Transparent System for E-mail Security

Tony Wu

Computer Science Department, the University of Auckland

{twu027}@ec.auckland.ac.nz

Abstract. E-mail Security is becoming major concern for people. Integrate the security functions such as cryptographic technology into the existing e-mail system is a good approach since the antivirus software doesn't protect the security of the e-mail[1] systems. In this paper, I evaluated two different approaches to provide the e-mail security transparently. One approach [1] is to integrate the trust engine "PolicyMaker" developed by Matt et al [2]. The other one is a non-intrusive approach by Roth et al [3]. The term "transparent" means protect the user's e-mail without acquire too much attention from the users unless the security is breached. A framework for evaluating the e-mail security system has been developed in this paper. By comparing the two systems using this framework, the potential improvements have been proposed. The improvements include adding blacklist into the PolicyMaker, using PKI to verify and manage the keys in the non-intrusive approach etc.

1 Introduction

The term "e-mail system" includes the components such as software installed on the client and server side, hardware such as personal computer and e-mail server, e-mail protocols such as POP, IMAP, SMTP, etc. They are the necessary elements in order to send and receive e-mail messages. But in this paper, I only focus on the software being installed and configured at the client side. The term "e-mail system" will be used to describe the software components installed on the client side. A brief description to the two systems has been provided in this section. The primary and secondary goals of the each system are also discussed. The two systems worked on the different layers of the existing Internet e-mail infrastructure. In section 2, I provided the security framework for evaluating the e-mail security based on Lampson's security model [4]. By using this framework, the two systems have been compared and contrasted in section 3. Some potential improvements for the two systems have been proposed in section 4. Section 5 concludes with a summary.

[1] In fact, the antivirus software protects our computer system from the vulnerabilities exposed by the e-mail systems.

1.1 The PolicyMaker approach

The PolicyMaker [2] is a policy language for describing the "trust" and managing the access control. It takes the users' credentials and access queries as inputs, performs the compliance checking on the inputs and returns simple "Yes/No" answers to the queries. The credentials can be any "standard" certificates such as PGP and X.509 and native PolicyMaker certificates. The paper written by Levien et al [1] makes use of PolicyMaker to provide the security for the Internet e-mail transparently. It integrates the PolicyMaker engine into the mail transport agent (MTA) layer of the existing e-mail architecture. So the PolicyMaker can perform the security checking when the MTA routing and queuing the messages. The paper claims by adding the PolicyMaker engine into the MTA layer can provide the encryption of the messages without the user's notices (which means encrypt e-mail messages automatically). That's because the standard format of current e-mail can form queries to the PolicyMaker naturally and the security e-mail system they provided can search the database for corresponding certificates or keys automatically. The PolicyMaker will provide the encryption and decryption according to the policies specified by the user. For example, if the user puts the keyword "confidential" in the message header, the PolicyMaker will ensure the message is sent only when the receiver's public key is available. For the incoming messages, the user could specify policy such as only display the messages from Bob when the messages are encrypted. The PolicyMaker will evaluate the messages from Bob. If the messages are not encrypted, the user is not able to see them.

The primary goals of this approach are to provide the encryption and decryption silently as well as strong security. The user will not be involved with the cryptographic issues such as key or certificate management once the policy has been defined. The secondary goal of this approach is to give the user flexibilities when he or she specifies the policy. Because the PolicyMaker can cope with either PGP certificates or X.509 certificates, so the user is not bound any type of certificate, hence the flexibility will be provided as well.

1.2 The non-intrusive approach

The paper "*Security and usability engineering with particular attention to electronic mail*" by Roth et al [3] describes an approach opposites to the PolicyMaker approach in 1.1. It focuses on the mail user agent (MUA) to provide sufficient security and greater usability to the end users. The term "non-intrusive" means to not interrupt the user even when his e-mails are not totally secured. The user can choose to forgo the security or enforce the security. The authors' primary goal is the usability rather than security. They provided the justification to their primary goal by doing a cost-benefit analysis. The secondary goal of the paper is to provide the "out-of-band" key verification which is similar to Peter Gutmann's idea in [5]. The "out-of-band" key verification means when the sender or receiver wants to introduce the new keys to each other, the new keys must be signed by the old valid keys. They tried to avoid using the complicated PKI to verify the keys from the "strangers".

To support their primary goal, the authors have provided metaphors in their user interface design for the e-mail client. The metaphors are “Send as letter” and “Send as postcard”. “Send as letter” means send the e-mail securely and “Send as postcard” means send the e-mail in plain text. They argued that these metaphors are particularly useful. Cause all the people know the fact everyone can see what you have written on your postcards but not the letters. When the public keys of the receiver are not found, the “Send as letter” icon simply grayed out.

2 Framework for the evaluation

In [4] Lampson has provided the famous model of security evaluation. In this section, the specific framework for evaluating the e-mail security will be based on Lampson’s model.

2.1 **Lampson’s model**

There are four major headings in Lampson’s security model. They are the secrecy, the integrity, the availability and the accountability. The secrecy means keep the important information secret. There must be mechanisms to ensure the important information only can be viewed by authorized people. The integrity means we must protect information from unauthorized modification. The availability means the authorized people must have sufficient access to the system being protected. The accountability means auditing. The system should be able to trace the “bad” guys if they do anything bad to the system. This model is generic, it can be used to assess security systems for different areas such as banking system, online store etc. In this paper, I modified Lampson’s model slightly to create a specific framework for evaluating secure e-mail systems.

2.2 **Specific framework for evaluating the e-mail security**

I have listed five criteria in the framework. They are the following:

- Confidentiality
  The confidentiality in here has the same meaning in Lampson’s model. We must keep our e-mail messages secret if necessary. The proper use of cryptographic technology will ensure the confidentiality of our e-mail messages. Confidentiality should be primary goal for all the secure e-mail systems. This is the fundamental difference between a secure e-mail system and a non-secure e-mail system.

- Integrity
  The secure e-mail system should protect the messages being changed during the transmission. Especially in the commercial environment, any e-message might be used as evidence when the arguments arose between different par-

ties. Normally, the integrity will be achieved by digital signatures and hash-code of the messages.

- Simplicity
  Stamp suggests that any list of good security principles would likely include simplicity, open design (Kerckhoffs Principle) [6]. This is particular true in the e-mail security system. The simplicity is not only a functional requirement but also a security requirement as well. As the popularity of the Internet increases dramatically in the last decade, almost everyone has access to the Internet would have one or more e-mail accounts. But not every e-mail user is a computer expert. There is no point to develop a "perfect" secure e-mail system but it is impossible for normal users to install it by themselves. The simplicity also includes the usability. If the system is too hard to use, I believe the most of users will simply forgo the security functions. So the simplicity is an important attribute of a good secure e-mail system.

- Auditable
  Auditable means the accountability in Lampson's model. In the secure e-mail system, be able to trace the "bad" people would make the "bad" people think before they do.

- Transparency
  A good secure e-mail system should hide the complex encryption and decryption details from the users. The users normally have no idea what the "public key" or "certificates" mean. This is also the feature the two e-mail system claimed to have. This attribute is not as same as the simplicity. If the system is very user friendly, the user will not care too much if the security functions are transparent or not.

In this framework, I replaced the availability from Lampson's model by simplicity. That is because the availability attribute should belong to the server side of a complete e-mail system. As shown in figure1, the green parts mean the trusted territories. We have to trust the computer we are using to send e-mails and the e-mail servers such as hotmail, Gmail etc. But we have no reason to trust the communication channels and the receiver. They are displayed in red color. If we are receiving e-mails, we then have to trust the computer we are using and the servers, but not the sender. I have also added the transparency into this framework. It is the functional requirement of these systems. It can be argued if this requirement is really necessary for a secure e-mail system. The discussion is in more detail in section 4.

3 Compare and contrast

The two e-mail systems have been compared in this section. According to the framework, their features are discussed as well. Table 1 shows the performances of the two systems measured by my framework.

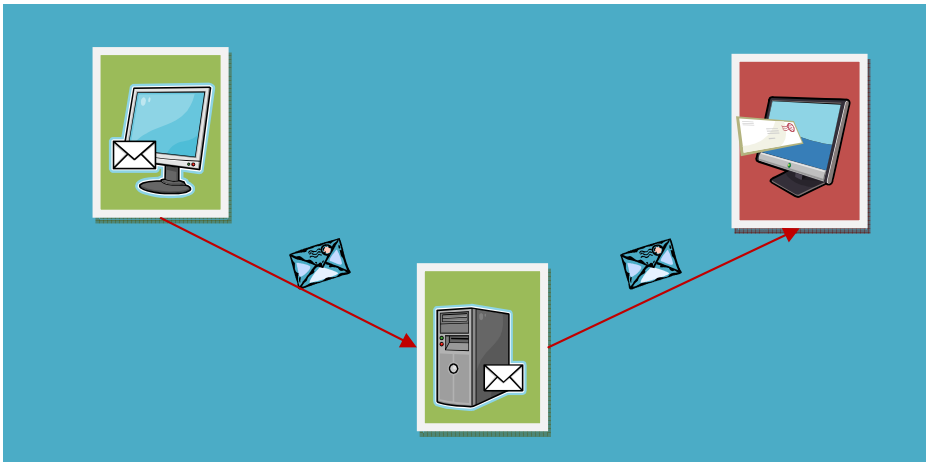

Figure 1: Security diagram of E-mail system

Criteria	PolicyMaker approach	Non-intrusive approach
Confidentiality	Strong	Medium
Integrity	Strong	Medium
Simplicity	Weak	Strong
Auditable	None	Strong
Transparency	Strong	Medium

Table 1: Comparison for the two system

3.1 Appreciations and criticisms for the two systems

The advantages and disadvantages of both systems have been listed in table 1. They are discussed in more detail in here.

3.1.1 The confidentiality and integrity of both systems

Both of the systems used the cryptographic technology to achieve the confidentiality and integrity. The difference is the PolicyMaker approach requires a formalization of trust by certificates and explicit policies [3] and the non-intrusive approach uses the "out of band" key verification. The PolicyMaker approach is susceptible to malicious user queries and the non-intrusive approach is susceptible to the "man-in-the-middle" attack.

The PolicyMaker approach can be visualized by figure 2.

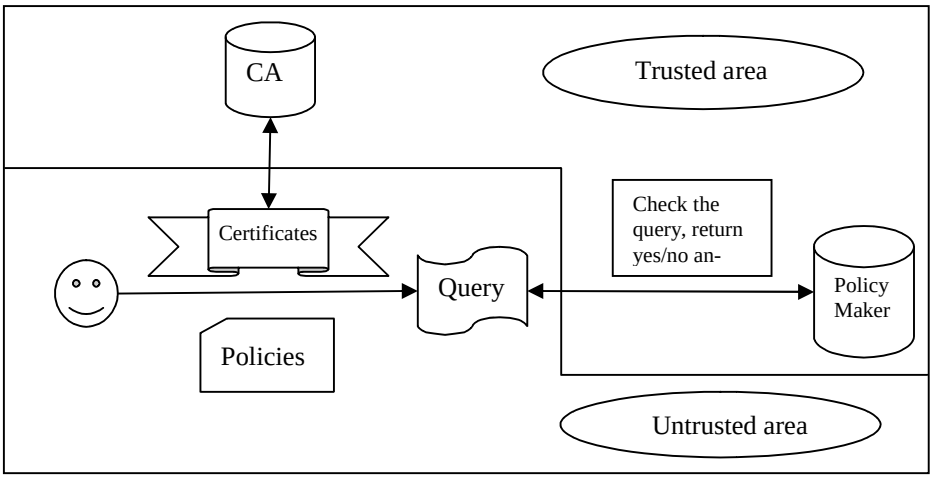

Figure 2: PolicyMaker approach

The security boundary has been drawn between the queries and the PolicyMaker engine. The queries from the user cannot be trusted. In Levien's e-mail system, the queries are formed by the e-mail message header. It may look like this:

"From: sender's name
Organization: sender's organization
Subject: confidential: sender's subject"

The PolicyMaker will look at the "From" header to search for the corresponding credentials in the database. Based on the predefined policy, the PolicyMaker will decide if the message can be sent confidentially. But it is not too difficult to change the e-mail header to form a bad query which can confuse or even crash the PolicyMaker engine. The bad user would launch an attack similar to "SQL injection". The PolicyMaker cannot deal with negative credentials. For example, if we specified the policy such as "I would like to receive letters from Bob if and only if his key is not revoked by his certificate authority". The PolicyMaker couldn't perform compliance checking on such policy. If we can trust the user, this system will provide strong confidentiality and integrity, because the certificates or public keys are verified by the certificate authorities. The CAs must be trusted in any security system.

The non-intrusive approach relies on the old keys to introduce the new keys rather than the CAs. Each peer would maintain a list of the keys used by each other. So the first time to exchange the key with someone you never know before is a big leap of faith. The impersonation attack by active adversaries who substitute the keys for the keys of legitimate users cannot be completely avoided. If Alice received the initial key from Trudy but she thinks the key is from Bob, then Alice and Bob will not aware the existence of Trudy unless Trudy makes a mistake when she transcodes all the e-mail

messages. The authors claimed the user of this system has advantage is that the Murphy's law is in the user's favor – the attackers will make a mistake eventually [3]. But can we rely on the enemy's mistakes to win the battle? On the other hand, if we can successfully exchange the key for the first time, then we can provide sufficient confidentiality and integrity.

3.1.2 Simplicity and transparency of both systems

Compare the two approaches, the non-intrusive approach provides simpler user interface for the e-mail client and it is easier to setup. This approach also avoided using any technical jargons to describe the security functions. The users don't see anything like encryption or decryption in the user interface. These technical words will make the users feel lost and mystery about the system they are using. Instead, they used words like "Send as letter" or "Send as postcard". This idea is particular useful for the normal users. Everyone has the experiences of sending letters and postcards. Everyone can see what you have written on the postcard but not the letters. So instantly the users would get a feeling that if they choose "Send as letter", the messages will be more secure than "Send as postcard". The authors also have carried out an experiment to see the reactions from the e-mail users when they see the metaphors. The results show that most of the users who participant the experiment would guess the meaning of these metaphors correctly. For the PolicyMaker approach, to defin all the necessary policies is the biggest problem for normal users. The policy must be written in a "safe" language which the PolicyMaker can understand. This is not the job for end users. But the PolicyMaker system is not able to predefine the policies for the end users, because each user's policy is different.

Once the policies are properly defined, the user would use this system as normal e-mail system. The user will even not aware the existence of the PolicyMaker. The e-mail messages composed by the user will be sent to the PolicyMaker first, the PolicyMaker will do the security transform and deliver the encrypted messages to the MTA layer. For the non-intrusive approach, the security functions are not completely transparent to the users.

3.1.3 The auditing of both systems

Auditing is the security function that all the security system should have. But the e-mail system uses PolicyMaker doesn't have this function at all. The authors didn't even mention this in their paper. Because the PolicyMaker only return "Yes/No" answers to the user, the user only gets to know if the messages are sent successful or not. By contrast, even the non-intrusive approach is susceptible to the "man-in-the-middle" attack, but the attack cannot leave the channel without being detected. The attacker will use his own key to transcodes the messages. This will give us a clue that who the attacker might be.

4 Discussion

In this section, I discussed some potential improvements to the two systems.

4.1 PolicyMaker approach

The PolicyMaker is a very good approach to manage the access control. But it still needs some improvements when we integrate it into the e-mail system. For the auditing problem, it is not too difficult to solve. In Lampson's paper [4], he provides a security diagram.

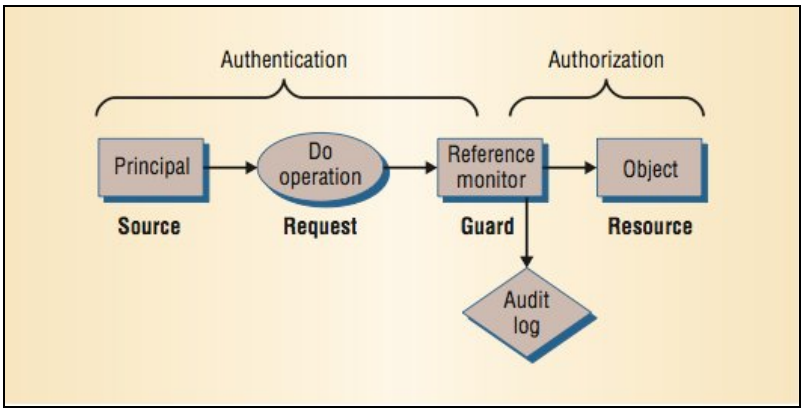

Figure 3: Lampson's security diagram

The PolicyMaker engine in the e-mail system will act as the reference monitor in Lampson's diagram. So it is possible to provide the audit log for PolicyMaker as well. The PolicyMaker can be modified to not just return Boolean answer to the user but also records the information such as who is sending the request, the reason why the request fails etc.

One of the inventors of the PolicyMaker, Matt specified the PolicyMaker only works correctly when the policy is monotonic. Some policies make explicit use of "negative credentials" such as revocation list cannot be checked by the PolicyMaker [7]. But in my opinion, this is not too important in e-mail system which uses the PolicyMaker.
There are several reasons for this. First, most of the time we need the positive credentials only. For example, if we received a letter says it is from Bob, we want to be sure it is really from Bob but not Trudy. If Bob can provide keys to proof he is Bob, then he doesn't have to proof that he is not Trudy or anyone else. Second, the negative credentials will be used in situation such as "I don't want to receive letters from my ex-wife". In such case, we can implement a blacklist. The PolicyMaker will check the "from" header first when it received letters. If the name in the "from" header is in our blacklist, we can simply block or destroy the letter without the knowledge of the user.

Adding the blacklist functionality into the PolicyMaker will not be too difficult. Third, the PolicyMaker will perform faster and be more robust if it only deals with the positive credentials.

Once we add these functions into PolicyMaker, it will be a good choice for the commercial and military users.

4.2 Non-intrusive approach

In this approach, the security functions are not completely transparent to the users. But I argue this might be a good thing for the users. For example, if people spend enough money in computer security, their system might be secured. What do these people get in return of their money spent in security? The answer is "nothing happens to their system". As a consequence of "nothing to happen", some of the people would reduce their security budget. If the users don't see all the security functions and the users don't see the security problems occurred very often, they might think there are not so many threats on the Internet. Hence they might stop paying for the security. So it will be a good idea to keep the security functions visible to the normal users if the security functions are really easy to use. The non-intrusive approach satisfied this requirement.

This approach also argues that the "out-of-band" key verification has advantages over the PKI. I cannot agree with this. Normal users will not change their keys very often. This is justified by their experiment in [3]. The key exchanging for the first time becomes even more important. But the "out-of-band" key verification makes the attackers more difficult only if the users change the keys very often. To provide better security, we could just use the PKI to verify the keys. When we are surfing the Internet, sometimes we can have the certificate error. The user then has the choice. He can either choose to continue or just close the page. The non-intrusive system can just handle the certificate error this way. We can highlight the name of the sender if his certificate cannot be verified by the CAs. The user can choose to send/receive the letters in such case. So the usability will not be affected.

Because of the simplicity and usability the non-intrusive system provides, this system is a good choice for the non-commercial users.

5 Conclusion

The simplicity is not just a functional requirement but as well as security requirement. In order to make most of the e-mail users to protect their e-mail messages actively, the simplicity is very important. We are not looking for the perfect secure e-mail system since it is not possible to have one. We are actually looking for the system can balance the security and the usability. By comparing the two systems, the non-intrusive system provides the better trade-offs between the simplicity and security for the non-

commercial users. As we discussed in section 4, if we can use the certificates provided by the CAs to make the first key exchange between the peers, this approach will provide sufficient security. I also argued that the transparency is not as important as the simplicity or usability in the secure e-mail system. In fact, it might be a good thing to let the user to see the security functions. This might help the users to keep the computer security in mind. The PolicyMaker would be the choice for the commercial and military users due to its complex setup and maintenance (the policies may change overtime).

References

[1] R. Levien, L. McCarthy, M. Blaze. *Transparent Internet E-mail Security (DRAFT)*. Unpublished manuscript, 16 pages, 9 August 1996. Available on third author's web-site at http://crypto.com/ papers/email.pdf, 3 October 2007.

[2] M. Blaze, J. Feigenbaum, and J. Lacy. *Decentralized Trust Management*. In Proc. of the 17th Symposium on Security and Privacy, pages 164-173. IEEE Computer Society Press, Los Alamitos, 1996.

[3] V. Roth, T. Straub, K. Richter. *Security and usability engineering with particular attention to electronic mail*. International Journal of Human-Computer Studies Volume 63, Issues 1-2, Pages 51-73, July 2005

[4] B. Lampson. Computer security in the real world. IEEE Computer volume 37, issue 6, pages 37-46, June 2004.

[5] P. Gutmann. *Plug and play PKI: a PKI your mother can use*. In Proc. of the 12th Symposium on USENIX Security.Washington, DC, USA, August 2003.

[6] M. Stamp. *Information Security: Principles and Practice*, John Wiley & Sons, Inc., pages 325-338, September 2005.

[7] M. Blaze, J. Ioannidid, A.D. Keromytis. *Experience with the KeyNote Trust Management System: Applications and Future Directions*. In Proc. of the 1st International Conference on Trust Management, Crete, Greece, pages 284-300, May 2003.