Trustnet architecture for e-mail communication

David Pölz
University of Vienna
Research Lab Computational Technologies and Applications
Lenaugasse 2/8, A-1080 Vienna
Email: david.poelz@univie.ac.at

Wilfried N. Gansterer
University of Vienna
Research Lab Computational Technologies and Applications
Lenaugasse 2/8, A-1080 Vienna
Email: wilfried.gansterer@univie.ac.at

Abstract—In this paper we discuss a new architecture to reduce unsolicited e-mail messages. We propose a system architecture that introduces two classes of messages - trusted e-mail and e-mail from untrusted sources. Trusted e-mail messages are signed with an S/MIME signature. To address usability problems that occurred previously with S/MIME signatures, outgoing e-mail messages are automatically signed on the e-mail server without any user interaction. A validation of the signature by the receiving server classifies the message either as trusted or untrusted, which enables the receiver to employ additional security checks for untrusted messages or to omit these checks for trusted messages. A comparison of the proposed system to a common setup with spam and anti-virus filtering shows that the trustnet architecture not only reduces processing time but also significantly reduces the amount of data transferred.

I. INTRODUCTION

E-mail communication has become one of the most used applications of the internet. Although everybody uses it there are a few annoyances that affect an efficient usage of this medium. Unsolicited bulk e-mail ("spam") has become very common. Lately, these unsolicited e-mail messages evolved from a mere annoyance to a threat for the user. Phishing messages are sent to spy on private data and viruses can be spread via e-mail. This reduces the trust of the user in this otherwise convenient communication medium. Many users are unfortunately not aware of these shortcomings or do not know how to overcome these threats.

The possibility to sign and encrypt e-mail messages has been around for many years but so far has not been accepted by a broader user community due to usability problems. Therefore, it is necessary to build a layer of security that is transparent and almost invisible to the end-user. This paper presents such a software architecture that provides more security for the end-user. Additionally, it reduces costs for e-mail service providers in terms of spam and anti-virus filtering. The architecture proposed in this paper is based on a trust system between various e-mail service providers. All members of this trust community have a personal X.509 certificate and sign every outgoing e-mail message with this personal S/MIME certificate so that the receiver can validate whether the sender is the one who he claims to be. A further validation of the certificate reveals if the sender is a member of the trust community - thus it becomes possible to prioritize the handling of the trusted e-mail over untrusted e-mail.

II. RELATED WORK

Most current anti spam methods concentrate on filtering spam after the delivery of the e-mail message. Post-send measures like rule based systems, blacklists and bayesian filters address the spam problem at this stage. Rule based system were efficient in the beginning of the spam era but soon spammers adopted to the rules and adjusted spam messages to circumvent this kind of filters.

Another method to block unsolicited e-mail are blacklists (e.g. DNSBL\(^1\), SBL\(^2\), PBL\(^3\), ...). It is very time consuming to keep such lists updated. Furthermore, extensive evaluations of blacklists showed that IP addresses in these lists are used by spammers only for a very short time. More specifically, 90\% of the listed IPs are used only three days for spam delivery [1]. A further disadvantage of blacklists is that dynamic IP addresses can be assigned to a not spamming e-mail server later and this server is then wrongly blacklisted. They also are quite traffic intensive due to regular updates and huge amounts of listed IP addresses.

Bayesian filters were a reliable anti spam method for some time but due to obfuscation techniques and randomized spam messages their efficiency decreased. A retraining of these filters is needed on a regular basis. Recent commercial and open source anti spam filters such as Spamassassin\(^4\) use a combination of the above mentioned filtering techniques. A machine learning based method for classifying e-mail messages especially considering dangerous phishing messages that closely resemble legitimate ham messages has been presented in [2]. A more general framework which intercepts e-mail messages before the are delivered has been proposed in [3]. It temporarily blocks incoming messages by employing greylisting. An additional reputation system minimizes the drawbacks of greylisting (delayed delivery) and ensures a timely delivery of the e-mail messages. Pre-send measures block unsolicited e-mail messages before they are actually sent. In [4] a method has been suggested that blocks spam before the message is sent by applying a human interactive proof (HIP) for every sent message which increases the cost for every sent message for the spammer. Another way to impose additional costs

\(^1\)http://www.heise.de/ix/nixspam/dnsbl/
\(^2\)http://www.spamhaus.org/sbl/
\(^3\)http://www.spamhaus.org/pbl/
\(^4\)http://spamassassin.apache.org/
for sending spam messages is Hashcash [5] which forces the spammer to invest computational time for each sent message. Other suggested solutions to the spam problem include a redesign of the SMTP protocol. An example of a new protocol that shifts part of the control over the message transfer from the sender to the receiver of the e-mail is DMTP which was introduced in [6]. The transition from one widely used protocol to a new improved protocol is always difficult and the major drawback of such anti spam measures. Therefore the approach discussed in this paper suggests a method that is compatible with current technologies and provides a local solution for a global problem.

Filtering unsolicited e-mail is only part of a comprehensive e-mail security system. Often encryption and signatures are required to provide confidentiality and trust. The encryption and signing methods PGP [7] and S/MIME [8] are the most common ones. In [9] a software solution has been proposed to automatically sign e-mail traffic within domain boundaries. Pereira et al. also suggested in [10] an opportunistic encryption of e-mail messages with S/MIME. In contrast to the architecture discussed in this paper which uses S/MIME features to classify e-mail messages, these approaches emphasized the improved transfer and system security.

Various commercial tools such as Entrust Entelligence Messaging Server, Z1 SecureMail Gateway or Tumbleweed Secure Messenger act as e-mail proxy and provide the possibility to sign and encrypt e-mail messages. However, these products do not provide any possibility to react to certain certificates and are therefore not suited to build a trustnet architecture as we propose in this paper. Whitten et al. showed in [11] that the average user is not capable of using basic methods to secure his e-mail traffic. Usability problems and a lack of knowledge about basic encryption and signature techniques are the main reason for this and these drawbacks still exist in current e-mail encryption and signing software.

III. BACKGROUND

S/MIME (Secure / Multipurpose Internet Mail Extensions) is a standard that allows to encrypt and sign e-mail messages. It is built upon the well established MIME content type specifications which describe which kind of content a data block contains. For example the part of an e-mail which contains the binary data of a picture declares its MIME type as image/jpeg. E-mail messages that use S/MIME’s functionality declare their MIME type for encrypted messages as application/pkcs7-mime and for signed e-mail messages as multipart/signed or application/(x-)pkcs7-signature. The S/MIME standard heavily relies on the Public Key Cryptography Standard 7 (PKCS#7)[12].

The S/MIME standard uses a public key infrastructure based on certificates. These certificates are issued by a certification agency (CA). A certification agency can also issue certificates that have the right to create certificates themselves (SubCA). In Figure 1, the root certificate would be owned by a CA. Internet service providers and other big companies can be certified to become a SubCA that can issue certificates for their users.

For the encryption of an e-mail the sender would need the public key of the receiver, which might be complicated to obtain. For producing a signature of a message the sender signs the message with his private key and attaches his public key to the message. Furthermore, the sender’s certificate (without the private key) is also appended to the e-mail message. This certificate includes the certificate path up to the root CA. That way the receiver has everything he needs to validate the message. On the one hand, he has the key to verify the integrity of the message and on the other hand, he can verify that the sender owns a certificate issued by the same CA. This is called the chain of trust.

In contrast to PGP, which needs additional software packages, the S/MIME standard is supported natively by all major e-mail clients on Windows, Linux and Apple. This fact, the availability of well documented APIs for S/MIME, and the chain of trust validation supported our decision to choose S/MIME as the signing algorithm for the trustnet architecture.

IV. CREATING A NETWORK OF TRUST

Based on the methods and technologies summarized in Section III, we propose an architecture that automatically signs outgoing e-mail messages and prioritizes trusted e-mails. Note that this approach neither requires any change in the settings of the user’s client software nor is any other additional interaction burdened upon the user.

A. System architecture

In the following we introduce the term “trustnet”. A trustnet consists of an arbitrary number of members who trust each other not to send any malicious or unsolicited e-mail messages. Thus, e-mail messages from one trustnet member to another do not need to run through a spam or anti virus system. Such messages can be handled with priority and will not be caught in a filter.
Since the advantages of a trustnet increase when more e-mail messages are sent this model is intended for big companies or internet service providers where the setup costs are quickly compensated for by the reduced e-mail filtering efforts. A participation of other members of the trustnet, for example smaller businesses or private users that individually operate an e-mail server, is also possible. These members would need to use a scaled down version of the ISP e-mail server software that has only a single certificate pre-installed that enables it to send signed e-mail messages. That way trustnet members can only send e-mail via trusted e-mail servers. This approach improves security since we assume that most small businesses and private users do not have the knowledge or resources to properly secure the system that the e-mail server is running on.

We also suggest to block outgoing connections to port 25 (smtp). That would further enhance the security of the trustnet and eliminates the possibility to send unsigned messages from computers that are within the responsibility of the trustnet operator. Users who need to operate their own e-mail server can do this by installing the provided scaled down trustnet e-mail server software or use one of the "central" ISP e-mail servers directly. These users need to be excluded from blocking port 25.

As distinguishing mark between messages from trusted sources and untrusted sources we chose the S/MIME signature. Every e-mail which is sent from a server participating in the trustnet is digitally signed by a certificate that can be assigned to a specific user. This certificate is automatically created at the first use. Messages to be delivered to an address outside of the trustnet are signed, too. By using the widely supported S/MIME signature users within the trustnet enjoy the advantage that even users outside the trustnet receive signed e-mail messages. This improves the trustworthiness of trustnet users and provides incentives to outsiders to join the trustnet. Certificate management is very flexible in this software architecture. Every SubCA is responsible for managing and providing security for its own customers because the customer data is nearest to the SubCA and any privacy issues should be addressed at the lowest level possible where personal data is available.

In case of a misuse of the trusted relationship a single certificate can be revoked and thus the user excluded from the trustnet. This is possible by using certificate revocation lists (CRLs) that are employed withing the S/MIME architecture and publish revoked certificates to other members of the trustnet community.

As shown in Figure 2 we suggest the use of two different types of e-mail servers within the trustnet - enterprise e-mail servers and personal e-mail servers. The enterprise e-mail server is suited for e-mail service providers and big companies and can handle a large numbers of users. Additionally, it integrates the key management for creating new user certificates and administering the certificate revocation lists, as well as the storing of the certificates for the multiple users, who do not use a personal e-mail server. The personal e-mail server has only one pre-installed certificate and thus matches the requirements of private users or small business owners who want/need to operate an own e-mail server.

B. Implementation

The architecture described above was implemented as a plugin to the Apache James mail server. Apache James was chosen because S/MIME APIs are readily available for java implementations.

The implementations of encryption and digital signatures were assisted by the bouncy castle java libraries. Due to export restrictions Sun's java development kit only allows a key length of up to 1024 bit for asymmetric encryption and up to 56 bit for symmetric encryption. For a stronger and more secure signature it was necessary to import the java unlimited encryption policy file which enables java programs to use a key length greater than 1024 bits.

In the current implementation the enterprise e-mail server stores the private keys and the certificates for each user in a MYSQL database. This data amounts to 4 kilobyte per user and thus it is easy to store the certificate database directly into the RAM memory for a rapid access.

C. Vulnerability analysis

A major disadvantage of the trustnet concept designed here is that as soon as an attacker gains access to the trustnet infrastructure a lot of mischief is possible. All kind of e-mail messages could be sent to all trustnet members that circumvent spam filters and are therefore guaranteed to reach the users. Widespread spam campaigns would be a rather harmless consequence. A more serious threat pose phishing e-mail messages or messages containing viruses or trojan software. To exploit the trustnet architecture an attacker would need to gain access to a valid certificate. Two possible attack points to the trustnet exist:

- The enterprise e-mail servers. A security breach on such a server would be a major issue since all private certificates of trustnet members are stored on a central server.
- The personal e-mail servers. A security breach on such a server only compromises a single certificate.

The certificates stored on the enterprise e-mail servers should be safe since these servers are administered by IT professionals. The risk for an unauthorized access to a certificate on the personal e-mail server is rather high because these servers can also be operated by uneducated users. A successful attack of an enterprise server would result in the revocation of the SubCAs certification and affect many users. Besides that, the provider of the enterprise server loses the trust of other trustnet members and has to take serious efforts to convince the other trustnet members to get a new SubCA certificate and be trusted again. A security break on a personal e-mail server grants access to the trustnet as well. In this situation it is rather simple for the provider to invalidate the certificate.

\[http://www.bouncycastle.org/java.html\]
affected certificate by including it in the certificate revocation list and therefore revoking the trustnet privilege.

Of course there is still the possibility that a trustnet member misuses the network and sends unsolicited e-mail messages. The same measures as a security breach on a personal e-mail server can be applied here. A revocation of the certificate excludes this member from further trustnet access.

Other rather organizational considerations that may complicate the foundation of a trustnet are:

- Setting reasonable conditions and security standards to join the trustnet,
- Establishing the same level of trust between the members,
- Possible privacy issues, due to personal data included in the certificates, and
- Agreeing about a common administration for the certificates

V. Evaluation

To demonstrate the advantage of using a trustnet over a classic spam/anti virus filter two different e-mail server configurations were compared. Both were set up on the same two machines and used the Apache James 2.3.1 e-mail server.

- Trustnet: A trustnet was established between the two machines. One machine was signing the outgoing messages - the other one received the messages and validated the signature. Both used the Apache James Plugin as described in Section IV-B.
- Spam/Anti-Virus Setup: Apache James offers plugins to check incoming e-mail with ClamAV\(^6\), an open source anti-virus scanner, and Spamassassin. Incoming e-mail messages are handed to ClamAV and Spamassassin, and Apache James processes the results. ClamAV version 0.94.2 and SpamAssassin version 3.2.5 were used.

Both setups processed incoming e-mail messages with ten concurrent threads.

The test data consisted of 1000 ham and 2000 spam e-mail messages randomly selected from the TREC 2007 spam corpus (http://plg.uwaterloo.ca/~gvicomac/treccorpus07/). In total, these 3000 e-mail messages amounted to 23.6MB of data.

<table>
<thead>
<tr>
<th></th>
<th>Trustnet</th>
<th>SA/ClamAV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Transfer</td>
<td>32.1 MB</td>
<td>53.1 MB</td>
</tr>
<tr>
<td>Processing Time</td>
<td>832 s</td>
<td>1754 s</td>
</tr>
<tr>
<td>Avg. CPU load</td>
<td>4.5</td>
<td>2.5</td>
</tr>
<tr>
<td>Memory Usage</td>
<td>400 MB</td>
<td>500 MB</td>
</tr>
</tbody>
</table>

Table I: Performance comparison of Trustnet vs. SA/ClamAV setup

Table I shows a comparison of the two setups. The most significant difference was the amount of data transferred between the two e-mail servers. SpamAssassin uses a lot of online queries to classify the messages which increases the amount of data transferred. The slight increase of transferred e-mail data compared to the original data for the trustnet system can be explained by the additional attachments of the S/MIME certificates. The processing time for the 3000 e-mail messages could be reduced by 50% by the trustnet system. However, this reduction was only possible because the CPU load, which was measured with the top-command under Ubuntu 8.10 linux, increased during the test run. A possible reason for that can be the online queries of SpamAssassin again, which delayed the quick processing in the SA/ClamAV setup. An inspection of the java processes with JConsole revealed the memory usage during the evaluation run and again the trustnet system produced better results than the SA/ClamAV setup.

Another very positive feature of the trustnet architecture is that not only messages within the trustnet are digitally signed, but all outgoing e-mail as well. This is very convenient as popular e-mail service providers do not filter signed e-mail messages. An evaluation of sending a spam e-mail message that was signed with a self signed certificate to Gmail, Hotmail and Yahoo has surprisingly shown that the same message was delivered to the inbox when it was signed, but was blocked when it was unsigned.
VI. CONCLUSION

A new architecture for e-mail transfer was presented in this paper. The process involves signing outgoing e-mail messages with an S/MIME signature. This signature is validated on the receiving mail server via the chain of trust of the certificates. Earlier efforts to establish digitally signed or encrypted e-mail messages were not successful in practice because of usability issues. Hence, our trustnet system works directly on the e-mail server without any user interaction. The results of a performance comparison of the trustnet architecture with a common Spamassassin/ClamAV setup showed important advantages for the trustnet architecture.

ACKNOWLEDGMENT

The authors would like to thank the Internet Privatstiftung Austria for supporting the work summarized here.

REFERENCES