

Mentor: Dr.-Ing. S. Rupp

Public Key Infrastructures – PGP vs. X.509

Natasa Prohic

*Institute of Communication Networks and Computer Engineering
University of Stuttgart*

ABSTRACT

Reliance on electronic communications makes information more vulnerable and users require confidentiality, message integrity, sender authentication and sender non-repudiation. Public key cryptography provides these services. The goal of a Public Key Infrastructure (PKI) is to enable secure, convenient and efficient discovery of public keys. There are various types of PKI that are deployed. They differ in the certificate format, trust rules and configuration. In this article systems with X.509 and PGP certificates are described, with their advantages and disadvantages. The goal of this analysis is to highlight the differences between both systems and to provide the reasons for their usage.

Keywords: asymmetric encryption, symmetric encryption, public key, private key, public key infrastructure, digital signature, digital certificate, X.509, PGP, network of trust, key revocation

1. INTRODUCTION

The growth of computer networks and the opening that their interconnection brings, especially through Internet, mean that a great amount of information is traveling through network and crossing numerous intermediate systems. This results in the increase of the number of possible attacks and illegal operations. In order to protect networks from these dangers it is necessary to equip them with security services. They should guarantee the identity of the communicating parties, the prevention that communicating parties won't deny the transmitted message, the protection against unauthorized writing and, in some cases, unauthorized reading of transferred data. These services of authentication, non-repudiation, integrity and confidentiality, respectively, can be provided using cryptosystems.

In the chapter 2 the basic principles of cryptosystems are given, including the symmetric and asymmetric types of encryption, the principles of private and public keys, digital signature, digital certificate and network of trust. Chapter 3 explains the certificate format of X.509, its trust model and key revocation procedure. Also it contains the example of X.509 usage: Secure Sockets Layer (SSL). Chapter 4 discusses the PGP (Pretty Good Privacy), with its type of certificate and web of trust. One of projects that deploy PGP service is presented. In the last chapter the differences between X.509 and PGP are pointed out.

2. BASIC PRINCIPLES

Cryptosystem consists of cryptographic algorithm, keys and protocols that make it work. A cryptographic algorithm is a mathematical function used for the encryption and decryption process. This algorithm works in combination with a key, which can be a word, number or phrase, used for encryption of plaintext into ciphertext. The size of keys is measured in bits. The bigger the key, the more secure the ciphertext. There are two types of encryption: symmetric and asymmetric. In symmetric (also called secret-key encryption) one key is used for encryption and decryption. This key is shared secret between communicating parties. Usually the size of key is up to 128 bits. One example of this type is Data Encryption Standard (DES). It is very fast, but it can be very expensive due to difficulty of secure key distribution. This problem is solved by asymmetric encryption (also called public-key encryption). It uses the pair of keys, one for encryption and one for decryption. One is published and called the public key; the other is kept secret and called the private key. It is computationally infeasible to deduce the private key from the public key. The size of keys is up to 1024 bits. Some examples of this type are: RSA, Elgamal,

Diffie-Hellman (DH), and Digital Signature Algorithm (DSA) [1]. Public-key encryption systems can be used in two ways: for encryption and for digital signatures.

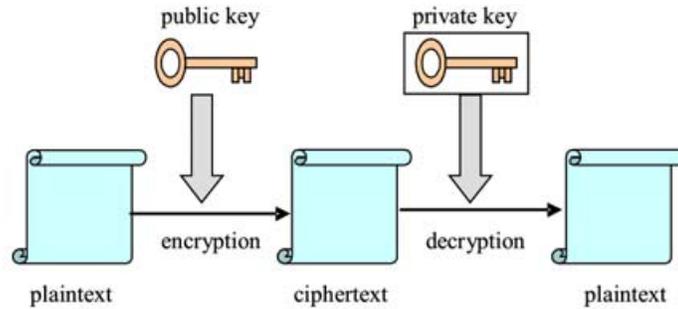


Figure 1: Asymmetric encryption

When used for encryption, the data to be sent to another party is encrypted by using the public key of the recipient and then transmitted to that party. When another party receives the data, she uses her own private key to decrypt it, as it is shown in Figure 1. If the secrecy of the private key is preserved, the confidentiality of the data is guaranteed. When used for digital signing, a one-way function is used to produce a non-reversible hash of the message that has fixed length and is much shorter than the message. The result of the transformation is called a message digest. It must be computationally infeasible to construct a message that has the same message digest. Then the message digest is encrypted using the signer's private key. The digest plus information about the hashing and encryption algorithm compose the signature. The message and signature are logically concatenated and sent together. As illustrated in Figure 2 the receiver applies on the original message the same hash function that the signer used. She uses the sender's public key to decrypt the encrypted data. This is compared to the resulting message digest. If the content of the message is unchanged and the public key is from the same key pair as the private key, the signature is verified correctly. A digital signature provides integrity, authentication and non-repudiation.

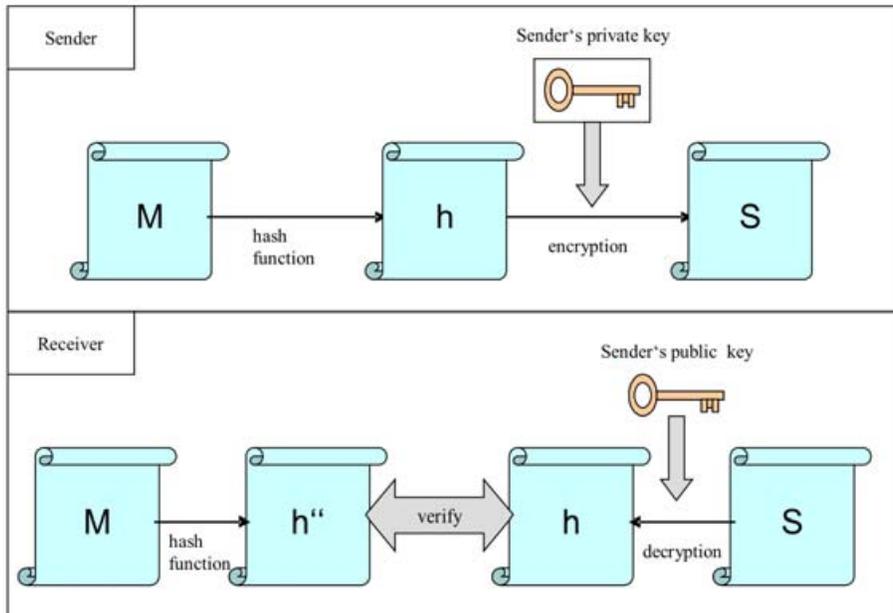


Figure 2: Digital signature

Let us now consider the situation that Alice wants to send a private message to Bob. She computes the ciphertext using Bob's public key. If Eve could trick Alice into using her (Eve's) public key as Bob's public key, Eve would be able to decrypt the message. Thus, Alice has to be confident that she has the

correct public key of Bob. Similarly, Eve could masquerade as Bob, sign a message with her private key, claim the message is from Bob and trick Alice in using her (Eve's) public key as Bob's public key for signature verification. The signature could be verified and Alice would accept a message from Eve as one from Bob. In short, the security of public key cryptography scheme depends on the integrity of public keys and their bindings to proper individuals. The Internet community and ITU have developed the concept of a public key certificate. The fundamental idea behind a public key certificate is the digital signature on the binding of an entity and its public key by a Trusted Third Party (TTP). The certificate basically has three parts: public key, information about the user of public key (such as name, user ID, e-mail, and so on) and one or more digital signatures which state that certificate information has been attested by TTP.

If the receiver has the sender's public key, she can verify the signature and prove that the key is genuine. If she doesn't have the public key, she gets the sender's certificate from some place (e.g. sender or a directory server). The receiver can verify the sender's certificate if she has the public key of the sender certificate issuer. If she doesn't have this key, she obtains her certificate. The receiver can repeat this process until she has the public key of the party she trusts. Now she can unwind the chain by verifying the signatures on the certificates in the reverse order of their accumulation, until she verifies the sender's certificate. The purpose of the public key infrastructure is to facilitate generation, distribution and administration of public key certificates. PKI enables the basic security services in the variety of systems: SSL and HTTPS for communication and transaction security, S/MIME and PGP for e-mail security, SET for value exchange and so on [2].

3. X.509

X.509 is a widely used standard for defining digital signatures. It is actually an ITU-T Recommendation, which means that it has not been officially defined for standardized use. As a result, companies have been implemented it in different ways. IETF's PKIX (Public Key Infrastructure X.509) has developed a profile based on X.509 for Internet use.

X.509 Version 1 has been available since 1988, is widely deployed, and is the most generic. A certificate contains [3]:

- **Version** – This identifies which version of the X.509 standard applies to this certificate, which affects what information can be specified in it.
- **Serial number** - The entity that created the certificate is responsible for assigning it a serial number to distinguish it from other certificates it issues.
- **Issuer name** - This is the X.500 name of the entity that signed the certificate. This is normally a CA. Using this certificate implies trusting the entity that signed this certificate. In some cases, such as root CA certificates, the issuer signs its own certificate.
- **Signature algorithm identifier** - This identifies the algorithm used by the CA to sign the certificate (e.g., RSA, DSA, etc.)
- **Validity period** - Each certificate is valid only for a limited amount of time. This period is described by a start date and time and an end date and time, and can be as short as a few seconds or almost as long as a century. The validity period chosen depends on a number of factors, such as the strength of the private key used to sign the certificate or the amount one is willing to pay for a certificate. This is the expected period that entities can rely on the public value, if the associated private key has not been compromised.
- **Subject name** - The name of the entity whose public key the certificate identifies. This name uses the X.500 standard, so it is intended to be unique across the Internet. This is the Distinguished Name (DN) of the entity, for example,
 - CN=Natasa Prohic, OU= Institute of Communication Networks and Computer Engineering, O=University of Stuttgart, C=DE(These refer to the subject's Common Name, Organizational Unit, Organization, and Country.)
- **Subject public key information** - This is the public key of the entity being named, together with an algorithm identifier which specifies which public key crypto system this key belongs to and any associated key parameters.
- **Issuer digital signature** - This is the digital signature of the issuer

One way of understanding it is presented in the Figure 3. It can be imagined as paper certificate with the public key bound to it. There is the name and information about the owner of the key on it, plus the name and signature of the person who issued it.

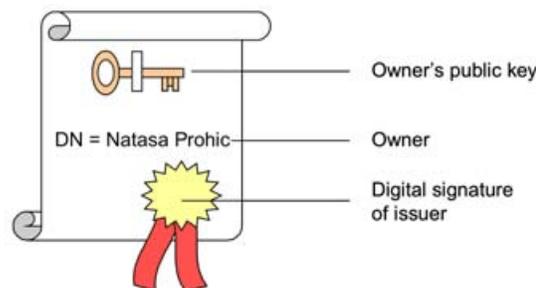


Figure 3: X.509 certificate

Version 2 introduced the concept of subject and issuer unique identifiers to handle the possibility of reuse of subject and/or issuer names over time. Most certificate profile documents strongly recommend that names not to be reused, and that certificates should not make use of unique identifiers. Version 2 certificates are not widely used. Version 3 supports the notion of extensions, whereby anyone can define an extension and include it in the certificate. Some common extensions in use today are: Key Usage (limits the use of the keys to particular purposes such as "signing-only"), Alternative Names (allows other identities to also be associated with this public key, e.g. DNS names, e-mail addresses, IP addresses) and Policy Information (indicates the rules under which the public key may be used). Extensions can be marked critical to indicate that the extension should be checked and enforced/used. For example, if a certificate has the Key Usage extension marked critical and set to "keyCertSign" then if this certificate is presented during SSL communication, it should be rejected, as the certificate extension indicates that the associated private key should only be used for signing certificates and not for SSL use. Version 4 is published in 2000, but still not widely implemented [4].

The main feature of the Public Key Infrastructure with X.509 certificates are Certification Authority (CA) and Registration Authority (RA). CA is the entity that issues digital certificates. It is an example of TTP. The CA attests that the public key contained in the certificate belongs to the person, organization, server, or other entity noted in the certificate. There are many commercial CAs that charge for their services, e.g., VeriSign. Institutions and governments may have their own CAs, and there are free CAs, e.g., CAcert. RA refers to persons, processes and tool used for supporting the registration of users with PKI and administration of users. They vouch for binding between public keys and certificate holder entities. Certificate holders can sign and decrypt digital documents using private keys. Clients validate digital signatures and their certification paths from known public key of trusted CA and encrypt documents using public key from certificates of certificates holders. One important part of PKI is repositories. They store and make available certificates and Certificate Revocation Lists (CRLs).

There are many reasons for revoking a certificate: the user's private key is assumed to be compromised, the user is no longer certified by this CA, the CA's private key is assumed to be compromised. When any of these reasons is determined, the serial number of the certificate is added to the list of the revoked certificates that had been issued by the same CA. Since Version 3 also the reason can be indicated. CRL is digitally signed by CA and is published at some regularly scheduled interval. If the validity date for a revoked certificate has expired, the serial number of that certificate may be removed. CRL contains: the name of the CA, the date and time the list was created, an optional field to indicate the date and time for the next update for the list, list of revoked certificates and the date each was revoked and the digital signature of the CA [5]. During verifying not only the signature must be verified, but also the chain of certificates signed with that signature. In addition the verifier must obtain the most recent version of the CRL for each certificate to determine if the certificate has not been revoked.

In the earlier network of trust model, there were a number of root certificates from which trust extends. These certificates may certify certificates themselves, or they may certify certificates that certify still other certificates down the chain, as shown in the Figure 4. Alice starts out knowing the root key and retrieves all the certificates from the root down to her own key. Alice can authenticate to Bob by sending him those certificates. Since Bob also starts out knowing and trusting the same root key, he has the path to

Alice's key. The certification path is represented bold. There have been extensions to this model for later versions of X.509, where a parent can certify the key of the child (down certificate), a child can certify the key of the parent (up certificate) and any node certifies the key of any other (cross certificate), as shown in the Figure 5. Now Alice starts at the bottom, with her own key. She looks for cross-certificates to an ancestor of Bob (ancestor means the first from down common node of Alice's and Bob's chain of certificates (least common ancestor) or closer in the hierarchy to Bob than least common ancestor). If there are no cross-certificates, Alice follows the parent certificate to go up a level. Then Alice repeats the same procedure until she finds a cross-certificate to an ancestor of Alice and Bob, or reaches the least common ancestor. In either case, at that point Alice starts to go down to Bob [6]. It can be noticed that Alice does not need to trust root CA.

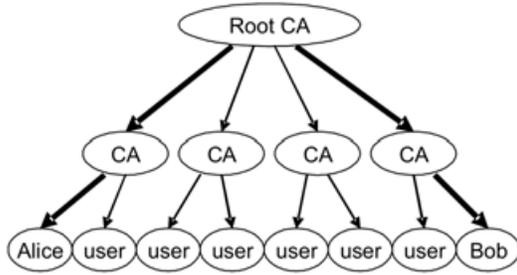


Figure 4: Top-down model

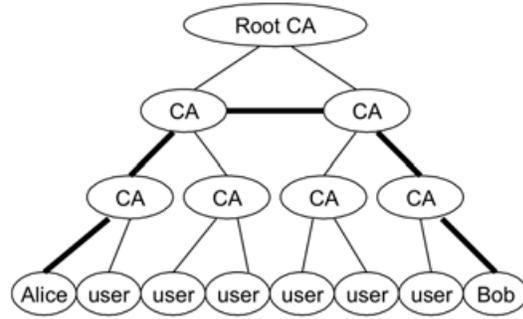


Figure 5: Up-cross-down model

The problem of assuring correctness of match between data and entity when the data are presented to the CA is difficult, which is why commercial CAs often use a combination of authentication techniques including checking government bureaus, the payment infrastructure, third parties databases and services, and custom heuristics. Notaries are required in some cases to personally know the party whose signature is being notarized. For example, at VeriSign for the class of certificates with the highest assurance (Class 3), a person has to show government-issued photographic identification and one other identification credential [7].

One of the examples of X.509 certificate usage is SSL. Developed by Netscape Communications, SSL version 3.0 was released in 1996, which later served as a basis to develop Transport Layer Security (TLS) version 1.0, an IETF standard protocol. They are cryptographic protocols that provide secure communication over Internet. It uses X.509 certificates to verify the identity of endpoints – client and server.

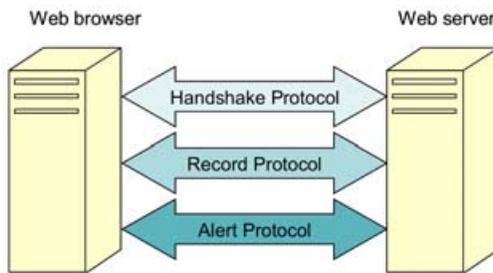


Figure 6: SSL protocol stack

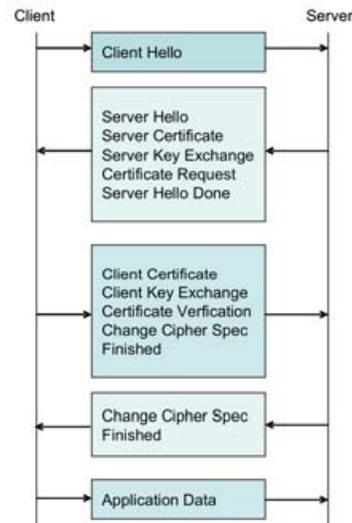


Figure 7: Handshake protocol

In the Figure 6 three protocols are presented as parts of SSL: the Handshake Protocol, the Record Protocol and the Alert Protocol [8]. During the Handshake Protocol the following important steps occur: the encryption algorithms are negotiated and the server is authenticated to the client. The server may optionally ask the client to authenticate itself. Although SSL uses symmetric cryptography for data encryption during the transfer, asymmetric cryptography is used to negotiate the key used for that symmetric encryption. Figure 7 gives the steps of this protocol. When the session is initiated and the handshake is finished, the data transfer is encrypted during the Record Protocol phase. If there are any alarms during the session, the alert is attached to the suspicious packet and handled according to the Alert Protocol. SSL runs on layers beneath application protocols and above the TCP transport protocol. While it can add security to any protocol that uses TCP, it is most commonly used with HTTP to form HTTPS. HTTPS is used to secure World Wide Web pages for applications such as e-commerce. Nowadays there are a lot of frauds like acquiring through deception sensitive personal information such as passwords and credit card details (phishing). Alternative Names on the certificate (e.g., URL) need to be checked in order to prevent them.

4. PGP

PGP is a public key cryptographic package, which is intended for public usage. It provides sender authenticity, message integrity and non-repudiation of the sender. Although PGP can encrypt any data or files, it is most commonly used for e-mail which has no built-in security as originally implemented. It was originally designed and developed by Phil Zimmermann in 1991[9]. For that time it has been sufficiently influential that its algorithms and data formats have been standardized for interoperability between different pieces of software. Eventually, the PGP design was made in Internet standards-track specification known as OpenPGP. It is described in RFC2440 [10]. OpenPGP is now an open standard used by PGP, GNU Privacy Guard (GnuPG), Hushmail and others.

PGP combines symmetric and asymmetric cryptography. The user generates a pair: (public key, private key) that is associated with his unique ID. Public keys are stored on public key rings and private keys are stored on private key rings. On the sender's side, PGP creates a session key, which is random number generated by the keystroke characteristics of a user. Once the data is encrypted with this key, the session key is encrypted with the recipient's public key and sent together with the cipher text to the recipient. The recipient's copy of PGP uses her private key to recover the session key, which then allows the recipient to decrypt the cipher text. PGP uses passphrase to encrypt the private key on its owner's machine. Passphrase is longer and more complicated version of the password. The private key is encrypted on the disc using a hash of the passphrase as a secret key. In order to use her private key, user has to decrypt it using the passphrase. The distribution of public keys is usually done by key servers. They are mirrored at various locations around the world. They possess the recipients' public keys and on the demand of sender, they give the sender the recipients' public key. These computers can be, and are, mostly run by individuals as a pro bono service.

A PGP certificate includes (but is not limited to) the following information [9]:

- **PGP version number**—this identifies which version of PGP was used to create the key associated with the certificate.
- **Certificate holder's public key**—the public portion of the key pair, together with the algorithm of the key: RSA, Elgamal or DSA.
- **Certificate holder's information**—is information about the user, such as his or her name, user ID, e-mail address, ICQ number, photograph, and so on.
- **Digital signature of the certificate owner**—also called a self-signature, is the signature using the corresponding private key of the public key associated with the certificate.
- **Validity period**—the certificate's start date/time and expiration date/time; indicates when the certificate will expire. If the key pair contains subkeys, then this includes the expiration of each of the encryption subkeys as well. Subkeys enable convenient use of separate keys for signing and encryption
- **Preferred symmetric encryption algorithm for the key**—indicates the encryption algorithm to which the certificate owner prefers to have information encrypted. The supported algorithms are CAST, IDEA, Triple-DES, and Blowfish.

PGP certificate can be imagined as a public key with one or more labels tied to it as shown in Figure 8. On these labels there are information identifying the owner of the key and a signature of the key's owner, which states that the key and the identification go together. Each label contains a different means of identifying the key's owner (e.g., the owner's name and corporate e-mail account, the owner's nickname and home e-mail account, a photograph of the owner—all in one certificate). Single certificate can contain multiple signatures. Several or many people may sign the key/identification pair to attest to their own assurance that the public key definitely belongs to the specified owner. The list of signatures of each label may differ. Signatures attest to the authenticity that one of the labels belongs to the public key, not that all the labels on the key are authentic.

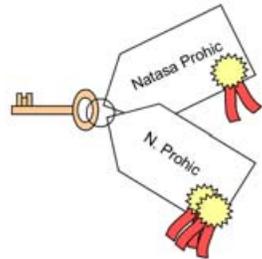


Figure 8: PGP certificate

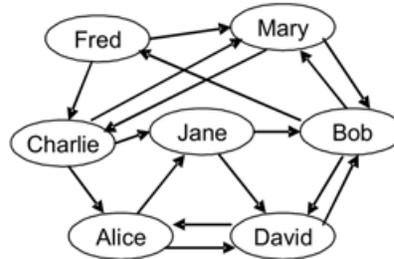


Figure 9: Web of trust

PGP has a certificate vetting scheme to assist with this. It is called web of trust. The signing of certificates is commonly done at key signing parties. A key signing party is an event at which people present their PGP-compatible keys to others in person, who, if they are confident the key actually belongs to the person who claims it, digitally signs the PGP certificate. Participants write down a string of letters and numbers, called a fingerprint, which represents their key. The fingerprint is created by a hash function, which compresses the public key down to a unique string. Participants exchange these fingerprints as they verify each others' identification. Then, after the party, they obtain the public keys corresponding to the fingerprints they received and digitally sign them.

The specification of trust signatures supports the creation of certificate authorities, which are in PGP named trusted introducers [11]. A trust signature indicates both that the key belongs to its claimed owner and that the owner of the key is trustworthy to sign other keys at one level below their own. A level 0 signature is comparable to a web of trust signature, because only the validity of the key is certified. A level 1 signature is similar to the trust one has in a certificate authority because a key signed to level 1 is able to issue an unlimited number of level 0 signatures. A level 2 signature is allows the owner of the key to make other keys certificate authorities. The owner of this key is called meta-introducer. If Alice wants to send message to Bob and doesn't directly trust him, she has to find the person(s) that can confirm that that key belongs to Bob.

There are different tools for doing it: AT&T PathServer, Signature Path Tracing, etc. To be able to verify such paths it is very important that everyone signs the key of others and submits these signatures to the key servers, so others can benefit from such signatures. The paths between two keys have to be as short as possible. Figure 9 depicts an example of a web of trust. There are two paths from Alice to Bob, via Jane and via David, which means that, since Alice trusts Jane and David, she will trust Bob. If the path between Alice's and Bob's gets longer Alice is less sure about the validity authenticity of his key. Paths which do not share a common key between the starting key and the last key are called disjoint paths. It is important to have as many disjoint paths as possible. The more disjoint paths between two keys, the less the probability that someone can fake a confirmation chain by issuing a wrong signature.

PGP also includes a vote counting scheme. User can give any key in his or her key ring a certain trust level. This trust level tells PGP how much you trust key certificates done by this key. The lower the trust value the more key certificates are necessary to validate a key. There are four levels of trust:

- untrusted - Key certificates done with this key are ignored.
- marginal - At least 2 keys with marginal trust have to sign another (third) key to make this third key a valid key.
- complete - At least one key with complete trust has to sign another key to make the key valid.
- ultimate - If user has the secret key for a public key this key is ultimately trusted. Every key user signs with an ultimate trusted key becomes valid.

There are three levels of validity: valid, marginally valid and invalid. The keys which are valid are specially flagged in public key ring. A key is valid if it is signed by the owner of the key ring or if it is signed by enough other key holders, which are trusted by the key owner. A key which is flagged as invalid in the key ring will only be used with reservation by PGP. Key certificates created with such a key are ignored. If a signed message is verified, there will be a warning that you can not be sure that the signing key belongs really to the person mentioned in the key.

User can revoke his or her entire certificate if he or she feels 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 task that is only possible if one has access to the private key and revoker has it. Usually by posting the revoked certificate on a server others who may wish to communicate with a user are warned not to use that public key.

Among the projects which provide PGP services is GNU Privacy Project (GnuPP) [12], funded by the German Ministry of Economics and Technology (BMW). It was developed to provide access for everyone to a secure, trustworthy and easy-to-use encryption system. All software belonging to this package is Free Software under the GNU General Public License (GPL). GnuPP consists of: the crypto-engine GnuPG (GNU Privacy Guard), the graphical user interface GPA (GNU Privacy Assistant) and the generic mail plug-in WinPT (Windows Privacy Tray).

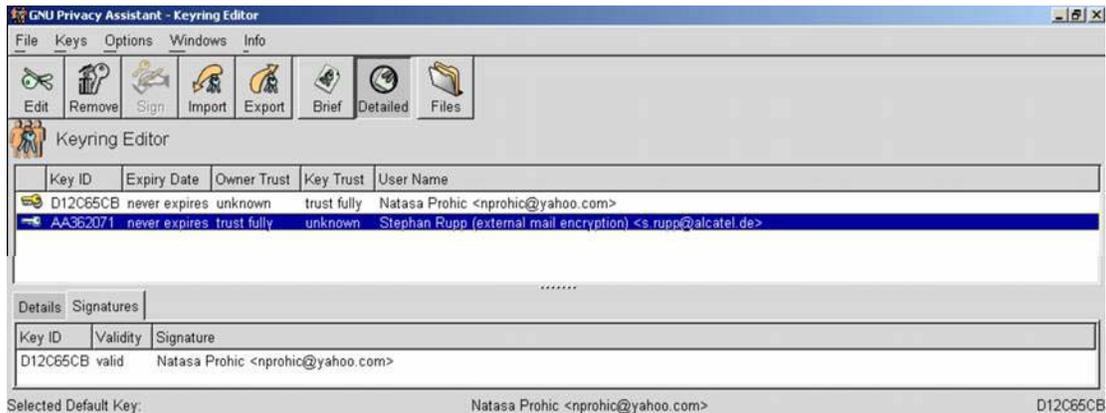


Figure 10: Keyring editor

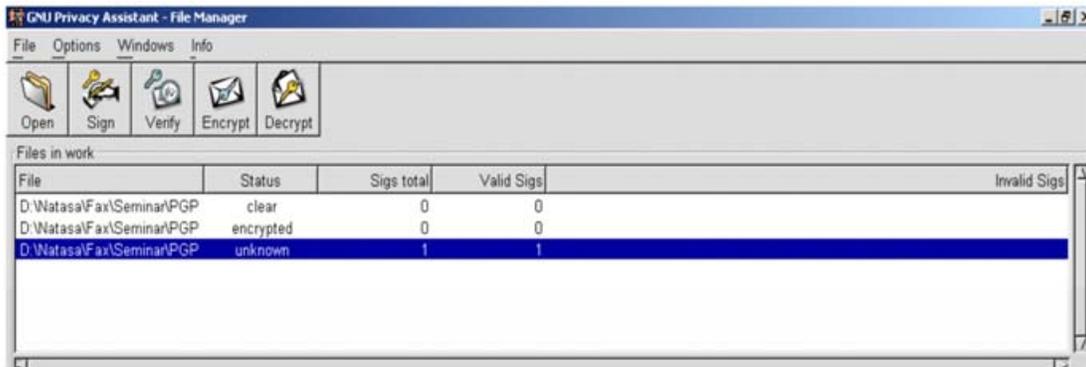


Figure 11: File manager

The graphical user interface, GNU Privacy Assistant (Figure 10), is used to manage the public and the private key pair. When started for the first time, it is necessary to create a key pair comprising the public and the private keys of the user. The public key can be written into a file using the 'Export' button. It can then be sent to user's email contacts. User has to get the public keys of communication partners and insert them into public key ring using the 'Import' button. The private key ring holds one or more keys that user has generated using GPA. They are used to decrypt mail that user receives. The copy of private keys

should always be kept in a safe place. If user loses secret key she will not be able to read messages or files encrypted with the corresponding public key. Using the 'Files' Button you can access the File Manager (Figure 11). Here you can select a file to encrypt with your own public key or the public key of a person to whom you want to send the file. Furthermore you can decrypt files sent to you and sign or verify the signatures of those files.

5. COMPARISON

Given the characteristics of X.509 and PGP described in chapters 3 and 4, respectively, the major differences between PGP and X.509 PKIs can be separated in three areas: differences in certificate, network of trust and revocation procedure.

Differences in certificate: Every PGP certificate contains a self-signature and can contain multiple signatures, while X.509 certificates support only a single digital signature to attest to the key's validity. X.509 certificates natively support only a single name for the key's owner. PGP certificate has public key with several labels which identify the user in different ways.

Differences in network of trust: User can create his or her own PGP certificate, but he or she must request and be issued an X.509 certificate from a Certification Authority. With X.509 certificates, the validator is always a Certification Authority or someone designated by a CA. PGP uses digital signatures as its form of introduction. When any user signs another's key, he or she becomes an introducer of that key. As this process goes on, it establishes a web of trust, so any user can act as a certifying authority. By this, many certification paths are formed to achieve fault tolerance in compensation for the fact that amateur certifiers are signing certificates. Any PGP user can validate another PGP user's public key certificate. However, such a certificate is only valid to another user if the relying party recognizes the validator as a trusted introducer. PGP user is the one that manages keys, while with X.509 CA does a managing of keys. In an organization using a PKI with X.509 certificates, it is the job of the RAs to approve certificate requests and the job of the CAs to issue certificates to users - a process which generally entails responding to a user's request for a certificate. In an organization using PGP certificates, it is the job of the CA to check the authenticity of all PGP certificates and then sign the good ones.

Differences in revocation procedure: 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 user can revoke his or her entire certificate if user feels that the certificate has been compromised. The certificate's owner or revoker can revoke a PGP certificate. Only the certificate's issuer can revoke an X.509 certificate. Communication of revoked X.509 certificates is most commonly achieved via CRL, which is published by the CA. With PGP certificates, the user usually posts the revoked certificate on a certificate server.

X.509 standard does not provide any guidelines on the use of cross-certificates, certification paths and CRLs, so the users of X.509 certificates have to provide these procedures themselves. The result of this is that it takes longer to establish X.509 user communities than PGP user communities.

6. CONCLUSION

The importance of secure key management cannot be overemphasized. The greatest weakness in any cryptosystem is the handling of keys. Through the use of certificates unauthorized access can be prevented in open networks without restricting authorized access. Most of organizations use X.509 certificates because of the RA's verification of the data provided in the certificate request form, while PGP is more popular in the circle of acquaintances. But, due to the difficulty of certificate management for user they don't have widespread use. Perhaps the most potential for widespread use have digitally signed access rights embedded in smart cards. The applications of smart cards include their use as credit or ATM cards, SIMs for mobile phones, authorization cards for pay television, high security identification and access control cards, public transport tickets, electronic wallets, etc. A large growing application is smart ID cards. They are used for authentication of identity. Examples include the US Department of Defense Common Access Card (CAC) and their use by many governments as ID cards for their citizens. When combined with biometrics smart cards can provide two or three factor authentication. Smart cards are a privacy enhancing technology and when used together with appropriate security and privacy policies can form a highly effective identity authentication technology. Both technologies will profit from the smart cards and PGP will try to expand its area of application on organizations.

REFERENCES

- [1] Prof. Dr. K. Rothermel, Lecture notes “Introduction to Distributed Systems”, edition 2004/2005, chapter 9.
- [2] J.Weise, “Public key Infrastructure Overview”, Sun Blueprints Online, August 2001, <http://www.sun.com/blueprints/0801/publickey.pdf>.
- [3]Y. Isobe, Y. Seto, M. Kataoka, “Development of Personal Authentication System Using Fingerprint with Digital Signature Technologies”, Proceedings of the 34th Hawaii International Conference on System Sciences, 2001.
- [4] ITU-T Recommendation X.509, “Information technology – Open Systems Interconnection –The Directory: “Public-key and attribute certificate frameworks”, <http://www.itu.int/ITU-T/publications/recs.html>.
- [5] H.L.Kesterson II, “Digital Signatures – Whom Do You Trust?”, IEEE Electronic Database 0-7803-3741-7/97.
- [6] R. Perlman, “An Overview of PKI Trust Models”, IEEE Network, November/December 1999
- [7] “VeriSign Certification Practice Statement” , version 3.0 , April 1, 2005, http://www.verisign.com/repository/CPS/VeriSignCPSv3_03.15.05.pdf.
- [8] Cisco Systems, “Introduction to Secure Sockets Layer”, White paper from Internet, http://www.cisco.com/en/US/netsol/ns340/ns394/ns50/ns140/networking_solutions_white_paper09186a0080136858.shtml.
- [9] “PGP User’s Guide: An Introduction to Cryptography”, <ftp://ftp.pgpi.org/pub/pgp/7.0/docs/english/IntroToCrypto.pdf>.
- [10] J. Callas, L. Donnerhacker, H. Finney, R. Thayer, “OpenPGP Message Format”, RFC 2440, November 1998, <http://www.ietf.org/rfc/rfc2440.txt>.
- [11] D.W. Chadwick, A. J. Young, N. Kapidzic Cicovic, “Merging and Extending the PGP and PEM Trust Models – The ICE-TEL Trust Model”, IEEE network, May/June 1997.
- [12] “The GNU Privacy Project”, <http://www.gnupg.com/start.html>.

ABBREVIATIONS

CA Certification Authority
CRL Certificate Revocation List
DES Data Encryption Standard
DH Diffie-Hellman
DNS Domain Name System
DSA Digital Signature Algorithm
GnuPG GNU Privacy Guard
HTTP Hypertext Transport Protocol
IDEA International Data Encryption Algorithm
IETF Internet Engineering Task Force
IP Internet Protocol
ITU-T International Telecommunication Union – Telecommunication Standardization Sector
MIME Multipurpose Internet Mail Extensions
PGP Pretty Good Privacy
PKI Public Key Infrastructure
PKIX Public Key Infrastructure X.509
RA Registration Authority
RFC Request For Comments
RSA Rivest-Shamir-Adleman
SET Secure Electronic Transaction
SSL Secure Sockets Layer
TCP Transmission Control Protocol
TTP Trusted Third Party
URL Uniform Resource Locator
X.509 ITU-T Recommendation X.509