on the results of 16 experiments conducted to compare the proposed technique against the standard technique of email processing using 4 variant email sizes.

6. Future Work

1. Try different encryption techniques in hope to reduce the time overhead.
2. Implement RSA key exchange mechanism to provide robustness against key hijack during the key exchange process.
3. Implement the proposed method on MIME data types and measure its performance.
4. Measure the performance of the proposed method on a dedicated server environment.

References