GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Security Essentials (Security 401)"
at http://www.giac.org/registration/gsec

Attacks on PGP: A User's Perspective

Ryan Thomas
GSEC Practical Assignment, Version 1.4b

Summary

The focus of this paper is to inform users of the practical and theoretical strategies that may be used in an attempt to compromise PGP (Pretty Good Privacy), potentially exposing the contents of a PGP encrypted message to an attacker. This paper will also recommend various countermeasures that can enhance a user's ability to defend themselves against some of the more realistic threats facing the use of Pretty Good Privacy. All countermeasures recommended in this paper may vary according to the specific system in place, the threat model associated with the organization and the level of sensitivity of the data being protected.

Practical Attacks

These attacks attempt to exploit the weaknesses present during the implementation and common, everyday use of PGP. The attacker may attempt to compromise a user's pass-phrase, gain access to the location of a user's private key or may deceive others by distributing fake or compromised public keys, et cetera. This sections intention is to inform users of the existence of these attacks and recommend various countermeasures that can help them securely and effectively use PGP.

Theoretical Attacks

These are attacks that attempt to exploit a possible weakness in the cryptographic methods deployed by PGP through mathematical techniques, or otherwise. There are numerous techniques for performing cryptanalysis and these attacks have a tendency to be extremely time, effort and cost intensive. This section will examine some common cryptanalysis techniques, and the feasibility of a successful attack against selected algorithms available to PGP users. Recommendations will also be made advising users of steps they should take to in order mitigate the threat that these attacks pose.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

An Introduction and Brief History of Pretty Good Privacy

What is PGP?

PGP (Pretty Good Privacy) is a data encryption program created by Phil Zimmermann in 1991, in response to United States Senate Bill 266[1], which was designed to force manufacturers of secure communications to provide a "Back door" which would allow the U.S. government to read those communications. The bill was ultimately defeated; PGP, however, has grown to become a de facto standard for encrypting e-mail and is widely used by home users, corporate and government offices worldwide. PGP uses public key cryptography to encrypt and decrypt files and messages so only those who are intended to read a message can read it. The concept of confidentiality and protecting one's privacy is nothing new. It has, however, become more urgent today because of the ease with which data (information in databases, e-mail, and so forth) can be accessed, intercepted and monitored[2].

Users of PGP can choose from several different algorithms. Please note that older v2.6 users can only use RSA/MD5/IDEA to decrypt and read messages. In version 5.0, PGP has introduced the use of Diffie-Hellman/ElGamal keys. The change in emphasis from RSA key usage to DH/ElGamal keys appears to have taken place because RSA remained patented until September 2000 and required payment of a royalty fee if used commercially[3]. The use of the Diffie-Hellman/ElGamal algorithm in PGP remains totally unencumbered by copyright and patents as of the date of this paper.

Why Use PGP?

No one wants his or her private e-mail or confidential documents to be read by anyone else other than the intended recipient. This is the same reason that traditionally most mail is sent in an envelope addressed to the recipient and not simply filled out on a postcard.

Electronic messages travelling on an insecure network like the Internet are no more secret than that same postcard sent in the mail. Potentially, whoever handles this message can read it. Even the United States courts have ruled that there exists no legal expectation of privacy for email communications[4]. By providing the ability to encrypt messages, PGP enables the user to add an “envelope” to the electronic letter or document.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

How it Works

PGP is a hybrid cryptosystem combining both conventional symmetric cryptography and asymmetric or public key cryptography. This means that it uses two different keys for encrypting and decrypting data: one public key and one private key.

Your public key should be distributed to PGP users you intend to communicate with, they will use this public key to encrypt messages specifically for you. Your private key must be kept secret; it is used to decrypt all the messages that have been encrypted for you by others using your public key. This means that the message encrypted using your public key can only be decrypted by you, the owner of the corresponding private key. You can also use your own public key to encrypt and store your own files/messages, then use your private key to decrypt them at a later date.

Practical Attacks on PGP

No data security implementation can ever be completely safeguarded from the attacks of an adversary. PGP like most cryptographic and security applications can be circumvented in a variety of ways. This section will cover several of the practical attacks (both technical and non-technical) on Pretty Good Privacy that do not involve cryptanalysis or any attack on the cryptosystem protocols, but instead the implementation of PGP on a data system.

Pass-phrase and Private Key File Compromises

One of the most common ways for an attacker to compromise the security of PGP on a system is to obtain another user's pass-phrase and/or private key file. If an attacker has access to the pass phrase and secret key file they can read the compromised user's encrypted messages and make signatures using that user's private key[5]. This weakness is not just specific to PGP; this is a typical weakness found in most password/pass-phrase authentication cryptosystems. Users may select easily guessed pass-phrases or may store their private key in a location where someone with malicious intent may access it.

An attacker may also use a tool or utility that will try to obtain a user's pass phrase from the local workstation. Brute force or dictionary attack utilities such as PGPCrack or PGPpass are designed to crack PGP encrypted files. The attacker may also use a keyboard logger utility that can capture the keystrokes of an unsuspecting user and save it to a designated location where they can then retrieve the user's revealed plaintext pass-phrase.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

It is important for PGP users to recognize how crucial the protection of their pass-phrase and private key file is to the confidentiality of their electronic documents and messages. With these two items a malicious user can decrypt, view, modify and distribute sensitive information. They can, for example, pose as the compromised user and fool others into disclosing or sending other sensitive information, potentially causing a great deal of damage to the parties involved.

There are several precautions a user can take in order to protect their pass-phrase or private key file. Users should always be aware of the physical location of their private key file and should not store the file on any machine that they do not have complete physical control over[6]. Users may wish to keep the private key file on a write protected floppy disk and then store it in a safe, adequately secured location like their home.

Users should not store their pass-phrase on the computer where their private or secret key file is located.

> "Storing both the secret key and the pass-phrase on the same computer is as dangerous as keeping your PIN in the same wallet as your Automatic Teller Machine bank card. You don't want somebody to get their hands on your disk containing both the pass-phrase and the secret key file. It would be most secure if you just memorize your pass phrase and don't store it anywhere but your brain. If you feel you must write down your pass-phrase, keep it well protected, perhaps even more well protected than the secret key file.[7]"

Users should also make sure they select a unique and adequately strong pass-phrase for their PGP application. This will dramatically reduce the effectiveness of dictionary and brute force attacks.

If a PGP user ever learns or suspects their pass-phrase or secret key file has been compromised all people that the user has exchanged PGP encrypted messages with should be contacted, warned of the compromise and instructed to stop using the user's public key. A new secret/public key pair can now be generated and the new public key of the user can be published/distributed in a message containing both the new public key and the key compromise certificate for the old key.

Public Key Tampering

In a public key cryptosystem, the public keys of users should be distributed so all have the components necessary to securely communicate and exchange information with each other. A crucial component in this system is the fact that users must be able to trust that a public key really belongs to whom it appears

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

to belong to[7]. This particular vulnerability is quite important to be aware of, many novices and even IT professionals may fall victim to trusting the tampered public key of an impostor. Once a user encrypts a document or message with the tampered public key and sends it, the impostor will be able to decrypt and read the contents. The user may have just unintentionally disclosed sensitive personal or proprietary company information.

PGP users should make certain that they only trust the public key of someone if they have got it directly from its owner, or someone the user directly trusts has signed the public key of the new user. This is important because an impostor would have a much more difficult time forging the signature of a user you already have a trust relationship with. Furthermore, the key's user ID should have the full name of the user, not just their first name. There are different strategies that can be deployed in larger enterprise environments as illustrated by this quotation found in the PGP User's Guide.

> "A trusted centralized key server or Certifying Authority is especially appropriate for large impersonal centrally-controlled corporate or government institutions. Some institutional environments use hierarchies of Certifying Authorities[7]".

If a user is asked to sign someone else's public key, it is of the utmost importance that the signing user makes certain the key really belongs to that person named in the user ID of that public key certificate. This is because the user's signature on the other user's public key is an oath by the signer that the public key really belongs to the new user. Other people who trust the signing user will accept the new user's public key because it bears the signers signature.

In order to make certain that a user's own personal public key ring cannot be modified or tampered with, the user should maintain physical control of their public key ring, preferably on their own personal computer. This is to protect it from tampering, not from disclosure. Keep a trusted backup copy of your public key ring and your secret key ring on write-protected media[6].

Operating System Attacks

Many of today's most widely deployed operating systems also pose another risk to PGP users. It is very similar to working in an office environment where employees discard their sensitive paper documents in the recycling bin instead of the shredding them and then disposing of them[7]. Many users are not aware that in a lot of cases, when a file is deleted only the file allocation information changes and the file contents still reside on the disk until that space is overwritten by another file. If PGP is used to encrypt a file, and the original document is then deleted, the user may think that the deleted file is completely

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

gone forever. When in actuality one of many disk recovery tools can be deployed on the local machine to recover and resurrect several of that user's previously deleted files. If a potential attacker has access to the deleted disk blocks before they have been reallocated, they may be able to recover the user's unencrypted original document.

Some applications such as word processors create a temporary copy of a user's document as they work on it. In most cases these files are deleted when the user finishes and closes the application, but fragments of the document remain on the user's drive in some area. Users may be unaware that these temporary files even exist, for example, they may be labelled "~WRL4054.tmp". If an attacker is resourceful enough and has access to the users hard drive these temporary files can potentially be accessed, viewed or even recovered in their entirety.

There are several third party "wiping" utilities (Norton, Entrust etc.) available to address these issues by overwriting the space with data before deletion. This prevents a determined attacker from using a disk recovery tool to retrieve a deleted file or scan the user's hard disk for any previously deleted sensitive documents.

Users should beware of applications such as word processors that use temporary files and they should make certain that they do not haphazardly leave sensitive plaintext documents or files on their workstations. All old media that contained secure data should be physically destroyed before being disposed of.

Trojan Horse Attacks

Trojan horse attacks pose one of the most serious threats to application security today. A Trojan horse is a seemingly harmless program that contains malicious code, which may infect a users PGP application or their operating system subverting the security of both. An attacker could use this code to capture a user's plaintext messages, or to obtain the user's pass-phrase or private key.

The wide scale use of the internet as a software delivery mechanism has increased the chances a user downloading and installing a rogue or modified copy of PGP software. Because the PGP source code freely available, any attacker with enough time, skill and motivation could potentially develop a "trojaned" version of the software. This trojaned version of PGP may be widely circulated, and may claim to be from a reliable or well-known source. A trojaned version may also be introduced to the target user's computer by other tactics. The attacker may replace the legitimate commercial or open source copy of PGP already installed on a user's computer with a rogue copy when the user is

© SANS Institute 2003, Author retains full rights.

not at their workstation. The attacker may also conduct a “Man-in-the-middle attack” hijacking a user’s download session from a legitimate download site and substituting the download with the attacker’s altered PGP copy. These altered versions may appear to behave like PGP in many respects, but they may be crippled in some manner, for instance the software may not check signatures properly, allowing counterfeit certificates to be accepted. The random number generation routine may be modified to produce predictable results, the cryptographic routines involved may be weakened, and the program may even encrypt the infected user’s messages with an additional key giving the attacker access to all files encrypted using the altered version.

Like most of the previously mentioned attacks, the education of the user is the key to preventing the circumvention of any security application present on a data system. Users should make a concerted effort to obtain their copy of PGP from a trusted and reliable source. They can also use a file comparison or checksum utility to verify the copy that has been obtained is indeed unaltered. Users can also search for a version of PGP that has been digitally signed by a trusted source such as MIT and then validate that digital signature. Digital signatures contain code that uniquely identifies the author of the software, guaranteeing that the application can be trusted once verified by the user.

The installation of an anti-virus software package and a firewall deployed to deny rogue network activity would also help to identify and prevent the appearance and/or propagation of malicious code on a user's workstation. Keeping the user's software patched, current and up-to-date will also help to ensure the workstation is protected from the majority of malware, as well as product and operating system specific vulnerabilities.

Electronic Surveillance Attacks

In addition to computer-based attacks against the implementation of PGP, a well-equipped attacker may attempt to compromise the security of PGP by remotely detecting the electromagnetic signals from a user’s computer. Electronic devices that display, store or transmit data emit Electromagnetic Radiation (EMR) signals. Possessing equipment designed to intercept and reconstruct this data, it is possible to steal information from unsuspecting users by capturing the EMR emitted from the devices they use. This costly and labour intensive attack has been used by intelligence agencies, law enforcement and even in the private sector for decades. It is important for security conscious users to be aware that these types of attacks do exist and tactics like video and audio surveillance may be used in conjunction with the other types of attacks described in this paper.

Make certain that all computer equipment and related cabling is properly shielded to Tempest standards so that it does not emit EMR. Tempest

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

originated from a United States military study examining the security of telecommunications devices that emit EMR. The word later became an acronym for Telecommunications Electronics Material Protected from Emanating Spurious Transmissions and an abbreviation of Transient Electromagnetic Pulse Emanation Standard[8]. Originally the United States government began to study this occurrence in order to prevent possible breaches in military security, but the Tempest standard can now be applied to commercial and personal technological devices shielding them from EMR leakage.

If a company's threat model deems it necessary, Technical Surveillance Countermeasure Methods can be deployed in order to ensure adequate protection.

Non-Technical Attacks

Many people feel that technological security solutions are a blanket solution when it comes to protecting their sensitive information. In reality, attacks using people and social engineering tactics can many times be much cheaper, more effective and efficient.

An attacker could very well be a fellow co-worker, in which case they may be in a situation where they can be in close physical proximity to their potential target. In this scenario, an attacker could attempt to compromise the pass-phrase or private key of the unsuspecting and possibly trusting target in a subtler manner. The employee may attempt to "shoulder surf" which is attempting to obtain the pass-phrase of another user by physically watching them type it into a workstation. There are hundreds of things an insider may say or do in an effort to obtain a fellow co-worker's PGP encrypted information. Insiders are, and will continue to be one of the biggest threats to corporate security and breaches resulting from the level of trust most employees have in one another.

A very determined and motivated attacker may even attempt to retrieve sensitive plaintext documents by physically breaching a company's security. The attacker may trespass during or after business hours, break and enter or even bribe/extort employees in an effort to gain access to the premises or sensitive company information.

> "How much effort do you wish to throw at the problem? The person can be filmed, their telephone bugged, the computer electronically scanned, keystrokes sampled, networks sniffed, IP's spoofed, routers attacked. If you are using PGP remotely, packet sniffers could be used (this is really no different to monitoring your plain text e-mail). We are now in a different ball game. If you are worth this much effort

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

then you have slightly bigger problems than worrying about the theoretical security of PGP." [9]

The appropriate level of paranoia associated with non-technical attacks a user should have when using PGP will vary drastically depending on the level of threat associated with the sensitive information involved. Users should utilize common sense and refer to company policies with regards to the handling of sensitive or proprietary information if they are in doubt. If a user is unsure if such a policy exists, an appropriate supervisor and/or manager within their particular department should be contacted for clarification purposes.

Theoretical Attacks on PGP

As previously stated, PGP is a hybrid cryptosystem. It contains four cryptographic elements: an asymmetric cipher, a symmetric cipher, a one-way hash, and a random number generator. Each of these four elements is subject to different types of cryptographic attack. An attack on the cryptographic elements deployed by PGP involving cryptanalysis is rather unlikely because it would be very expensive and would require vast technological resources to undertake. It may only be feasible by very formidable adversaries such as government intelligence agencies. The following section will examine the feasibility of a successful attack on PGP through some of the more popular cryptanalysis techniques. Furthermore, this section will also examine the selected algorithms used in the Diffie-Hellman/ElGamal public key version of PGP.

Cryptanalysis Techniques

Cryptanalysis refers to the study of ciphers, ciphertext or cryptosystems in an attempt to reveal a weakness, exposing the original plaintext message from the encrypted ciphertext. Users should always assume that a potential attacker knows all the details of a targeted system, but do not have knowledge of the cryptographic keys being utilized; this principle is stated in Kerckhoffs' law[10]. Therefore to successfully break a cryptosystem an attacker would have to reconstruct the user's key through observing the cryptosystem in operation. These observations on the manipulations of the cryptosystem then allow them to determine the best mode of attack. The goal of the cryptanalyst is to find a way of cracking the encrypted message that is easier than a brute force attack.

Passive Attacks

One such way or mode is called a passive attack. In this mode attackers are restricted to only making observations on the cryptosystem as it operates. Although passive attacks are still rarely attempted, it definitely serves in the

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

best interest of PGP users to be aware of their existence. In a chosen-ciphertext attack the cryptanalyst intercepts encrypted messages then attempts to find statistical regularities in the stream of ciphertext, or departures from randomness that may potentially expose information about the key used for encryption. A somewhat stronger passive technique is a known-plaintext attack where the cryptanalyst observes both a stream of ciphertext and the corresponding plaintext stream that produced it. Since data such as email messages contain header information it is probable that the cryptanalyst can obtain copies of both some original plaintext as well as the encrypted ciphertext.

> "PGP encrypted files carry a header stating that PGP was used and the encryption method. This at least tells an attacker where to start and what tools to use, even if that attack using current technology and mathematical knowledge may not be successful. The headers can be stripped off. All that is then left is a file of random data[9]."

Most non-proprietary cryptosystems in use today face extensive security analysis and produce ciphertext with a very high degree of randomness, any cryptosystem broken by these types of attacks are generally considered weak, obsolete and are probably not available in recently published cryptographic applications.

Active Attacks

In an active attack the cryptanalyst actually interacts with the ciphertext and/or plaintext and may alter or inject something into the data stream. These active attacks allow the cryptanalyst the greatest amount of freedom when analyzing a cipher. In a chosen-plaintext attack the cryptanalyst chooses plaintext to be encrypted and then analyzes the plaintext together with the resultant ciphertext in an attempt to obtain the secret key. A clever attacker may alter the typical chosen-text attacks by making them adaptive. This means that the cryptanalyst has the additional ability to choose the text that is to be encrypted or decrypted after seeing the results of previous requests. There a various adaptive techniques that may also be attempted by a potential attacker, below this paper will describe an adaptive chosen-ciphertext or "Man-in-the-Middle" attack:

This attack differs slightly from a chosen-cipher attack because it involves tricking users into surrendering their keys, and is independent of the algorithms in use. In this scenario there are three people: the sender, the recipient and the malicious attacker. The attacker intercepts an encrypted message from the sender intended for the recipient; unable to open the message the attacker then re-encrypts it with the attacker's key. The attacker then forwards the newly re-encrypted message to the unaware recipient. Once the recipient receives the attacker's re-encrypted message, they attempt to use their private key to decrypt the message. However, since the message has been re-encrypted

© SANS Institute 2003, Author retains full rights.

using the attacker's key, it does not decrypt completely and remains scrambled. The recipient puzzled by the unreadable message replies to the attacker for clarification with the scrambled text in their reply, and if their PGP application is set to encrypt automatically, the attacker may be able to read the original sender's message in plaintext.

Although a recently published paper by Counterpane Internet Security has raised some concern by proving that an adaptive chosen-ciphertext attack is in fact possible in practice[11]; there are numerous complications associated with this attack, making it difficult and somewhat unlikely. The attacker must convince or tempt the intended recipient into replying. If they're already acquainted, this may be easy. If they are strangers, then a bit of social engineering may be required. The most obvious point of failure in this scenario is the recipient not responding to the message. The recipient also has to set PGP to encrypt messages automatically, or somehow choose to encrypt a reply to what appears to be a nonsense message from the attacker. Users should again use common sense to avoid falling victim to a "Man-in-the-Middle" or adaptive chosen-cipher attack. Users should never sign random or nonsensical documents that they receive; a one-way hash of the message should be used instead.

As stated before, attacks involving cryptanalysis require a substantial amount of computational effort and resources. A potential adversary needs to be able to accumulate the amount and the type of data that will successfully enable them to conduct a successful attack on the cryptosystem. All of these are important factors in assessing the practicality of these types of attacks. They remain in many circumstances difficult to mount, but PGP users should not ignore or completely disregard them.

Brute Force attacks

A brute force attack involves the systematic searching through all possible keys, meaning the attacker tries every possible combination of key parameters. If a sufficient key length is used, trying all possible keys until the message can finally be decoded will most likely be doomed to failure, especially if the attacker has a relatively limited amount of time and/or resources.

> "The possibility of doing brute-force key-space searches is often compared to the age of the universe, number of atoms in the planet earth, and the yearly output of the sun. For example, Bruce Schneier has calculated that according to what we know of quantum mechanics today, that the entire energy output of the sun is insufficient to break a 197-bit key" [12].

Basically users should choose adopt a two-pronged strategy in order to make completely sure that any attempt at a brute force attack would prove to be

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

unproductive. An adequate key length should be selected rendering a search of the keyspace difficult and time consuming; this paper will recommend adequate key lengths for selected algorithms in a later section. Users should also make certain that they change or upgrade their key in a somewhat regular frequency.

The Asymmetric Algorithm

PGP is largely based on asymmetric encryption. In asymmetric encryption every user has a pair of keys: one is public and the other is private and must be kept secret. The strength of the asymmetric key used is crucial to the secure use of PGP. Hypothetically, if an attacker can break a PGP symmetric key they will be able to read a single message. However, if an attacker is able to break the PGP asymmetric key all encrypted documents or messages of the past, present and future may be compromised. Therefore it is very important that the public key algorithm PGP users select is proven to be strong, secure and immune to cryptanalysis.

PGP is available in two public key versions: the traditional RSA version and the openly published Diffie-Hellman/ElGamal version. RSA has diminished as the algorithm of choice in the more recent versions of PGP and is no longer supported in freeware versions due to a number of issues. The Diffie-Hellman/ElGamal public key version that shall be discussed in this section is essentially a straightforward extension of the Diffie-Hellman key exchange concept where a shared secret is generated and used as a one-time pad to encrypt one block of data. The security of Diffie-Hellman is based on the difficulty computing the Discrete Logarithm Problem (DLP); this difficulty is illustrated in the quote below:

> "The best way to describe this problem is first to show how its inverse concept works. The following applies to Galois fields (groups). Assume we have a prime number P (a number that is not divisible except by 1 and itself, P). This P is a large prime number of over 300 digits. Let us now assume we have two other integers, a and b. Now say we want to find the value of N, so that value is found by the following formula:
>
> $N = a^b \bmod P$, where $0 <= N <= (P \cdot 1)$
>
> This is known as discrete exponentiation and is quite simple to compute. However, the opposite is true when we invert it. If we are given P, a, and N and are required to find b so that the equation is valid, then we face a tremendous level of difficulty" [13].

Diffie-Hellman or DH is generally considered to be secure when sufficiently long keys and proper generators are used. Like with most other algorithm related

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

issues users should make sure their keys are of adequate length and considered strong.

The Symmetric Algorithm

When using a symmetric cipher, both the sender and the receiver must know the same secret key in order to communicate with one another. In symmetric encryption the file or message is both encrypted and decrypted with the same key. With PGP, a symmetric encryption key is created for each encryption session. This session key is used to encrypt and decrypt the file or message and the session key itself is encrypted asymmetrically. Recent versions of PGP allow users to select a preferred symmetric algorithm that will be used when people encrypt messages for them. The options available to users include: IDEA, CAST, Triple DES and in newer releases the AES and Twofish algorithms. All of the algorithms are considered strong and adequately secure enough to use with confidence. All Diffie-Hellman/ElGamal private keys are encrypted to the CAST, and will therefore be the symmetric cipher examined in this section.

Entrust Technologies' CAST symmetric encryption algorithm, is one of the fastest and most secure algorithms available. This version of CAST known as CAST5, or CAST-128 has a block-size of 64-bits and a key length of 128-bits. Research conducted for this paper has shown that CAST is resistant to most known cryptanalysis techniques and that there really is no known way of breaking CAST short of brute force[14].

The One Way Hash Function

The one-way hash in PGP is used to hash the passphrase into the symmetric key and to digitally sign documents. The receiver uses the sender's public key to decrypt the hash code. If it matches the hash code is sent as the digital signature for the message, then the receiver is sure that the message has arrived securely from the stated sender. The RSA key version of PGP uses the MD5 algorithm to generate the hash code for backward compatibility issues, and the Diffie-Hellman/ElGamal key version uses the stronger SHA-1 algorithm to generate the hash. The SHA-1 algorithm will be the hash function examined in this section.

The Secure Hashing Algorithm or SHA-1 is a cryptographic hash algorithm published by the United States Government. SHA-1 is one of four algorithms announced on August 26, 2002 to be tested and validated to the FIPS 180-2 secure hash standard[15]. It like other message digests generates a condensed representation of a message. SHA-1 produces a 160-bit hash value from an arbitrary length string and is considered to be very secure.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

The Random Number Generator: PRNG

Software random number generators cannot be absolutely random; therefore they are called pseudo random number generators or PRNG. Pseudo random number generators usually query the user for some random seed information when they are activated. The PRNG used in most versions of PGP is called the ANSI X9.17 generator and it conforms to the FIPS 186 standard[16].

When installing PGP, the user is prompted to enter some text from the keyboard. The trueRand function then measures the time spacing between the user's keystrokes, and then saves that information in a file called "randseed.bin". That file then serves as the seed information for future PGP cryptographic use by the X.917 generator. The true randomness of the trueRand function is completely dependant on the input from the keystrokes received. In order to ensure maximum randomness the user should make certain their keystrokes are as random as possible.

Most PGP users do not give the PRNG much thought, but random numbers are used quite frequently in cryptography. Random numbers are present in session keys, initialization vectors, public-key generation, and many other places. In many cases if the random numbers generated are insecure, the entire cryptographic application is can be considered insecure. For the most part, the random seeds generated by X.917 are considered strong enough for cryptographic purposes because they represent more bits of information than used in most keys. Users should however, keep in mind that in many cases the cryptographic algorithms and protocols can't cover for a bad random number generator.

Conclusion

This paper was intended to inform users of several of the practical and theoretical strategies and attacks that may be used in an attempt to compromise the Confidentiality, Authenticity and Integrity of PGP (Pretty Good Privacy) which may potentially expose the contents of a PGP encrypted message to an adversary. This paper also made recommendations on possible countermeasures that may enhance a user's ability to defend themselves against some of the more realistic threats facing the use of Pretty Good Privacy. The countermeasures suggested in this paper should be modified according to the specific system in place, the threat model associated with the organization and the level of sensitivity of the data being protected.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

[1] Senate Bill 266. cpsr.org “SEC. 2201. Cooperation of Telecommunications Providers with Law Enforcement”
URL: http://www.cpsr.org/cpsr/privacy/crypto/s266_and_commentary.txt (1991).

[2] Caftori, Lal, Rosenberg, Poole. "Tutorial for Beginners to PGP". (Revised April 8, 2001).
URL: http://www.neiu.edu/~ncaftori/PGP.htm - foreword

[3] McCune, Tom. ”PGP Questions & Answers, RSA Support?”
URL: http://www.mccune.cc/PGPpage2.htm#RSAsupport

[4] Unruh, Bill. "Cryptography". 1998.
URL: http://axion.physics.ubc.ca/crypt.html

[5] Orsulic, Josko. “PGP User’s Guide, Volume II: Special Topics, Compromised pass phrase and secret key”
URL: http://pgp.rasip.fer.hr/pgpdoc2/pgpd2_53.html

[6] Orsulic, Josko. “PGP User’s Guide, Volume II: Special Topics, How to protect secret keys from disclosure”
URL: http://pgp.rasip.fer.hr/pgpdoc1/dis.html

[7] Zimmermann, Philip. “PGP User's Guide, Volume I: Essential Topics: How to protect public keys from tampering”.
URL: http://www.stud.uni-hannover.de/stud/serv/pgpdoc/pgpdoc1/how-to-protext-pk-from-tamp.html

[8] “Webopedia.com Definitions: Tempest”. 13 January 2003
URL: http://www.webopedia.com/TERM/T/Tempest.html

[9] “CommSec – How Secure is PGP?”. 01 January 2001
URL: http://www.commsec.com/pgpinfo/pgp-secure.htm

[10] “Wikipedia.org Definitions: Kerckhoffs' law”. 23 December 2002
URL: http://www.wikipedia.org/wiki/Kerckhoffs%27_law

[11] Jallad, K., Katz, J., and Schneier, B. **“**Implementation of Chosen-Ciphertext Attacks against PGP and GnuPG”. 2002
URL: http://www.counterpane.com/pgp-attack.html

[10] Unruh, Bill. "The PGP Attack FAQ". 1996.
URL: http://axion.physics.ubc.ca/pgp-attack.html

[11] Fisher, Denis. “Experts Debate Risks to Crypto”. 27 March 2002.
URL: http://www.eweek.com/print_article/0,3668,a=24663,00.asp

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

[12] Graham, Robert. “Hacking Lexicon”. 11 November 2001.
URL: http://www.robertgraham.com/pubs/hacking-dict.html - brute-force

[13] “SearchSecurity.com Definitions: Cryptanalysis”. 12 January 2003
URL:
http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci214532,00.html

[14] Simpson, Sam. “PGP DH vs. RSA FAQ“. 20 September 1999.
URL: http://www.samsimpson.com/pgpfaq.html#SubCAST

[15] **“**Cryptographic Toolkit: Secure Hashing”. 29 August 2002
URL: http://csrc.nist.gov/CryptoToolkit/tkhash.html

[16] Simpson, Sam. **“**PGP DH vs. RSA FAQ”. 20 September 1999.
URL: http://cryptome.unicast.org/cryptome022401/pgpfaq-ss.htm#SubRNG

© SANS Institute 2003, Author retains full rights.