# PGP: Pretty Good Privacy

## What is PGP?

Pretty Good Privacy was originally developed by Phillip Zimmerman to provide a means of secure communication in an insecure electronic environment. “Pretty Good” is an understatement – the framework it is based on, PKI (Public Key Infrastructure) and its encryption standards (it can use Diffie-Helman or RSA algorithms of varying strengths), have been subjected to rigorous cryptanalysis.

PGP has since grown into a more versatile application under the direction of its current owner, Network Associates (www.nai.com). Until the most recent release PGP has been completely open source, allowing anyone to review the code and suggest improvements.

## How does PGP work?

When someone starts using PGP, they generate a Key Pair. These are really just text files that look like gibberish to a human. The keys can be created at various levels of strength – 512, 1024, or 2048 bit strengths are used. The higher the number, the stronger the encryption value of the key. One key of the pair is the Private key – this key should always be kept safe and never given to anyone. The other key is the public key – this key should be given to as many people as possible.

## What are the uses of PGP?

The most commonly used aspect of PGP is the signing and encryption of email or files. “Signing” a document is a way of verifying the integrity of the original work. The method is as follows:

1. Make a digest or “hash” of the file or email. A hash is an algorithm that produces (theoretically) a unique output (the hash) from a given input (the message).
2. Add the hash to the end of the message.
3. When someone wants to verify that the message has not been modified, they run the hash algorithm on the message and compare it to the hash at the end of the message. If the signatures match, the message has not been altered.

This is demonstrated in the following example:
The hash algorithm: take every third letter of the message (ignore punctuation), and convert the letter to a number (a=1, b=2…z=26). Add the numbers together.
The message:

Hello, This is a sample message to demonstrate signatures.

The hash algorithm in progress:

He**l**lo, **T**hi**s** is **a** sa**m**pl**e** me**s**sa**g**e t**o** de**m**on**s**tr**a**te **s**ig**n**at**u**re**s**.

```
12 +20 +19 +1 +13 +5 +19 +7 +15 +13 +19 +1 +19 +14 +21 +19
= 217 (therefore the hash value is 217)
```

The message after adding the hash value becomes:

Hello, This is a sample message to demonstrate signatures.
Hash value: 360

If the message is altered, the hash value will not be the same.

Altered message:
Hello, This is an altered message to demonstrate signatures.
Creates a new hash:

```
Hello, This is an altered message to demonstrate
signatures.
12 +20 +19 +1 +12 +18 +13 +19 +5 +4 +15 +20 +20 +9 +1 +18
```

= 206 (therefore the hash value is 206)

***Since the hashes are not equal, the message has been altered.***

Actual hashing algorithms are much more complex. Additionally, the hashing algorithm is used in conjunction with the user's private key in such a way that the signature is unique. That is, if different people (thus different private keys) signed the same email, the signatures would be different. Then the public key of the key pair is used to compare the hash created by the private key, and if the hashes match, then two things are assured: 1) The message has not been modified since signing and 2) the signature was not be forged.

**Encryption:**

Encryption is a method of changing plaintext (text that is readable by humans) into ciphertext (text that is meaningless to humans). There are many different ways of encryption, some stronger than others. Two main categories of encryption are symmetric and asymmetric. In symmetric cryptography, the same key that encrypts a file also decrypts it. In asymmetric cryptography, which is what PGP uses, one key (the public key) encrypts the file, and the other key (the private key) decrypts it. So, if user A wants to send an encrypted message to user B, user A would first obtain user B's public key. This is possible because public keys are meant to be widely distributed. Then user A encrypts the message using user B's public key. The encrypted message can now only be decrypted with B's private key, which only he possesses. Not even user A, who wrote the message, can decrypt what he has encrypted, because he does not possess user B's private key. This ensures that the message is unreadable by anyone other than user A.

Encryption and signing are often combined. In this scenario, user A would use user B's public key to encrypt the message, then use his own private key to sign the message. This will ensure that no one but user B can read the message, and when user B receives it, he can be assured that the message was not altered. To read the message, user B would first use user A's public key to verify that the signature matches. Then user B would use his private key to decrypt the message that user A wrote.

For more information on PGP, visit www.pgp.com.
The freeware version of PGP can be downloaded from any of the following sites:

From PGP (owned by NAI)
http://www.pgp.com/products/freeware/default.asp
    Windows and Macintosh Versions

From MIT
http://web.mit.edu/network/pgp.html
    Windows, Macintosh, AIX/HP-UX/Linux/Solaris

PGP international:
http://www.pgpi.org
    Many versions, with translations into many languages, pgp news.

Licensed versions of PGP can be bought at:
    McAfee
    http://mcafeestore.beyond.com/Product/0,1057,3-18-ML100111,00.html