Global Information Assurance Certification Paper

Title: PGP for Everyday Use
Name: Jeremy Hoel
Assignment: GSEC Assignment v1.4b

Abstract

As an encryption program, PGP has become a common tool for everyday encryption and security. Its use allow for simple, easy and fairly complete verification and encryption of files and messages. There are many versions of PGP and many different tools for a wide variety of operating systems.

This paper will also show briefly what PGP is, where to get PGP from and what the differences are between the versions. It will also address how to use PGP for everyday tasks such as e-mail signing and encryption along with file encryption and signature verification. This will include simple key management, verification of files and verification of signatures. A general understanding of how encryption works would be helpful, but is not required.

What is PGP?

Originally designed and programmed by Phil Zimmerman in 1991, PGP is a freely downloadable encryption program that uses a wide variety of publicly proven algorithms with a variety of key lengths? It has the ability to do signing, digital verifications, public and private key generation, file encryption, and can even encrypt Internet phone conversations. It has had a long history and has gone from public code review and releases to private ownership (from NAI) and come back again to being opened back up (both in source and free downloads) in the current for of PGP Corporation [1]. It was also the origin of many legal battles against Phil Zimmerman and a large number of legal questions dealing with encryption for private use and the exportation of encryption to other countries [2]. Since its one of the most commonly used programs for its purpose, it understandably has been reviewed many times by many different people and can be trusted.

Originally created for MS-DOS, PGP was quickly ported to UNIX, Linux and a wide variety of other operating systems. It now currently has ports for Unix, Linux, MS-DOS, OS/2, Amiga, Atari, BeOS, EPOC (Psion), PalmOS, VMS, Windows (3.1, 9x, NT, 200 and XP), MacOS and more [3]. It has also spawned the openPGP format, RFC 2440[4], that is used in other encryption programs to make them compatible with PGP formatted messages, files and key.

Now that PGP Corporation has control of the PGP product, they have decided to go back to public review of PGP source and releasing PGP Freeware free for

home and personal use. They also sell and license PGP for business and enterprise situations.

Where to get PGP

The latest version of PGP (PGP 8 freeware) for Windows or Macintosh can be obtained from PGP Corporation (www.pgp.com). This downloaded version includes all the features of PGP Personal 8, but some of these features are locked (such as PGPdisk and the mail plug-ins) and can only be unlocked once an individual purchases a license. If one wants an older (more peer review, more known bugs fixed) version or a version for a different operation system, refer to www.pgpi.org, the International PGP homepage. It should also be noted that some of the other older versions have the ability to use the plug-ins built in. Check their particular websites for more information. There are also a number of other lesser well-known PGP tools that can be obtained from search engines like Google.

Getting Started with PGP

From here on out, this paper uses the assumption of PGPFreeware 8, Windows 2000 and Outlook express as mail client. Since this system layout may differ from your system, some text, buttons or procedures may be slightly different, but you should be able to get the general idea.

Once you get PGP installed onto your system, the first thing you will do is to create a public/private key pair. When making your first key pair, it's important to consider a few things (these options are accessible via the expert key creation button instead of using the wizard):

1 – What algorithm to use? With the current version of PGPfreeware there are three choices: Diffie-Hellman/DSS (DH/DSS), RSA or RSA Legacy (RSAL). The default is DH/DSS and should be fine for new users. However, if you are going to be corresponding with people that have used PGP in the past or that use RSA keys you might want to consider making a RSA key pair. The difference between RSA and RSAL is that the RSAL keys do not include a method for Addition Decryption Keys (ADK) and other newer message recovery methods and key features.

The security of each of these methods and the possible attacks are beyond the scope of this paper, but have been well researched [5] and can be found from various sources on the net. For this paper, we will use a DH/DSS key.

2 – How big do you want the key? The purpose of encryption is to keep messages or files out of the public eye. So you, as a user need to determine how long you might want this information hidden. Depending on the key type

(algorithm) and size, it could take days or years to have someone crack the message.

For most purposes (encrypting mail and files for a few years, until the information is of no great value to anyone anymore) a key size of 1024 or 2048 is fine (in fact, with PGPFreeware you can only generate RSA keys up to 2048, but this restriction is removed in other PGP versions that are available [6]. It should be said that there are a number of articles that show that 512 bit keys can be cracked in a few months [7] and that it is suggested that for long term storage (many years), keys of 2048 or greater be used.

3 – Key expiration date. If you want to make a key that is active for a limited amount of time, you could turn this on, otherwise, leave it off (no expiration date), If you need to revoke a key for some reason, use the tools in the software to revoke a key.

4 – The passphrase. This is one of the most important steps (next to backing up your keyring). If someone were to get a hold of your secret key, and figure out your passphrase then there is NOTHING that you can do to prevent that person from opening your encrypted files or reading your mail.

The passphrase should be a good length, and include upper and lower case characters, number and symbols. A good choice is to make a sentence and add punctuation and numbers. Of course, you should not write it down anywhere where others may find it. If your passphrase is weak, then it can be hacked.

Once you have filled out the required blocks and generated random data, the key creation takes place. At this point, it is a very good idea to back up this newly created key pair. If you loose the private key you will not be able to read any messages encrypted to you. So once you make this key, make a copy of both parts of the key on a reliable backup medium and store it in a safe place.

To do this, launch the PGPkeys application, go to the *Keys* menu, then *Export* and select the checkbox that says “Include Private Keys”. Give it a location and hit “Save”. Extreme care now needs to be taken with this file. It contains both your public and private keys. If someone got this, and figured out your passphrase they would be able to send and open mail, pretending to be you and you wouldn’t know. All you could do is revoke that key and start over again. A good form of security would be to encrypt this file with a conventional encryption and then store in a secure location and not leave the plaintext version lying around on the computer.

Sharing your public key

Now that you have a key pair created, you need to get your public key out to where people can use it. You can add your key to the keyservers that are available or just export it and sign it, giving it to those who you want to have it. Both of these methods will be discussed.

The purpose of sending keys to keyservers is so that anyone can (knowing your name or key fingerprint) download your public key to use it. This allows someone who hasn't contacted you yet to be able to send you something encrypted from the start. However, it should also be noted that some trust is then placed on the keyservers to provide security and prevent public keys from getting tampered with. This hasn't been a problem in the past and is not likely to be a problem in the future; it's just something that should be mentioned.

From the PGPkeys screen, go to the *Server* menu then *Send To* and pick either keyserver.pgp.com or europe.keys.pgp.com (you can use other key servers also, there are many available (some more being MIT's at psp.mit.edu and the openPGP key server www.keyserver.net/) and you can add them to PGPFreeware by going to *Edit*, *Options*, clicking on the *Servers* tab and adding them there.

The other method of sharing keys is to export your key, sign it and then pass it to whomever you want to have it (either via floppy, e-mail or a website link). The reason you sign your exported public key is to allow the recipient to ensure, once he has imported that key, that it is a valid key and hasn't been tampered with. If I export my key, and then someone changes it before you get it, then I will not be able to open any messages you send to me and in fact, if the person that did the tampering inserted his public key, he would be able to read messages that where not meant for him. By signing the key, it has my name and time stamp and can be verified.

To export your public key only, follow the same method described above (go to the *Keys* menu, then *Export)* but this time, do not check the box labeled "Include Private Keys". Again, give this file a location and a name and hit "Save". Now we are going to sign this file, so that it can be verified later by some recipient. There are two ways to sign the file (since it's a plain text file, with binary files you can only make a separate signature file) the first is to open it in a text editor and sign the text in the file itself and the other way is to do a digital signature of the file as a whole.

To place a signature on the key inside the file itself, find the saved .asc file and open it in notepad (again this assumes windows, if you have another OS, your procedure will differ). You should see some text that looks like this:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGP 8.0 - not licensed for commercial use: www.pgp.com

mQGiBD5ZVksRBADX2PFLVVeEbeqS+NAa8l2CmF69LtXnSWREqf7ArfZVhPQgRGmP
..more key text here..
oJXK+rGjpQVmCd1d9jJg4B8pK9ecAJwI/gycZn0b10ZY+l0R8OtdiwjX0Q==
=vVxL
-----END PGP PUBLIC KEY BLOCK-----
```

With the file open, highlight the text and then copy it into your clipboard. Go to the PGPtray (it should be running in your taskbar), right click on the lock icon, choose *Clipboard* then *Sign*. Enter your passphrase at the pop-up and then choose OK. Now, back in clipboard, with all the key text highlighted, hit "Paste". Now you see that your key is signed and looks like this:

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

- -----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGP 8.0 - not licensed for commercial use: www.pgp.com

mQGiBD5ZVksRBADX2PFLVVeEbeqS+NAa8l2CmF69LtXnSWREqf7ArfZVhPQgRGmP
.. more key text here..
oJXK+rGjpQVmCd1d9jJg4B8pK9ecAJwI/gycZn0b10ZY+l0R8OtdiwjX0Q==
=vVxL
- -----END PGP PUBLIC KEY BLOCK-----

-----BEGIN PGP SIGNATURE-----
Version: PGP 8.0 - not licensed for commercial use: www.pgp.com

iQA+AwUBPlleBz8soWhnOt1ZEQKQsQCgkTpNw8DOrDqku2s5XClwCSKFFjkAljD6
+VXvvaf+ZDcsCAPibFmOpxI=
=SfVE
-----END PGP SIGNATURE-----
```

This shows the key has been signed. Any text within the "`-----BEGIN PGP PUBLIC KEY BLOCK-----`" and "`-----END PGP PUBLIC KEY BLOCK-----`" has been verified. If someone changes that text, when the signature is checked it will not pass and you know that something has changed. Save this file either as a new file or over the old one and put it away for later use.

Digitally signing a file

The other method involves creating a digital signature on the file itself. This is a hash that is one way, and when checked, if it doesn't match the outputted, already saved signature then you can tell that the file has been tampered with. To make a signature file, launch the PGPmail application from the PGPtray and click on the icon of the "paper and pen". It opens a dialog box, choose the originally saved .asc file and click "Open". In this case, since the input is a text file, go ahead and check "Input is text". You may also want to check "Text output" for better compatibility. Enter your passphrase and hit ok. This makes a file called <filename>.sig". That is the signature file that goes with your key file. Now someone can verify the integrity of your key file with the signature file.

When someone asks for your PGP key, you can give them the public key file that has the signature inside it (the first method), the untouched public key file and the signature file that goes with it (the second method) or tell them what key server your key is on and give them the key ID (found by choosing key properties on your key and looking at ID, 0x673ADD59 for example). Now people can begin to send you messages encrypted to your public key.

Adding keys to your key ring

Now that you have your key out where people can get it, you need to get the public keys of others in order to send them encrypted messages. There are many ways to get keys. You can find them on people's web sites, receive them via e-mail or have them given to you on a disk. The method of getting them into your key ring is pretty much the same.

If you see a key on a web page on in an e-mail message and you want to add it to your list, highlight the key (the part from the "`-----BEGIN PGP PUBLIC KEY BLOCK-----`" to the "`-----END PGP PUBLIC KEY BLOCK-----`" then copy that text to the clipboard (hitting control-c, or right clicking and hitting copy). Go up to PGPtray, choose *Clipboard* and select *Decrypt and verify*. This brings up a new dialog box that shows the key you want to import. Select key and hit import. Now if you go to your PGPkeys program, you will see the new key in your key ring.

If you get a file that has an extension of .asc you can just double click on it and it will bring up the prompt to add the key to your ring.

Before you can use these new keys you have to sign them. This ensures that in the event someone else added keys to your ring, you still couldn't use them unless you have signed for them and trust them. To sign the key, right click on the key and choose "Sign". You can here decide if you want your signature to be exportable (go back to the key severs) or just to stay on your keyring. By exporting your signature, you can have other people that may not trust someone see your signature and they trust you so they will trust them. It's part of what's called the "web of trust"[8].

PGP and e-mail

PGP Personal comes with support for a few of the main e-mail clients in use today; including Eudora, Outlook, Outlook Express, GroupWise and ICQ. Plug-ins for other mail programs is available from other web sites on the web [9]. There is even support for pine, mutt and kmail in *NIX just to name a few. This support normally allows you to sign, verify, encrypt, decrypt and add keys from the mail program. This saves time, as you don't have to save the message out of the program and then load it up separately in another program to do any of the PGP functions. PGPfreeware has these features but they are locked and require a license key to use them. For general purpose use we can work without using

the plug-ins, but if you want to support the product or have those features, you need to contact pgp.com to purchase a license.

For starters, let's send ourselves a signed message. Open your mail client, enter your own mail address, a subject and enter a small message. Leaving the cursor in the main window, go up to the system tray, right click on the lock icon, and choose *Current Window* then *Sign* and again you are presented with the dialog box to enter your passphrase.

> Note: you end up doing this a lot with PGP and there are tools that will cache the passphrase for you, but please remember that if its cached, and someone uses your machine, they could sign your name. So please keep security in mind when you make a choice like that.

Once you enter your passphrase, you will see it put the signed text back into the current window. If you end up with signed text somewhere else, or an error box, then you didn't have your cursor in the main text window. Now hit "Send" and dispatch the message..

Now check your mail, and hopefully you have a new message from yourself. Open it up and you can see the signed text, but how do you know it hasn't been tampered with? With that message open and in front, go up to the PGPtray again and choose *Current Window, Decrypt & Verify*. A text box should pop up showing you the status of the verification. You should see something like this:

```
*** PGP SIGNATURE VERIFICATION ***
*** Status:   Good Signature
*** Signer:   test user <user@user.org> (0x673ADD59)
*** Signed:   2/23/2003 4:37:46 PM
*** Verified: 2/23/2003 4:46:22 PM
*** BEGIN PGP VERIFIED MESSAGE ***
```

This tells us that the message passed verification, who the signer was, when it was signed and when it was verified. That's a great deal of information. Had it not passed you would have seen:

```
*** PGP SIGNATURE VERIFICATION ***
*** Status:   Bad Signature
*** Alert:    Signature did not verify.  Message has been altered.
```

That is a very easy way for you or someone else to tell if a message or file has been altered. There is no way to fake the signature and the signing.

Encryption instead of signing

Instead of just signing, you might want the message to be encrypted. That's just as easy as signing, except that this time you select *Encrypt* instead of S*ign*. You also need to choose the recipients that you are encrypting to so you can encrypt the there public keys. Also note that your key is already in the recipients by

default. You can turn that off, but it's not a bad idea to leave it alone. You can also choose to use conventional encryption.

What's the difference? Conventional encryption allows you to encrypt the message to no one and just use a password. That way, you could send it to a number of people, you would not need their public keys, just tell them the password (not in the same mail message) and they could open the message. This is more useful in the form of public forums or newsgroups where you only want a group of people that already know the password to be able to read the messages.

There are some options on the left side of the screen. Most of the time, you don't need to change them, but you can to suite your needs. More info about those options can be found in the help screen. Once you hit "OK", you are again asked to input your passphrase.

The procedure for checking is the same and the results are shown in the same way. The only problem with this is if the person that your e-mail client says sent the mail really didn't. Consider that anyone can fake or forge a senders name and even a sending or return address (to a point). Most clients don't show the full e-mail address, just the sender's name. How do you trust that the person you receive mail from, even if it's encrypted is really who its supposed to be from? Anyone can send you an encrypted message to your public key, and they don't have to sign it or say who they are. This is why digital signatures are so important. If you want a person to verify who you are, you still need to sign the message before sending it. And the best way to do that is to sign first, then encrypt.

PGP and files

Another good use of PGP is to encrypt personal files. Things like old IRS information, banking files, personal letters, password lists (if you must keep them in a list, at least encrypt them) and the like. Encrypting files is also a good protective way of sending them over e-mail or on a web site securely. You can encrypt to a person or use a common password that you give to a group of people so only those people can look at it. In fact, some new versions of PGP include a self-extractor that doesn't even require the recipient to have PGP installed; they just need to know the common password.

PGP products from PGP, NAI and other sources tend to put a right-click context menu that enables you to encrypt the file from the file manager. Using explorer (or whatever your file manager is) find the file that you saved that had your public and private key. Right click on the file and go to the PGP menu. Notice however that there is no option to encrypt the file. That is because the extension ".asc" is registered with PGP and it won't encrypt its own files that way. You can, however, launch the PGPmail bar form the PGPtray and select the encrypt

button (the envelope with the lock on it) and encrypt it that way. Another method would be to rename the file, and that's not a problem either, since its just text. Whichever way you choose, once you choose to encrypt you should get a prompt that lets you choose the recipients of the file or to use conventional encryption. Make your choices and hit Ok and go back to the file. Now unless you selected "Wipe Original" you should see the file you started with and the new encrypted file with the same name, but with a ".pgp" extension. That is your encrypted file.

To decrypt this file you can double click on it and enter the passphrase at a dialog box. It's that simple.

For those with a heightened sense of security, there is a Wipe option. This tool will wipe a file by writing over the file multiple passes. It's a good way to remove that file and not ever be able to get it back. You can access the Wipe feature from the PGPmail menu bar. You can also wipe the freespace of a drive. This is not a bad idea if you're a system builder or giving a drive to someone else or ensuring old floppies are erased. It's not the fastest thing, but then again, security never is.

Conclusion

As you can see, PGP is a relatively simple program and provides an easy method of keeping things private when you need to and ensuring that when you send information, that information is not readable or at least has not been changed. The tools are almost always free, easy to use and work with most hardware and software in use today. This paper has shown how to get PGP, protect files on your drive, protect your e-mail messages and manipulate your key ring. There is a large amount of information about PGP and encryption available on the web and in books should you want to learn more.

For further reading and information

For a mathematical explanation of how PGP works, please see:
http://www.giac.org/practical/GSEC/Karen_Coe_GSEC.pdf - by Karen Coe.

A well written PGP for windows FAQ with lots of pointers and help:
`http://www.mccune.cc/PGPpage2.htm#Freeware` - by Tom McCune's

The CryptoRights Foundation – Info PGP, GPG and OpenPGP
`http://cryptorights.org/`

List of References

[1] PGP Corporation - PGP History
http://www.pgp.com/display.php?pageID=49

[2] Ronald Bailey, 1994 - Code Blues
http://www.banned-books.com/truth-seeker/1994archive/121_3/ts213c.html

[3] PGP International Webpage - List of supported OS's
http://www.pgpi.org/products/pgp/versions/freeware/

[4] The Internet Society, 1998 - OpenPGP RCF 2440 -
http://www.ietf.org/rfc/rfc2440.txt

[5] SecurStar GmbH, 1999 - PGP DH vs. RSA FAQ
http://www.scramdisk.clara.net/pgpfaq.html

[6] Imad's PGP Page – PGP ckt build information
http://freepages.computers.rootsweb.com/~irfaiad/assets/readckt.txt

[7] Bruce Potter, 1999 – Lucky Green on The Security of RSA
http://securitygeeks.shmoo.com/article.php?story=20020325172200563

[8]Keith Parkins, 1997 - PGP – The Web of Trust
http://www.heureka.clara.net/sunrise/pgpweb.htm

[9] Mail Client Capabilities: PGP-related
http://www.rjmarq.org/pgp/mail-clients-pgp.html

Last Updated: July 23rd, 2017

Upcoming Training

SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
Community SANS Omaha SEC401*	Omaha, NE	Aug 14, 2017 - Aug 19, 2017	Community SANS
Community SANS Trenton SEC401	Trenton, NJ	Aug 21, 2017 - Aug 26, 2017	Community SANS
Virginia Beach 2017 - SEC401: Security Essentials Bootcamp Style	Virginia Beach, VA	Aug 21, 2017 - Aug 26, 2017	vLive
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
Community SANS Pasadena SEC401 @ NASA	Pasadena, CA	Aug 23, 2017 - Aug 30, 2017	Community SANS
Mentor Session - SEC401	Minneapolis, MN	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
Mentor Session - SEC401	Edmonton, AB	Sep 06, 2017 - Oct 18, 2017	Mentor
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
Mentor Session - SEC401	Ventura, CA	Sep 11, 2017 - Oct 12, 2017	Mentor
Community SANS Albany SEC401	Albany, NY	Sep 11, 2017 - Sep 16, 2017	Community SANS
Community SANS Columbia SEC401	Columbia, MD	Sep 18, 2017 - Sep 23, 2017	Community SANS
Community SANS Dallas SEC401	Dallas, TX	Sep 18, 2017 - Sep 23, 2017	Community SANS
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Copenhagen 2017	Copenhagen, Denmark	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Boise SEC401	Boise, ID	Sep 25, 2017 - Sep 30, 2017	Community SANS
Baltimore Fall 2017 - SEC401: Security Essentials Bootcamp Style	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
Community SANS New York SEC401	New York, NY	Sep 25, 2017 - Sep 30, 2017	Community SANS
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Sacramento SEC401	Sacramento, CA	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Community SANS Charleston SEC401	Charleston, SC	Oct 02, 2017 - Oct 07, 2017	Community SANS
Mentor Session - SEC401	Arlington, VA	Oct 04, 2017 - Nov 15, 2017	Mentor