Experiences With Secure Electronic Mail

Michael Roe
Computer Security Group
Cambridge University Computer Laboratory

13th May 1993

Abstract

The PASSWORD project provided a pilot secure electronic mail service across Europe. This paper describes some of the lessons we learned.

Introduction

PASSWORD (Piloting Authentication and Security Services within OSI Research and Development) was a two-year EC-funded research project bringing together academic and commercial partners from the UK, France and Germany. The objective of the PASSWORD project was to provide an information security infrastructure for a real community of users, in order to gain practical experience of the operational problems involved.

Our main goal was to gain understanding of how to provide security for a very large and geographically dispersed user community. Techniques for providing secure messaging within small, restricted communities were well understood. We wished to investigate the technological problems involved in providing a much wider and more open service. It was an assumption of the project that the solution lay in choosing the right sort of data communications protocols. There is, of course, more to it than that. Other projects have examined the same problem from alternative perspectives, such as the legal implications [1].

To provide the project with a focus, it was necessary to be very specific about whose problems we were trying to solve. As our target community, we chose academic and commercial researchers working on collaborative projects. The reasons for this choice were that members of this community were easily persuaded to take part in pilot projects, and that we felt this community had security needs which might be met with current technology.

The security requirements we addressed can be summarised as follows:

- Members of the community regularly exchange information with each other, primarily by electronic mail.
- Some of this information should be protected from disclosure to unauthorised persons, e.g. because it has commercial value.
- There is a strong requirement for *integrity.* Messages should be protected from being altered in transit, and it must be possible to know who a message came from.
- Members of the community do not necessarily completely trust each other, even if they are engaged in collaborative work. It is necessary to protect users from each other, as well as from outsiders. This is particularly true across organisational boundaries. Even if two companies are both authorised users of the system, it is unacceptable for employees of one company to have unrestricted access to all of the other company's private data, or to be able to forge the signatures of the other company's employees.

It is important to note that all uses of the system are ultimately interactions between people, and that these interactions take place in a social and legal context. In the simplest case, two users have met each other personally before hand, have agreed to do some work together, and are now using the electronic communications system to carry out that work. It is also highly desirable for people to be able to do collaborative work together even if they have never met face-to-face. However, even if they have never met there always exists some form of prior agreement that gives each communicating person rights and obligations in respect of the other, and hence makes interaction meaningful. This agreement can be created by them being members of the same organisation, or members of different organisations that have made a mutual agreement, and so on.

Although PASSWORD was primarily concerned with people acting in the role of employees, this principal is true more generally. Being citizens of a country, and subject to its laws, also creates similar relationships between individuals. It is almost always the case that membership of any kind of social structure confers rights and obligations.

This conference is about data communications, not sociology, so I'll quickly get the point and explain why this is important. If you start by regarding an authentication protocol as an exercise in pure mathematics, you are in danger of building a "pure" authentication protocol. This provides one party with the assurance that it is communication with another party, who is named by an essentially meaningless symbol. The only assurance given is that this symbol will never be used to identify anything else. Such a protocol gives absolutely no information about *what* you're really communicating with. This is not, by itself, useful; to act on a message, the recipient usually needs to know something about the senders rights and obligations.

It is conventional in computer security to separate authentication from access control. In the access control process, the identity obtained from the authentication process is used to look up (e.g. in a table) the corresponding entity's access rights — what it is permitted to do. This is fine for the case of a person accessing a computer (e.g. an on-line database); the target computer system can do an automatic comparison of the attempted access with the person's access rights and either allow or disallow the access, as appropriate.

This is much less satisfactory for person to person communication using a electronic network, which is the case with electronic mail (or the telephone!). The meaning of what is being communicated is typically expressed in natural language (e.g. a request to buy something; a contract; a proposal of marriage) and it is neither possible nor appropriate for the computer system to make the decision as to how to respond. In some sense, the access control process is in the mind of the person on the receiving end of the message.

However, unlike a computer program, a human being would like more than just an essentially meaningless number when deciding how to respond to a message.

Electronic mail security is in some ways easier to provide than other applications, as you don't need to specify and implement an access control mechanism. (This was one of the reasons the PASSWORD project concentrated on electronic mail). However, you do need to provide something else instead, and this "something else" can be much harder to understand than access control. What you need to do is to provide the (human) recipient of the message with all the information they need to decide how to react to it, in some simple and convenient visual form.

The real name of the message's sender is a good start. If you know who I am (e.g. by having met me before) then knowing that a message came from "Michael Roe" may be sufficient. You do of course need to be sure it came from *this* "Michael Roe" and not some other person with the same name. So the program had better provide some additional distinguishing information, such as who I work for or where I live.

If the message comes from someone you've never heard of, their name is not particularly helpful. The name of the organisation they work for may be all you need (assuming the message is between people acting in their role as employees). For example, suppose your company is collaborating with Cambridge University Computer Laboratory on the "foobar" project, and you get a message related to that project which is certified to be from one of our full-time research staff. This may be enough to persuade you that the message is legitimate.

In the last paragraph I said “certified”. This raises another series of questions: Who said that the message came from that person? Were they *authorised* to say that? What steps did they take to be sure, and what confidence can you place in their judgement?

In the PASSWORD project, the secure e-mail protocols we used were X.400(88) [2] and Internet Privacy Enhance Mail [3]. Although there are many important differences between these protocols, they use the same approach to identifying and authenticating users, so I will describe them both together. They both meet the requirements outlined above by providing the following facilities:

- User names are designed to be meaningful. There are permitted to be long, can contain punctuation characters such as space, and can contain accented characters for representing non-English names. Thus it is possible to encode my name as “Michael Roe”.
- There are several forms of user name, depending on which information is included. The “organisational” form we used contains the name of the country, the name of the user’s organisation, the name of the user’s department (if the organisation is divided into departments) and the user’s real name.

  The “organisation” part of a user name implies that the user is in some way affiliated with that organisation. It is not just extra information included to make sure the name is unique.
- Users are authenticated by means of a cryptographic key. Keys may either be stored on a “smart card” to which only the user has access, or stored on a multi-user computer system and protected by just the host’s access control mechanisms. The latter option was much more popular in the pilot project, as it is considerably cheaper. Smart cards are becoming much cheaper, so they may become more widely used.
- The cryptographic key must be associated with the user name, so that people will know that a message digitally signed with that key came from a particular user. This association is conveyed by means of a cryptographically protected data structure, an X.509 “certificate”.

Certificates and Organisational Certification Authorities

A certificate is effectively a signed mail message saying “This key corresponds to this user”, except that it’s ASN.1 encoded and can be processed automatically. There is nothing to stop anybody creating such messages, signed with their own key. If all certificates were treated equally, an attacker could sign a certificate containing whatever name they desired, and hence break the security of the entire system. For this sort of system to work at all, certificates must only be accepted if they are from an “appropriate” person.

One implementation approach would be to just display the name of the certificate issuer to the recipient of a message, and leave them to work out whether the issuer is “appropriate” or not. This is extremely flexible, but unacceptable to the typical end-user; it is not fun to have to carefully check the information in the header fields of every message you receive.

We can simply the task of the message recipient by making absolute rules, which are fixed for everyone running our software. Then, part of the “is this legitimate” check can be performed automatically by the software. The disadvantage is considerable loss of flexibility. Everyone is stuck with these rules (because they’re enforced automatically) even if the person using the software disagrees with them, or they’re totally inappropriate for the environment in which the software is being used.

Which check to automate is a hard design choice to make, and every possibility suffers from some disadvantage. With hindsight, I think we made some less than optimal choices in this area. What we did was as follows:

For each organisation, certificates are issued by the “organisational certification authority” for that organisation. This is a person, or department, within the organisation that has the task of initially identifying members of that organisation (by photograph or whatever) and issuing certificates for their keys. Each user runs the software to generate their own private key, then fills in the “Issue me with a certificate for this key” form and takes it to the person in their organisation responsible for issuing certificates. To save retyping information, the data is also sent by e-mail. However, the paper form must be checked to protect against an attacker forging an e-mail request for a certificate. (The request for a certificate cannot possibly be sent using secure e-mail, as you can’t use secure e-mail until the certificate has been issued!)

So that everyone can tell that a certificate has been issued by the right organisational certification authority, we made the rule that the name of the organisational certification authority must be the *same* as the name of the organisation. (This rule is also in the Internet Privacy Enhanced Mail RCS). This was where we made a mistake. This rule has the great advantage that it’s easy to explain, and it’s easy to implement in software. It’s also far too inflexible, and causes serious problems in practise.

Suppose that two organisations (which trust each other a great deal) wish to share the same organisational certifying authority, to save on equipment and personnel costs. This situation occurred during the PASSWORD pilot, and would appear to be forbidden by the certification authority naming rule. In fact, you can get around it. The same administrator, using the same physical hardware can in effect be two certification authorities, one for each organisation. All the administrator has to do is to change the CA name the software uses to the right name before issuing each certificate. With the right software, this is easy. (With the wrong software, it’s a serious nuisance).

A more fundamental problem with the certification authority naming rule is that it interacts in an unfortunate way with X.500 Directory Service. Each CA needs to store certain information about itself in the X.500 Directory. If the CA has the same name as the organisation, various obscure bits of security control information end up being attributes of the organisation’s entry (because it’s the same as the CA’s entry). Directory administrators may refuse to allow this. If the CA administrator and the X.500 Directory administrator are in fact the same person, a mutual compromise is easily reached. However, they are not always the same person. To make matters worse, with the QUIPU implementation of X.500 the organisation’s entry is usually stored in a DSA that is not controlled by the organisation; it’s usually stored in the master DSA for a country. This increases the likelihood of there being problematic interactions between the people who are trying to set up the service!

Many of our pilot sites ignored the naming rule. For example, we had the following names used:

```
c=GB;o=Cambridge University;ou=Computer Lab;ou=Certification Authority
c=PT;o=Universidade do Minho;ou=Autoridade de Certificacao
```

I’ve chosen these two examples to illustrate the following points:

- I’m as guilty as anyone, because I didn’t obey my own specification on this issue!
- The obvious improvement of the rule is to have certification authorities identified by some fixed string, perhaps “ou=certification authority”. This won’t be acceptable because it is specific to one language (English). A human being can probably guess that “Autoridade de Certificacao” is Portuguese for “Certification Authority” and act accordingly, but it’s hard to automate!

Policy Certification Authorities

So, messages are signed with user's cryptographic keys; you can find how to verify the user's signature by examining the user's certificate; certificates are signed with an organisational CA's key. How do you do you find out how to verify the organisational CA's signature?

To solve this problem, we added another level to the hierarchy: Policy CAs. Policy CAs are there to sign certificates for organisational CAs. They also serve an additional function: giving an indication of how strict the organisational CAs procedures are.

A certificate is really just a statement from an administrator that someone came to them, presented some form of identification, and asked to be issued with a certificate. Thus, a certificate is only as good as the procedural security of its issuer. When deciding whether or not to believe a message, the recipient may take into account their knowledge of how secure the organisational certification authority's procedures are.

Policy CAs are an attempt to present this information to the end-user in a reasonable concise and comprehensible manner. Each Policy CA is supposed to publish a written document describing a set of procedures to be followed by organisational CAs (a security policy). The idea is that each Policy CA will only issue certificates to organisational CAs which have agreed to follow the corresponding security policy, and so the name of the policy CA becomes a convenient shorthand for the policy.

This relies on the assumption that any one user will only have to deal with messages involving a very small number of Policy CAs, and hence is able to remember their names and policies.

In the PASSWORD pilot, there were the following Policy CAs:

```
c=DE;o=GMD;l=Darmstadt;cn=DE-PCA
c=GB;o=University College London;ou=Computer Science;cn=GB-PCA
```

The important features are as follows:

- There are no restrictions on what names can be used for a Policy CA. (Unlike organisational CAs, which must have names of a fixed format). The reason for this is that the protocols never give rise to a situation where it is ambiguous whether an entity is a PCA or not. PCAs are "well known" and there are only a few of them. As a result, PCAs can be named in whatever way is convenient without creating dangerous ambiguities.
- Policy CAs are explicitly permitted to sign certificates for organisational CAs in other countries. In our pilot project, we had approximately one Policy CA for each implementation. A certificate issued by one of these Policy CAs implies that the certificate subject is using a particular combination of hardware and software, and is following the operating procedures recommended for use with that software. As it turned out, German sites tended to use software provided by German suppliers, and UK sites tended to use software from UK suppliers, with the result that most organisational CAs had their certificates signed by a Policy CA in the same country. However, we gave all pilot sites a free choice of which software and which Policy CA to use.

  For example, the organisational certification authorities for Cambridge University (in the UK) and Universidad Politecnica de Madrid (in Spain) are both certified by the GMD policy CA (in Germany).
- We do not think it is appropriate for Policy CAs to continue to be run by the software providers. In the next few years, this service should be provided by national network operators (such as JANET in the UK or DFN in Germany), as they are in a position to make authoritative statements about what constitutes "good practise" for physical and procedural security.

Top Level CAs

The PASSWORD architecture [5], Internet Privacy Enhanced Mail, and the NIST Public Key Infrastructure study [4] all define another layer of structure above Policy CAs. Furthermore, each of these three makes radically different and incompatible recommendations. I leave discussion of the relative merits of each proposal to another time. In the PASSWORD pilot, this top level was for all practical purposes irrelevant. It will become important when the user community becomes much, much bigger.

Legal Aspects

Although there is interesting research to be done concerning the civil liabilities of certification authorities [1], this was outside the scope of the PASSWORD project. However, there was one legal aspect we could not avoid studying: we had to be sure that our pilot experiments did not contravene our own countries' criminal law.

Our initial concerns were export regulations. To be absolutely safe on these grounds, we produced three entirely independent implementations of all the applications: one in the UK, one in France and one in Germany. This entirely eliminated the need to transport hardware or software across national boundaries. We specified the applications in considerable detail to ensure compatability, and extensive testing ensured that these implementations were completely interoperable. From the experience gained on this project, I would be more confident of obtaining the necessary permissions for export in future. In addition, the formation of the European Union is supposed to eliminate barriers to trade such as this; it remains to be seen whether the EU makes any difference.

To further complicate matters, the French laws on domestic use of cryptography changed part-way through the pilot project. When we started, there were no applicable restrictions on the domestic use of cryptography in any of the participating countries. The effect of the change was that French users of cryptography for integrity must inform a government department, and French users of cryptography for confidentiality must obtain prior permission. To comply with these regulations, we produced two versions of the French version of the software, one with and one without confidentiality features.

The US "clipper chip" initiative was announced half way through our pilot project. Although the US government is carefully equivocating as to whether non-escrowed cryptography will be prohibited, no legislative change has yet occurred. However, this made no difference as we were unwilling to supply our software to US customers even before the Clipper fiasco; US patent laws had already created an effective ban on the use of public key cryptography within the US. We were able to make an exception for some US government agencies (including NASA) who wanted to use our software; in this case the patent licencing and export issues could be resolved.

Which protocol to use?

There are several protocols which can be used for secure e-mail, of which the most well-known are:

- Internet Privacy Enhanced Mail (PEM)
- X.400(88)
- 'Pretty Good Privacy" (PGP)

Our pilot project was based upon X.400(88) and PEM. The original proposal for the project was heavily biased in favour of X.400(88), as we believed that our sponsors were in favour of OSI protocols. We offered both protocols on an equal basis to pilot sites, but it turned out that PEM was very much more popular. The reasons for this were as follows:

- X.400 mail systems are extremely complex, extremely expensive, and extremely hard to manage. Some products improve on one of these three aspects (e.g. PP is free) but rarely all three.
- PEM messages can be carried by an X.400 mail system, but the reverse is not true; X.400(88) secured messages cannot be carried using the underlying Internet mail protocol, SMTP.
- There is no proper X.400(88) infrastructure in Europe. If you haven't even got X.400 messaging without security, it's very hard to add security to it! At the start of the PASSWORD project, it was widely hoped that there would be a large-scale X.400(88) pilot in Europe; we would provide security on top of this basic communications infrastructure. This didn't happen, and the only X.400(88) links were the ones we set up ourselves, especially for carrying encrypted messages.
- With communications software, it is highly desirable to be using the same protocol as everyone else, in order to be able to talk to them. When one protocol becomes dominant, everyone switches to it. By the end of the PASSWORD project *no-one* was using secured X.400(88). Those sites which were initially in favour of X.400(88) gave up, and switched to PEM.

I therefore conclude that X.400(88) is now completely dead, with the possible exception of some closed user groups with large budgets and peculiar restrictions on procurement.

The real contender to PEM is now PGP. As I have described in this paper, the main obstacles to deploying a PEM service were the establishment of the certification authority infrastructure, and the legal issues. PGP has rapidly risen to prominence by ignoring both issues. While this has undoubtedly aided its initial deployment, it is likely to cause problems for PGP in the long term. Failure to take adequate care over the legal problems resulted in the author of PGP, Phil Zimmerman, having to appear before a Federal Grand Jury investigation. While he has not been charged with any offence, much less found guilty, this is just the sort of inconvenience we were careful to avoid.

We still believe the the sort of highly structured certification authority system we used in PASSWORD is essential for communication within very large communities. The less structured alternative of PGP is much easier to set up and works well with small groups, but is difficult to manage when the user community becomes too large. I think both PEM and PGP have different roles to play, and they both be with us for a long time to come.

References

[1] M. S. Baum. *Federal Certification Authority Liability and Policy Issues.* Independent Monitoring, Cambridge, MA, July 1993.

[2] International Standards Organization. *ISO/IEC JTC1/SC21 N5532 : Information Technology — Security Frameworks for Open Systems — Part 1: Overview*, 1991.

[3] S. Kent and J. Linn. *RFC 1114: Privacy Enhancement for Internet Electronic Mail: Part II: Certificate Based Key Management.* IAB Privacy Task Force, 1989.

[4] National Institute of Standards and Technology. *Public Key Infrastructure Study*, November 1993.

[5] Michael Roe. *PASSWORD R2.5: Certification Authority Requirements.* Cambridge University Computer Laboratory, December 1993.