Key Distribution in PGP

Mayuri Karra

Abstract—
Pretty Good Privacy (PGP) [1] is a hybrid cryptosystemn that combines the advantages of both symmetric and public key cyrptography. It is used to protect the privacy of e-mail communication and files stored on disk. The key features of PGP include generating message digests, digital signatures, management of personal key rings and disitributable public key certificates. One of the main distinguishing features of PGP from other mechanisms such as X.509 based Public Key Infrastructure (PKI) is the concept of key distribution and management. Unlike these PKIs, where there is a rigid hierarchy of Certifiction Authorities (CAs) for securely associating a key with user (or email address), PGP proposed the concept of Web of Trust where the certification can be done by end users thereby avoiding the need for a single certification authority. While this decentralized trust architecture [2], [3] has its advantages, it comes with its own set of problems [4]. In this report, we will be exploring the distributed trust architecture of PGP, discuss its pros and cons in comparison with a PKI.

I. Pretty Good Privacy

Pretty Good Privacy (PGP) revolutionarized the field of cryptograhy as it was the first to make cryptography accessible to wide mass of on-line users. PGP [1] was created primarily for signing, encrypting and decrypting emails thereby increasing the security of e-mail communications.

A. How does it work?

In PGP, each user has a publicly known encryption key and a private key known only to that user. As depicted in Figure 1, A sender encrypts a message using his public key. The receiver after receiving it decrypts it using his private key (see Figure I-A). Since encrypting an entire message can be time-consuming, PGP uses a faster encryption algorithm(symmetric) to encrypt the message and then uses the public key to encrypt the shorter key that was used to encrypt the entire message. Both the encrypted message and the short key are sent to the receiver who first uses the receiver's private key to decrypt the short key and then uses that key to decrypt the message.

PGP encryption uses a serial combination of hashing, data compression, symmetric-key cryptography, and, finally, public-key cryptography; each step uses one of several supported algorithms. It comes in two public key versions - Rivest-Shamir-Adleman (RSA) and Diffie-Hellman[5]. The RSA version, for which PGP must pay a license fee to RSA, uses the IDEA algorithm to generate a short key for the entire message and RSA to encrypt the short key. The Diffie-Hellman version uses the CAST algorithm for the short key to encrypt the message and the Diffie-Hellman algorithm to encrypt the short key.

Authors are with the Department of Computer Science, University of California, Santa Barbara, CA 93106. E-mail: {mayuri}@cs.ucsb.edu

This work is supported by University of California Santa Barbara.

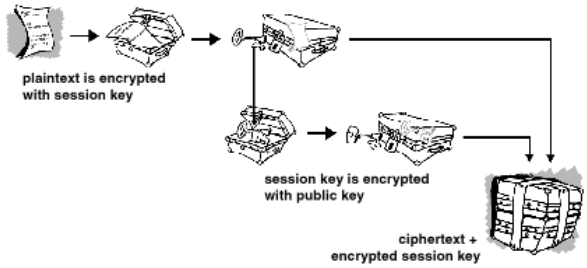

Fig. 1. How PGP encryption works

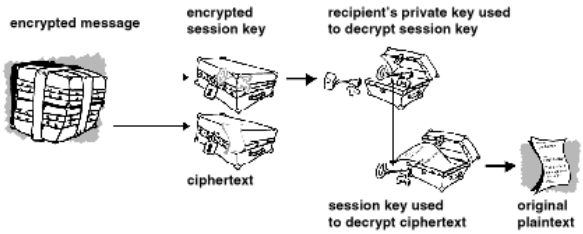

Fig. 2. How PGP decryption works

B. Related technologies and difference

Privacy-Enhanced Mail (PEM) is an Internet standard that provides for secure exchange of electronic mail. Just like PGP, PEM employs a range of cryptographic techniques to allow for confidentiality, sender authentication, and message integrity. The message integrity aspects allow the user to ensure that a message hasn't been modified during transport from the sender. The sender authentication allows a user to verify that the PEM message that they have received is truly from the person who claims to have sent it. The confidentiality feature allows a message to be kept secret from people to whom the message was not addressed.

Although PEM became an IETF proposed standard it was never widely deployed or used. One reason for the lack of deployment was that the PEM protocol depended on prior deployment of a hierarchical public key infrastructure (PKI) with a single root. This infrastructure is based on the another standard knowns as the X.509 standard. Deployment of such a PKI proved impossible as the operational cost and legal liability of the root and 'policy' CAs became understood. In addition to being an obstacle to deployment, the single rooted hierarchy was rejected by many as an unacceptable imposition of central authority.

This led to proposal of the Web of Trust as the PKI infrastructure for PGP. Unlike the X.509 based PKI, where there is a rigid hierarchy of Certifiction Authorities (CAs)

for securely associating a key with user (or email address), PGP proposed the concept of Web of Trust where the certification can be done by end users thereby avoiding the need for a single certification authority. While this decentralization is attractive, it comes with trust issues related to key distribution and management. Unlike their counterparts where the authentication of a key certificate is done by CAs and the trust is absolute, the lack of a fixed certification path in PGP and the uncertainity involved with trusting a PGP certificate is a significant issue. In this report, we will concentrate on several aspects of this problem.

II. Web of Trust

As discussed in the previous Section, the main difference between PGP and its competitors is the lack of a trusted central authority. Every user is given the choice of chosing its own set of trusted users. That is, instead of a central authority whom every body trusts, users certify each other's keys and build a Web of individual public keys where each edge in this webbed network is a signature. The set of all the keys which a key owner knows is referred to as a Key Ring.

As in any public key cryptography, each user who wants to use PGP is assigned public-private key combination. Any one who want to communicate with an user need to know the information such as the user's ID (e.g. email), its publc key and needs to know that this user ID, public key combincation needs to be trusted. This information is stored in a public key certificate. Public key certificates are central to PGP. A public key certificate consists of the following information:

- key owner user ID
- public key
- digital signature of the owner
- certificate's validity period
- list of introducers' signatures

The reason for an owner signing its own key is that it prevents key tampering. Once the key has been signed the next step is to get at least one other person (or key holder) to sign the owner's key. This extra signature provides a link to other keys. If you look at a PGP key, you will see that it has signatures from key(s) other than the owner. Anyone who receives a key in person from the owner is asked to sign the owner's key. There are therefore keys radiating out from the owner, each with possibly different signatures, but signatures of use for the path followed. These signatures are the replcaement for the centralized Certification Authority. Now that we described the basic certificate format, we will use an example to show how PGP's propagation of trust works.

A. An example

Every key owner has an immediate set of friends whom it knows in person. Say, Alice is a friend of Bob and Carol. Alice will sign Bob and Carol's keys and they will sign Alice's key as they trust each other. Any one who receivers Alice's certificate will realize that Bob and Carol attest to the verity of the certificate.

Say, Alice receives a key for a third party called Fred. Alice has not received Fred's key in person, therefore can't be sure with any degree of certainty that the key actually belongs to Fred. Although Fred is not a friend of Alice, he is a friend of Bob. Therefore, Bob signs Fred's certificate. Alice then looks into the list of signers for Fred and finds that they have a mutual friend Bob. Then Alice decides on wether to trust Fred or not based on if Alice trusts Bob to sign only for reliable people. That is, although Alice knows Bob as a friend, she still needs to trust Bob's ability to introduce a new certificate. Basically, PGP works the best if an owner never signs a key unless it is absolutely certain that the key belongs to its claimed owner. If in doubt do not sign as other people will be relying upon your judgement. In the rest of the section, we will discuss the topics of trust levels and evaluation of these trust levels.

B. Degrees of Trust

Every PGP user is given the option of associating a trust level with each of the keys in its Key Ring. In other words, this trust level describes confidence associated with the binding between the user ID and the public-key itself, both contained in the certificate. There are various degrees of confidence attached to a certificate's validity. These are categorised roughly in PGP as follows:

- undefined: we cannot say whether this public key is valid or not.
- marginal: this public key may be valid be we cannot be too sure.
- complete: we can be wholly confident that this public key is valid.

Another important topic is how much we can trust a public-key (ie. indirectly referring to the owner of the public-key) to be a competent signer of another PGP public-key certificate. PGP allows the user to assign four levels of trustworthiness to a public-key. These levels correspond to how much the user thinks the owner of this publickey can be trusted to be an 'introducer' to another trustworthy public-key certificate. Trust levels can be one of these:

- full: this public-key is fully trusted to introduce another public-key.
- marginal: this public-key can be trusted to introduce another public-key, but, it is uncertain whether it is fully competent to do that.
- untrustworthy: this public-key should not be trusted to introduce another, therefore anyr occurrence of this key as a signature on another public-key should be ignored.
- don't know: there are no expressions of trust made about this public-key.

The actual meaning of these trust levels are not explicit. It is only prudent to use them as rough guidelines to how much trust to place in an introducer. How the user arrives at her opinion about the introducer's trustworthiness is also left up to the user. However, guidelines on vetting a user's introducer trustworthiness is given in the documents

accompanying PGP, including various aspects of the candidate which may affect her credibility as an introducer.

When a user places trust in an introducer, it implicitly means that the user places a certain amount of confidence in the introducer's capability to introduce valid certificates. In other words, the user trusts the introducer with respect to introducing correct bindings between a user and her public-key. A marginally trusted introducer may not be as trustworthy as a completely trusted one, therefore more marginally trusted introducers are required to sign a certificate compared to fully a trusted one, for the same level of confidence to be placed in the validity of a certificate. How much trust is placed in any one particular introducer is up to the individual user and is kept secret.

C. Evaluation of trust

PGP allows the automatic evaluation of the trust level associated with a public-key's validity. This is achieved by allowing its users to tune two skepticism parameters: completes_needed and marginals_needed. The former defines the number of completely trusted signatures required to make a certificate completely valid, and the latter defines the number of marginally trusted signatures to achieve the same outcome. A certificate becomes completely valid if either one of these skepticism parameters are met. If neither is met, but at least one type (marginal or complete) of signature is present then the signed certificate attains a marginal validity status. Since PGP does not explicitly provide mechanisms for expressing security policies, this skepticism mechanism is the closest thing to a policy in PGP [2]. The skepticism level of PGP indirectly reflects the user's own policy regarding the threshold of her confidence in PGP signatures. We now look at an example. Let us assumer user Carol fully trusts Alice and Greg as introducers. Further, Carol receives a certificate from Bob. If Carol had defined in her PGP environment that at least two fully trustworthy introducers are required to make a certificate completely valid, then Bob's certificate will only be partially valid if only Alice or Greg had signed it. But since both Alice and Greg trusts the validity of Bob's certificate by signing it, Carol can now regard Bob's certificate as completely valid.

Introducer trusts are manually assigned by each user to the public keys, and exists only within each individual user's public-key ring. Publickey validity is also a secret piece of information because it is based on introducer trust (except direct trust). The rationale for this is twofold; firstly to protect each PGP user's personal opinion about other people's trustworthiness, and secondly different people will have potentially different personal opinions about other people's trustworthiness as an introducer. Therefore, introducer trust levels are secret to the user who assigned them.

D. Revocation of Trust

Another critical topic is the subject of how long a certificate can be trusted. It is unsafe to simply assume that a certificate is valid forever. In PGP, certificates are created with a scheduled validity period: a start date/time and an expiration date/ time. The certificate is expected to be usable for its entire validity period (its lifetime). When the certificate expires, it will no longer be valid, as the authenticity of its key/identification pair are no longer assured. (The certificate can still be safely used to reconfirm information that was encrypted or signed within the validity period it should not be trusted for cryptographic tasks moving forward, however.)

There are also situations where it is necessary to invalidate a certificate prior to its expiration date, such as when an the certificate holder terminates employment with the company or suspects that the certificate's corresponding private key has been compromised. This is called revocation. A revoked certificate is much more suspect than an expired certificate. Expired certificates are unusable, but do not carry the same threat of compromise as a revoked certificate.

Anyone who has signed a certificate can revoke his or her signature on the certificate (provided he or she uses the same private key that created the signature). A revoked signature indicates that the signer no longer believes the public key and identification information belong together, or that the certificate's public key (or corresponding private key) has been compromised. A revoked signature should carry nearly as much weight as a revoked certificate.

With X.509 certificates, a revoked signature is practically the same as a revoked certificate given that the only signature on the certificate is the one that made it valid in the first place: the signature of the CA. PGP certificates provide the added feature that you can revoke your entire certificate (not just the signatures on it) if you yourself feel that the certificate has been compromised.

Only the certificate's owner (the holder of its corresponding private key) or someone whom the certificate's owner has designated as a revoker can revoke a PGP certificate. (Designating a revoker is a useful practice, as it's often the loss of the passphrase for the certificate's corresponding private key that leads a PGP user to revoke his or her certificate. This is only possible if one has access to the private key.) Only the certificate's issuer can revoke an X.509 certificate.

Communicating that a certificate has been revoked when a certificate is revoked, it is important to make potential users of the certificate aware that it is no longer valid. With PGP certificates, the most common way to communicate that a certificate has been revoked is to post it on a certificate server so others who may wish to communicate with you are warned not to use that public key.

In a PKI environment, communication of revoked certificates is most commonly achieved via a data structure called a Certificate Revocation List (CRL), which is published by the CA. The CRL contains a time-stamped, validated list of all revoked, unexpired certificates in the system. Revoked certificates remain on the list only until they expire, then they are removed from the list. This keeps the list from getting too long.

The CA distributes the CRL to users at some regularly

scheduled interval (and potentially off-cycle, whenever a certificate is revoked). Theoretically, this will prevent users from unwittingly using a compromised certificate. It is possible, though, that there may be a time period between CRLs in which a newly compromised certificate is used.

III. Problems of Web of Trust

While PGP's trust model is unaffected by such things as company failures, it has its own set of problems. In this Section, I will discuss a few of those issues. We will also describe how some of these issues are addressed in the recent years.

A. Social aspects

One of the difficulty with a Web of Trust is that every web of trust without a Certification Authority depends on other users for trust. Those with new certificates (ie, produced in the process of generating a new key pair) will not likely be readily trusted by other users' systems, that is by those they have not personally met, until they find enough endorsements for the new certificate. This is because many other Web of Trust users will have their certificate validation process set to require one or more fully trusted endorsers of an otherwise unknown certificate (or perhaps several partial endorsers) before using the public key in that certificate to prepare messages, believe signatures, etc. Therefore, despite the wide use of PGP, it is possible in practice to be unable to readily find someone (or several people) to endorse a new certificate (eg, by comparing physical identification to key owner information and then digitally signing the new certificate). Users in remote areas or undeveloped ones, for instance, may find other users scarce. And, if the other's certificate is also new (and with no or few endorsements from others), then its signature on any new certificate can offer only marginal benefit toward becoming trusted by still other parties' systems and so able to securely exchange messages with them. Key signing parties are a relatively popular mechanism to resolve this problem of finding other users who can install one's certificate in existing webs of trust by endorsing it. Websites also exist to facilitate the location of other OpenPGP users to arrange keysignings. The Gossamer Spider Web of Trust (GSWoT) [6] also makes key verification easier by linking PGP users via a hierarchical style web of trust, where end users can benefit by coincidental or determined trust of someone who is endorsed as an introducer, or by explicitly trusting GSWoT's top-level key minimally as a level 2 introducer (the top-level key endorses level 1 introducers).

B. Loss of private keys

Users, whether individuals or organizations, who lose track of a private key can no longer decrypt messages sent to them produced using the matching public key found in an PGP certificate. Early PGP certificates did not include expiry dates, and those certificates had unlimited life. Users had to prepare a signed cancellation certificate against the time when the matching private key was lost or compromised. Later PGP certificates include expiry dates which automatically preclude such troubles (eventually) when used sensibly. This problem is also easily avoided by the use of "designated revokers". A key owner may designate a third party that has permission to revoke the key owner's key if the key owner loses his own private key and thus loses the ability to revoke his own public key.

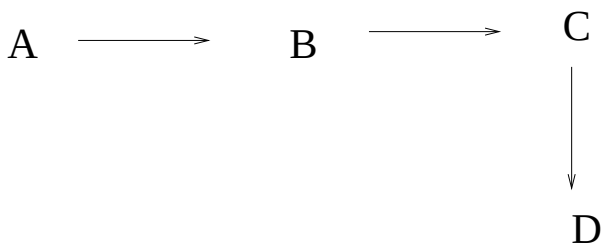

Fig. 3. Introducer chains in PGP

C. Chains of certificates

The possibility of finding chains of certificates is often justified by the "small world phenomenon": given two individuals, it is often possible to find a short chain of people between them such that each person in the chain knows the preceding and following links. However, such a chain is not necessarily useful: the person encrypting an email or verifying a signature not only has to find a chain of signatures from his private key to his correspondent's, but also to trust each person of the chain to be honest and competent about signing keys (that is, he has to judge whether these people are likely to honestly follow the guidelines about verifying the identity of people before signing keys). This is a much stronger constraint. PGP attempts to solve this problem by having the restriction that, no matter how long the chains are, the introducer of a new certificate should be trusted introducer. For example, say, user A trusts user B as an introducer and user B trusts C as an introducer and say C introduces D. As shown in Figure 3, there is a chain of trust from A to D. But as long as user A trusts C as an introducer, D's certificate can be trusted. Otherwise, it will not be validated.

There is however a CERT_DEPTH parameter in PGP which defines the maximum certification chain length, but it is unsure how this is used in evaluating certificate validity. It is suspected that all introducers in the certification chain must be directly trusted by the user. If this is true, then any other introducer, except the last introducer furthest away from the user, in the chain is redundant. Thus, this also makes the CERT_DEPTH parameter redundant in PGP. Further clarification from the creators of PGP on the use of this parameter is required.

D. Revocation of signatures and certificates

Revocation is one of the great weaknesses of public-key cryptography system like PGP. Before we use a public key, we must validate the key's certicate by checking the current Certicate Revocation List to see if the public key is still active(see Section II-D). It might seem that a user needs

to check his cached certicates only when he acquires them, but this is not true. It is a simple matter for a virus to corrupt a certicate cache, and a certicate may be revoked just before use, or even just after it enters the cache.

As mentioned in SectionII-D, revocation can happen in 2 ways: (a) revocation of a signature: A certificate's signer hence revokes its signature from the certificate. (b) revocation of certificate: A user decides that his certificate is not longer usable or valid and hence revokes it. In both cases, this revocation information needs to be sent to all the users whose public key ring contains the certificate. In a X.509 PKI infrastructure where there is a concept of a central authority, the CRL information can be stored in a central directory or exposed as a service. However, in the case of PGP which lacks a central authority, this is a bigger problem. Existing solutions suggest using a central directory for storing the CRLs. Although this solution is a feasible approach, it goes against the very basis of PGP as a decentralized solution.

Further, this solution results in problems of scalability. The CRL of the U.S. Department of Defense (DoD) approached 40MB in 2005 [7]. Imagine the required bandwidth for each DoD client to download 40MB only for certificate revocation every day. The problem exacerbates if some users also have to download external CRLs in addition to external partners having to download the DoD CRL. This enormous task also becomes meaningless when taking into account that as much as 99irrelevant to users because it concerns revoked certificates with which they have no contact. The public key infrastructure (PKI) community has tried to solve this problem The public key infrastructure (PKI) community has tried to solve this problem with two technical approaches: delta CRLs and partitioned CRLs. Delta CRLs allow the download of updates for existing CRLs rather than the entire list. Partitioned CRLs separate CRLs into sizeable chunks. Implementations of these proposals are so complex and fault-prone, that they have not been widely adopted.

D.1 A few proposals

Does adding signature expiration help? Adding expiration to certificates helps in automatic disabling of a certificate with out any propagation of the expiration information. We believe that if we add expiration to signatures associated with a certificate, then time-based revocation will not require any propagation thereby reducing the overall cost of the PGP system.

What happens to certificates signed by a revoked certificate? Although all documents talk about validation of number of signatures in a certificate as enough (in addition to getting the minimum number of signatures), they do not talk about what happens when one of the signers's certificate is revoked. What happens to all the certificates that were signed by this certificate? What happens if the removal of this signature from the other certificate's list result in the invalidation of the certificate?

IV. Conclusion

In this document, we described the trust model of PGP. The decentralized trust model of PGP is very attractive because it is resistant to company failures. At the same time, it increases the number of problems already existant in the X.509 PKI infrastructure as the key distribution information is stored at the end user. The more this information is distributed, the tougher it becomes to manage. We present several of these issues in detail and discuss how some of these issues are adressed or still unsolved.

References

[1] P. R. Zimmermann, *The official PGP user's guide*. Cambridge, MA, USA: MIT Press, 1995.
[2] M. Blaze, J. Feigenbaum, and J. Lacy, "Decentralized trust management," in *SP '96: Proceedings of the 1996 IEEE Symposium on Security and Privacy*, (Washington, DC, USA), p. 164, IEEE Computer Society, 1996.
[3] "The openpgp alliance," 2010.
[4] D. Davis and D. Davis, "Compliance defects in public-key cryptography," in *In Proceedings of the 6th USENIX Security Symposium*, pp. 171–178, 1996.
[5] P. R. Zimmermann, "Cryptography for the internet," in *Scientific American*, 1998.
[6] "The gossammer spider web of trust." Website, 2010. `http://www.gswot.org/`.
[7] R. Nielsen, "Observations from the deployment of a large scale pki," in *4th Annual PKI RD Workshop*, 2005.