Scaling the Web of Trust: Combining Kerberos and PGP to Provide Large Scale Authentication

Jeffrey I. Schiller
Massachusetts Institute of Technology

Derek Atkins
Massachusetts Institute of Technology

Abstract

Internet Security has become more important recently as the Internet grows exponentially and security breaches become more publicized. An important area of concern for many Internet users is the privacy and integrity of their electronic files and messages. Phil Zimmermann's Pretty Good Privacy (PGP) provides a general purpose utility for file and message protection. However PGP requires that communicating users be "introduced" to each other. This paper describes a scheme that permits an enterprise using Kerberos to create an automated introducer called the PGP Key Signer Service. Using this service people in the enterprise who have no common acquaintances to act as introducers can be introduced through the Key Signer.

1. Introduction

The Internet Community has grown from a small collection of Computer Science researchers into the International Information Infrastructure. Yet the security technology currently deployed on the Network reflects its original experimental roots. There are many and varied reasons why things are today the way they are. However, one of the key reasons is because most Internet users are unaware of the inherent lack of security on the net. When most people login using their password they assume that only they may use their computer accounts. Similarly when they send electronic mail they assume that only the marked recipients will see the message. When they receive electronic mail they assume that it originated from the person labeled in the "From" field of the message.

As the Internet has grown, so has the quantity of security problems. Recently these problems have received significant publicity and the public's understanding of the lack of security is growing. With this growth in understanding comes a demand for security.

Several systems now exist to provide security for computers and networks. Two of the more popular security programs are Kerberos [Kerb1, Kerb2] and PGP [PGP]. Kerberos is MIT's real time authentication system, it provides for the security of login sessions and client server transactions. PGP is Phil Zimmermann's "Pretty Good Privacy" public key encryption program. It provides for confidentiality and authentication of electronic mail (e-mail) and other "store and forward" types of transactions as well as encryption of private files.

Each of these systems has its strengths, weaknesses and applicable problem domains. This paper will show how the two can be combined to provide a hybrid security service which neither alone can provide.

In the sections that follow we will provide a brief introduction to PGP and Public Key Cryptography, the underlying technology that makes PGP possible. We will then briefly describe Kerberos. Readers who desire a more thorough background should read the papers referenced above. Finally we will discuss how PGP can make use of Kerberos in a significant way.

1.1 The Problem with Electronic Mail

SMTP, or Simple Mail Transport Protocol is the Internet standard for the delivery of electronic mail between computer systems. A tried and true protocol, it provides message services to millions of Internet users everyday. Yet it provides for no security services at all. Perhaps more to the point, SMTP servers believe that mail originates from where it claims to.

Forging e-mail messages is child's play. On the MIT campus alone we have seen several independently written shell scripts that permit even the most novice of computer users to send forged messages that are indistinguishable form the real McCoy.

An additional threat to e-mail is that very few networks are protected against eavesdroppers. Recently, bulletins form Carnegie Mellon's Computer Emergency Response Team (CERT)[1] have pointed out many incidents of computer crackers “sniffing” passwords from the Internet [CertSniff]. The same technology that provides for the sniffing of passwords can easily read e-mail in flight as well.

Today the only thing protecting e-mail from these prying eyes is the sheer boredom that most e-mail represents! However as e-mail is used increasingly to discuss confidential corporate plans, commercial transactions and even credit card numbers, it will be a target for network eavesdroppers.

Another important feature lacking in e-mail is the ability to prove after the fact that a message was “signed” by its sender. This ability is needed not only for casual proof of origination, but also for formal uses of e-mail such as electronic purchase orders, bid documents and other commercial interactions. Luckily there is a technology to address this requirement. Digital Signatures permit people to electronically sign documents in a fashion that allows anyone to prove that a document originated from whom it claims to have originated from.

Digital Signatures are provided by Public Key Encryption systems such as the RSA [RSA] system. RSA is used by Internet Privacy Enhanced Mail [PEM] and by PGP. Its use in PGP will be discussed here.

2. PGP

PGP was written by Phil Zimmermann in 1991 in reaction to a move by the U.S. Federal Government to require providers of communication services to provide plaintext copies of encrypted information to the government upon its request. PGP was designed to make this impossible by providing a general purpose cryptographic utility which people can use to protect information that they keep for themselves (say in private files) and share with others (for example electronic mail). As such the primary focus of the development of PGP was on privacy.

PGP provides privacy by using a combination of the IDEA [IDEA] symmetric cipher and the RSA public key system. PGP also includes the ability to make digital signatures. This makes PGP a complete electronic mail security tool. It provides confidentiality, authentication of origination and integrity protection for files and messages.

PGP originally had patent difficulties. Because RSA is patented within the United States, users require a license from Public Key Partners or software from RSA Data Security in order to make use of PGP without the threat of legal action being taken against them. In addition, cryptographic systems, be they hardware or software, currently may not be exported from the United States without a license from the State Department Office of Defense Trade Controls.

Version 2 of PGP was coded in Europe and imported into the United States. Starting in May, 1994 the Massachusetts Institute of Technology (MIT) has been distributing a

---

[1] CERT was founded after the Internet Worm Incident of November 1988. Funded by ARPA, it is operated by Carnegie Mellon's Software Engineering Institute.

version of PGP which includes the RSAREF software toolkit licensed from RSA Data Security. This release of PGP, although not exportable from the United States, provides non-commercial users of PGP a legally usable version. The ViaCrypt company[2] sells a commercial version that is also licensed from Public Key Partners. The effect is that PGP is just about globally available.

2.1. Public Key Cryptography and its use in PGP

Before we continue we should talk a little about Public Key Cryptography. Invented in 1976 by Whitfield Diffie and Martin Hellman [DiffieHellman], Public Key Encryption is a kind of cipher system where instead of the traditional single encryption key, two keys are employed. One key is used for encryption while the other is used for decryption. Knowing one key does not imply knowledge of the other.

PGP uses the RSA system, named for its three inventors: Ronald Rivest, Adi Shamir and Leonard Adleman. Each user of PGP generates a key pair. This pair consists of a **public key** and a **private key**. As their names imply, the public key may be made public while the private key is kept secret by its owner. Both the public and private keys are large values (not simple words or small numbers that can be memorized) so PGP stores them in files. To protect the private key, PGP prompts for a pass phrase during key pair generation. This pass phrase is converted into an IDEA key which is then used to encrypt (using the IDEA cipher) the file that contains the user's RSA private key. A prudent user will protect this file against unwanted perusal even though it is also protected by the pass phrase. Some people go to the length of storing their private key only on a removable floppy which they lock up when not in use.

PGP may be used to encipher a private message between two users if they know each others' public keys. A sender can send a message to a recipient by encrypting it in the public key of the recipient. The recipient can then decrypt the message by making use of her private key. This is shown graphically in Figure 1. A private key can also be used to construct an unforgeable digital signature which can be affixed to any message. The sender would use her own private key to construct the digital signature, before encrypting the message, and the recipient uses the sender's public key to verify the integrity of a message, after first decrypting it.

---

[2] ViaCrypt, 9033 North 24th Avenue, Suite 7, Phoenix, Arizona 85021, USA, Phone: (602) 944-0773, viacrypt@acm.org

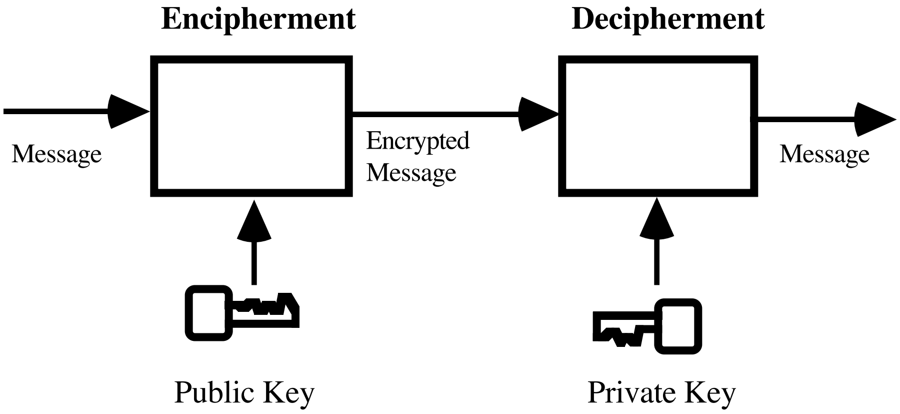


Figure 1: *Message Flow and Encipherment*

Note carefully that all sensitive operations (reading a private message or creating the unforgeable signature) make use of the private keys. Private keys are never exchanged or disclosed. Public keys are never used for these sensitive operations and therefore it is not a problem for them to be exchanged unencrypted (or published in a public place).

To facilitate the exchange of PGP public keys, a network of **key servers** has arisen that make use of software written by Michael Graff of Iowa State University. The key servers are electronic mail based automated responders. To fetch someone's public key a user sends an electronic message in a particular format requesting the key and the responder automatically sends back the requested key via electronic mail. Brian LaMacchia of MIT has added a World Wide Web server interfaceto this network of servers so keys may be retrieved and stored using popular World Wide Web browsing software.[3]

However there is an important issue that needs to be considered. How does someone know that a public key that claims to belong to a particular person in fact does so? After all anyone can generate a key pair and publish the public key along with a claim that the key belongs to, say, the President. There needs to be a way to securely associate a public key with a name.

PGP addresses this problem by the creation of the PGP **Web of Trust**.[4] The Web of Trust starts by having two people each generate and exchange public keys and names. Because this is done in person, and presumably between two people who know each other, they can now communicate with each other using signed messages and know beyond a shadow of a doubt that they are talking with each other. The Web grows from this simple one on one association when one of the participants performs a similar key exchange with a third person. An example will illustrate this best.

Assume that Alice and Bob meet in person and exchange public keys. Now Alice and Toni meet later on and exchange keys. Because Alice securely has Bob's key, Alice can provide a “signed” copy of Bob's key to Toni. Alice signs Bob's key by computing a digital signature on it using PGP (which has built in features for doing exactly this exchange). In this example Alice is acting as an **Introducer** between Bob and Toni. Although they have never met, Alice has introduced them digitally to each other. Figure 2 illustrates this concept.

---

[3] http://www-swiss.ai.mit.edu/~bal/pks-toplev.html

[4] Not to be confused with the World Wide Web

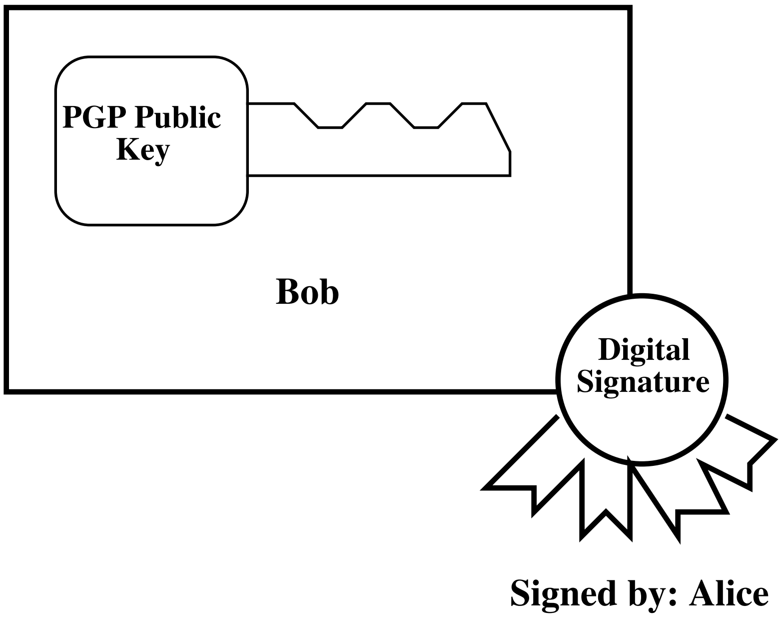


Figure 2: *Graphical Representation of a Signed Key*

PGP permits multiple people to digitally sign an association of a name with a public key. If Bob has another friend, Bill, Bob can ask Bill to sign his key as well. Now if someone gets Bob's key (either from Bob or from a key server) and they trust Bill, they can know that they have Bob's key, even if they never heard of Alice. Figure 3 illustrates this, Bob's key has two signatures on it now, one from Bill and one from Alice.

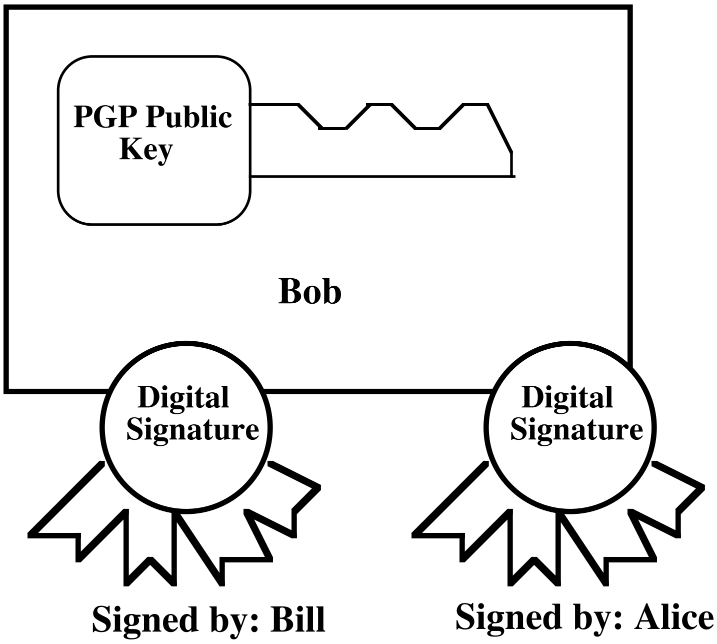
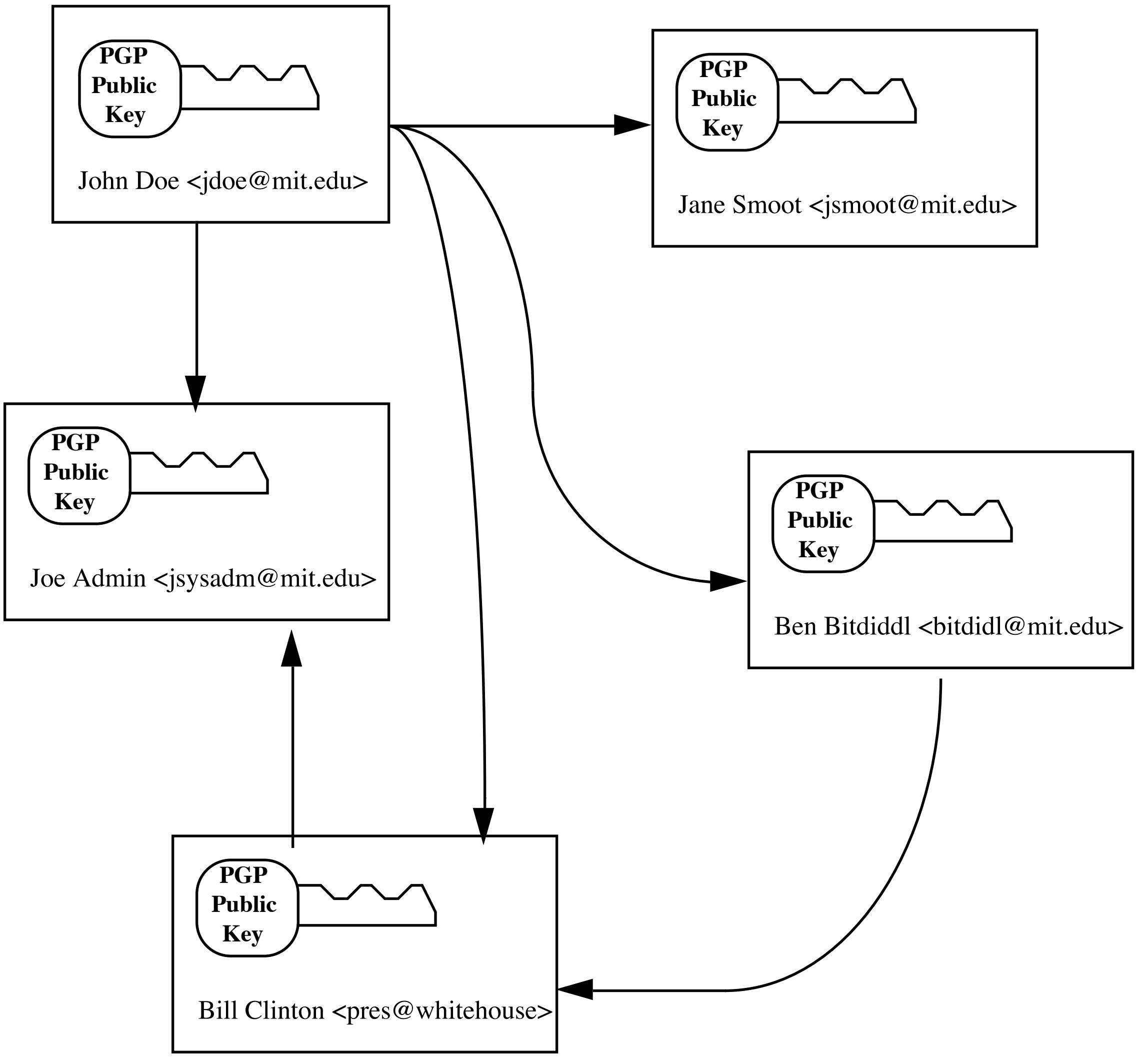


Figure 3: *Graphical Representation of a Multiply Signed Key*



John Doe vouches for:
☐Jane Smoot
☐Ben Bitdiddl
☐Bill Clinton
☐Joe Admin
Joe Admin vouches for:
☐*no one*
Bill Clinton vouches for:
☐Joe Admin
Ben Bitdiddl vouches for:
☐Bill Clinton
Jane Smoot vouches for:
☐*no one*

Figure 4: *A Sample Web of Trust*

As more people sign each others' keys, the Web of Trust builds up. Figure 4 schematically depicts a more complicated situation. The arrows between the keys represent the signatures. So the arrow pointing between John Doe and Jane Smoot represents that John has signed Jane's key.

The fundamental problem with the Web of Trust is that it depends on personal contact between many of the Web participants to build it up. It also depends on all of these participants properly verifying the identity of people whose keys they sign. It is not reasonable to expect a large group of people to be able to perform the necessary verifications without some significant number of them cheating. By cheating we mean that they sign someone's key without really knowing that the key really belongs to them. For example they may sign the key of someone who merely sends it by e-mail. Such a key cannot be trusted because the authenticity of e-mail is suspect in the absence of PGP, which is why we want the Web in the first place!

The Internet's Privacy and Security Research Group was aware of this problem when they designed the Internet standard Privacy Enhanced Mail (PEM). The PEM solution is to create a strict hierarchy of trust. At the root of this hierarchy is the Internet Society which will sign the keys of special organizations called Policy Certification Authorities (PCAs). In turn PCAs will sign the keys of organizations which finally will sign the keys of individuals.

Although this rigid hierarchical approach scales better then the PGP Web of Trust, it requires a fair amount of infrastructure to be in place before two people may use PEM to secure their e-mail. With PGP, if Alice and Bob wish to exchange secure messages and they have no common associates to Web their keys together, they can still meet in person and manually exchange keys. To use PEM, on the other hand, Alice's company will have to be registered with a PCA and be in a position to issue key signatures (called **Certificates** in the PEM terminology) to Alice. Bob's company will also have to be part of the game. Very few organizations are currently setup to use PEM. Most likely Alice and Bob are out of luck!

2.2. A Word on Names

PEM requires the use of X.500 style names. These names are of a special form and are not directly printable. By comparison, each PGP name is chosen by an end user. This flexibility is another reason why PGP has found quick acceptance in the community.

Although the PGP program will happily accept any string as the name for a key, Internet users typically use a name that would easily fit in the “To:” field of an e-mail message. For example a PGP key may be named:

John H. Doe <JDoe@xyzzy.com>

where John Doe's e-mail address is JDoe@xyzzy.com. Such a name may be provided to most RFC822 e-mail compliant systems without difficulty.

PGP is oriented toward electronic mail. Electronic mail does not have a strong performance requirement. Basically PGP processes mail messages and needs to be fast enough to run faster then reasonable human perception. However many applications require many transactions per second. Kerberos is designed to work in such an environment.

3. Kerberos

Kerberos was developed by MIT's Project Athena in 1986. It provides for cryptographically based secure real-time authentication of individuals and computers. Kerberos users are given **tickets** when they login which they can then present to network services to prove their identity. Kerberos was designed for real-time client server applications; it is fast and provides protection against replay of sessions and other security threats to a real time system.

In order to make Kerberos fast, and to avoid patent issues during its design, Kerberos today does not make use of Public Key technology. Instead it relies on the U.S. Data Encryption Standard[5] (DES) [DES] cipher system.

What this means is that with Kerberos one can prove to a particular service that a user is who she claims to be, but one cannot generate a digital document that is digitally signed such that anyone, now and in the future, can verify it.

Kerberos provides authentication services to a large number of users. A Kerberos **Realm**[6] can support over 100,000 users. At MIT we have approximately 25,000 users in our “ATHENA.MIT.EDU” realm and about 7,000 of them login every day. In general, each organization will have a Kerberos realm of its own. Kerberos can also be used to provide authentication between realms when a shared key is established by the realm administrators.

3.1. A Word on Names

Kerberos names have structure to them. In version 4 (and version 5 as provided by MIT) names have the form **name.instance@REALM**. Typically a company will use its Internet Domain name as its Kerberos realm name.[7] The **instance** portion of a name is typically only present in the names of computer services and in special circumstances. Most user names do not have an instance component. Therefore the John Doe of our previous example will likely have a Kerberos name of:

JDoe@XYZZY.COM

4. Combining Kerberos and PGP

Kerberos is designed for an organization size body of users, but doesn't provide digital signatures. PGP on the other hand provides document authentication via digital signatures, but its Web of Trust is cumbersome to use on an organization level. Below we present the design of a service which provides a way to integrate the PGP Web of Trust with the Kerberos authentication model. By doing so an organization can use Kerberos to leverage its use of PGP.

Our design goal is to provide e-mail and document security across an organization the size of MIT[8] and yet make no modifications to the basic Kerberos or PGP systems.

We do this by introducing a new Kerberos authentication service. We call this service the **PGP Signer** service. The PGP Signer service appears to PGP as just another user who has a public key. To Kerberos it is simply another service which makes use of Kerberos authentication.

To use the PGP Signer service, a user invokes the signer service client program. This program takes the user's public key (and name) and sends this in a Kerberos authenticated transaction to the PGP Signer server. The server, upon receipt of the transaction, compares the name in the PGP public key with the Kerberos authenticated name. This comparison is made using a set of rules that

---

[5] DES is a U.S. Standard Symmetric cipher. Kerberos uses it instead of IDEA simply because IDEA was not around when Kerberos was designed.

[6] A realm is the administrative domain of a set of Kerberos servers and the users they authenticate.

[7] Because Kerberos is case sensitive whereas Internet domain names are case insensitive, by convention realm names are always in all upper-case.

[8] The MIT community is roughly 14,000 people, 4,500 undergraduate students, 5,300 graduate students and about 4,000 faculty and staff.

determine if the claimed PGP name is congruent with the authenticated Kerberos name. If they are not congruent the request is rejected; If the names are congruent, then the PGP Signer service signs the PGP public key and name using its private key and returns the result.

To take full advantage of the PGP Signer service, each PGP user in the organization needs to obtain (usually at the same time as they obtain the PGP Signer client program) the PGP Signer's public key and add it to their personal PGP keyring as a trusted signer.[9] When this is done, the PGP Signer can act as a trusted introducer between people. Each user securely meets with the PGP Signer courtesy of Kerberos authentication.

When a group of users wish to exchange e-mail with each other, securing it with PGP, they merely need to use the PGP Signer as their introducer. As a program, the signer never sleeps and is always available to provide service!

4.1. Name Congruency

For PGP names and Kerberos names to be **congruent**, they need to describe the same entity. A simple congruency rule is to require that submitted PGP key names have the structure “Real Name <user@realm>” format. The PGP Signer may choose to ignore the “Real Name” portion of the PGP name and ensure that the portion between the angle brackets is equal to the Kerberos name. So:

John H. Doe <JDoe@xyzzy.com>

would be congruent to:

JDoe@XYZZY.COM

If the PGP signer has access to it a database mapping Kerberos user names to real names, then it can enforce that the supplied real name also corresponds to the actual user name. This permits the PGP Signer to reject names of the form:

Chief Executive Officer and King
<jdoe@xyzzy.com>

assuming that John isn't the Chief and King!

4.2. The Security of the PGP Key Signer

The PGP Signer service must run on a secure computer system. This is because it requires access to the PGP Signer's private RSA key. The PGP Signer's Private key is the private key generated along with the PGP Signer's public key, which is published. The PGP Signer **never** has access to any user's private key.Whoever obtains this private key may construct trusted PGP keys that do not belong to their claimed owner and the organization's security can be compromised.

Organizations that operate Kerberos servers are familiar with the requirement for a physically secure computer, as Kerberos key distribution servers also need to be managed securely. However the PGP Key Signer does not have to be as guarded as the Kerberos server because recovery from the compromise of the PGP Key Signer is not as disastrous.

4.2.1. Operating a Secure Signer Server

Along with using good host security techniques and providing physical security for the PGP Signer, some additional steps taken in advance can make recovery from a compromise easier.

The first step is to keep a copy of the PGP Signer private key in a safe place off-line. A floppy disk can easily store a PGP private key and should suffice for the off-line backup. We recommend that two such floppies be created and stored in safe and secure locations.

The PGP Signer should keep a copy of all public keys that it signs. This database of public keys should be backed up frequently.

---

[9] The Signer's key will be provided in a file. When PGP processes this file it will automatically recognize that it contains a PGP key and will ask the user if she wishes to add the key to her keyring. PGP will also ask if the key should be "trusted" to sign other keys.

Because it contains no confidential information, this backup of public keys need not be protected against reading, but should be protected against unauthorized modification. An off-line periodic tape backup suffices.

4.2.2. If the PGP Signer is Compromised

Unlike Kerberos, where compromise of the server is a disaster, security can be restored to a compromised PGP Signer with just a little effort.

The first step is to repair whatever host security or physical security problem resulted in the compromise in the first place. Once you are sure that the PGP Signer computing environment is safe, destroy the copy of the public key database and the file containing the PGP Signer private key (these will be “pubring.pgp” and “secring.pgp”, the standard PGP keyring files). Restore them from a safe backup. Generate a new PGP Signer key and use this to sign all the keys that are in the backup public key database. Use the old PGP Signer key private key to **revoke** the old PGP Signer public key. Finally distribute the new PGP Signer public key and ensure that the **key revocation certificate**[10] that PGP generated for the old key is distributed as well.

Although you will have to “go public” over the compromise of the PGP Signer, the steps that end-users will have to take to recover are minimal.

In the future it may be possible to store the PGP Signer private key in special purpose hardware. BBN manufactures a device that may be used for this purpose. Designed for PEM, the **BBN Safekeyper**™ may be adaptable for the use with the PGP Signer. The advantage that this hardware brings is that the PGP Signer private key can be stored in it in a fashion that makes it impossible to read it out. An intruder who gains physical access to the PGP Signer server may steal the Safekeyper, but a “read only” compromise where she secretly learns the PGP Signer private key is not possible.

5. Why not just use Kerberos?

At this point the reader may be wondering why we bother to integrate Kerberos and PGP in the fashion that we do here. Wouldn't it make more sense to simply extend Kerberos so that it can generate digital signatures and provide encrypted e-mail messages?

The answer to this has two parts. The first has to do with creating digital signatures.

To create a digital signature that is verifiable to anyone after it is created requires the use of Public Key Cryptography. Because Kerberos does not make use of Public Key Cryptography it cannot create a digital signature. The best that Kerberos can do is prove to one party that a message is from another party. However, before you can create the equivalent of a digital signature using Kerberos, you need to know the identities of the parties who will be verifying the signature. Using Public Key Technology one can create a signature that anyone, both known and unknown, can verify.

One might consider adding public key cryptographic algorithms to Kerberos in order to implement this function. However one of our design goals was to not make any modifications to the basic Kerberos system. It is also worth noting that if you add public key technology to Kerberos to implement digital signatures for messages you ultimately wind up re-inventing PGP (or PEM).

The other answer is more philosophical and has to do with confidentiality. Because Kerberos does not use Public Key Cryptography, a Kerberos “super-user” can always decrypt a message that was encrypted from one person to another using Kerberos techniques. Perhaps more ominous is that such a “super-user” can be compelled to decrypt messages presented to her under court order.

---

[10] When PGP is instructed to revoke a key it generates a key revocation certificate. It is a datum that instructs any invocation of PGP that receives it to consider the revoked key as invalid.

It is our belief that confidential messages should be exactly that: messages that are only readable by the parties communicating, which usually does not include the government!

6. Project Status

The PGP Signer design is described in this paper. A server is operating at MIT on the Internet host RFA.MIT.EDU. A client program exists for UNIX® systems. As of the time of the writing of this paper (November 1994) MIT has not released the signer system code. However, it is our intention to do so.

7. Future Work

Today PGP has a significant presence in the Internet community, a presence not shared by the standard PEM. However PEM may yet take on an important role as the required certificate infrastructure comes into existence, as it eventually will. A similar **PEM signer** can be created using most of the functions of the PGP Signer, but this signer will issue PEM certificates as well.

8. Conclusion

This paper has described the design of a simple service that permits a digital signature/privacy application, PGP, to leverage and bootstrap its authentication infrastructure from a real-time authentication service, Kerberos. It demonstrates that these technologies are not antagonistic to each other but instead are quite synergistic.

9. References

[IDEA] X. Lai, On the Design and Security of Block Ciphers, *ETH Series in Information Processing,* v. 1, Konstanz: Hartung-Gorre Verlag, 1992

[CertSniff] Cert Advisory CA-94:01 Ongoing Network Monitoring Attacks, Computer Emergency Response Team, Carnegie Mellon University, Pittsburgth, Pennsylvania (Februrary 3, 1994).

[DES] Federal Information Processing Standards Publication (FIPS PUB) 46-1, Data Encryption Standard, Reaffirmed 1988 January 22 (supersedes FIPS PUB 46, 1977 January 15)

[DiffieHellman] W. Diffie and M. E. Hellman, New Directions in Cryptography, *IEEE Transactions on Information Theory,* v. IT-22, n. 6, November 1976, pp. 644-654

[Kerb1] J G. Steiner, B. C. Neuman, J. I. Schiller, Kerberos: An Authentication Service for Open Network Systems, *Usenix Conference Proceedings* pp191-202, Dallas, Texas (February 1988).

[Kerb2] S. P. Miller, B. C. Neuman, J. I. Schiller, and J. H. Saltzer, Project Athena Technical Plan Section E.2.1: Kerberos Authentication and Authorization System, M.I.T. Project Athena, Cambridge, Massachusetts (December 21, 1987).

[PEM] Linn, J., Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures, RFC 1421, Internet Engineering Task Force, (February 1993).

[PGP] P. Zimmermann, PGP User's Guide Volumes I & II, MIT PGP Release 2.6, Cambridge, Massachusetts (May 1994).

[RSA] R. Rivest, A. Shamir, L. Adleman, A method for obtaining digital signatures and public key cryptosystems, CACM, Vol 21 No 2, pp 120-128 (February 1978).

10. Authors

JEFFREY I. SCHILLER received his S.B. in Electrical Engineering (1979) from the Massachusetts Institute of Technology. As MIT Network Manager he has managed the MIT Campus Computer Network since its inception in 1984. Prior to his work in the Network Group he maintained MIT's Multics timesharing system during the timeframe of the ArpaNet TCP/IP conversion. He is an author of MIT's Kerberos Authentication system. Mr. Schiller is the Internet Engineering Steering Group's (IESG) Area Director for Security. He is responsible for overseeing security related Working Groups of the Internet Engineering Task Force (IETF). He is also a member of the Privacy and Security Research Group (PSRG) of the Internet Research Task Force His recent efforts have involved work on the Internet Privacy Enhanced Mail standards (and implementation) as well as releasing a U.S. legal freeware version of the popular PGP encryption program. Mr. Schiller is also a founding member of the Steering Group of the New England Academic and Research Network (NEARnet). NEARnet provides Internet Access to institutions in New England.

DEREK ATKINS is currently a graduate student in Media Arts and Sciences at MIT. He is studying distributed, secure systems for multimedia data distribution. His intrests include, among other things, cryptography, flying, and playing guitar. He can be reached at warlord@MIT.EDU.