Published on *A Contrario* (http://www.acontrario.org)

By *BrianNapoletano*
Created *24 Jul 2008 - 15:41*

Protect your email with GnuPG

The fourth amendment to the U.S. Constitution ostensibly protects individuals from unreasonable search and seizure by the State, but says nothing about protection from others, such as supervisors or tabloid reporters who might wish to access your personal information. Moreover, the legal protections that fourth amendment safeguards offer are often circumvented by the State, sometimes legally under judicial review, and frequently illegally. Digital communications, and email messages in particular, are relatively easy to track and monitor without either party's knowledge. Consequently, any reasonable expectation of privacy depends on safeguards implemented by the end users. Many of the people who use digital media to communicate, however, have little to no knowledge of of digital encryption, and few consider their messages sensitive enough to merit such measures. Nonetheless, instances where a reasonable degree of privacy is helpful do arise, from the need to transfer confidential medical or human subjects data to a particularly sensitive personal message. This article briefly addresses the basic privacy issues, and then offers a general introduction to digital privacy protection, including how to implement useful encryption and authentication utilities provided by GnuPG with Mozilla Thunderbird, Microsoft® Outlook Express™, or Microsoft® Outlook™. Of the different email applications covered, I consider Thunderbird the most well-integrated and straightforward, and I highly recommend it over its commercial competitors. Mozilla Thunderbird and GnuPG both operate on Windows™ and Mac systems, so you don't *need* to run Linux to use them. In spite of the advantages, I recognize that many people are hesitant to venture beyond the realm of proprietary software systems, or prevented from doing so by their IT departments, so I will do my best to provide instructions that work with Microsoft® email systems.

Article Contents [hide]

Introduction

The U.S. Senate gave in to pressure from the Bush administration and from telecommunication industry lobbyists on 9 July, 2008 and approved the amendments to the Foreign Intelligence Surveillance Act (FISA) that were sponsored by Representative Silvestre Reyes of Texas. The FISA amendments that most people dislike are the ones that allow the Attorney General and the Director of National Intelligence to submit 'bundled' requests for authorization to monitor an unspecified number of targets, extend the deadline for approval of pre-emptive surveillance to seven days, and grant immunity from prosecution to industries (and landlords) who illegally supplied records to the NSA before the legislation was enacted or who do so in the future. It's not that opponents

have something to hide from the NSA, but most would include them in their recipients list if they really wanted the NSA to be reading their messages. In an effort to encourage the government to reconsider its surveillance policy, I am offering some simple steps to reduce the ease with which private messages can be intercepted and analyzed. While I haven't the slightest clue how to go about encrypting telephone conversations (although most people who talk to me have said that my conversations are already cryptic enough), I do know that you can encrypt your email messages (and digital documents) quite readily. This probably won't keep the NSA from peering into your private messages[1], but it can keep a nosy supervisor at bay. While GnuPG is relartively easy to use, you do need a basic understanding of certain aspects of encryption to use it wisely. Any digital encryption tool will only work as well as its end user. You also need to exercise some common sense when you're using these privacy tools, as misusing them can put your privacy at a greater risk than no protection at all. While this article will provide you with enough information to successfully implement basic protections, I strongly recommend that you consult some of the references listed at the end for more information on the topic.

Basic concepts

You'll need to understand a few basic ideas in order to use GnuPG effectively and safely. We'll cover some of these basic concepts before going into the details of making it work. If these concepts are already familiar to you, then feel free to skip ahead to the next section. Originally, I was going to try to wrap the terminology into an amusing narrative, but then I remembered how annoying that approach is when you're trying to look something up later. Instead, I have chosen to list the key concepts in alphabetical order, although you may need to skip around a bit, as some terms only make sense when you understand others. At first, I tried to list them from basic to advanced, but I found myself continuously rearranging the list and so gave up on that approach. I've compromised by enclosing terms in the definitions that have their own definitions in <*dfn*> tags and hyperlinking them to their definitions.

Asymmetric Algorithm
: With an asymmetric algorithm, the key used to encrypt the message can be different from the one used to decrypt it. This allows someone who wants to receive encrypted messages to keep one key private (a *private key*), and make a second key available to anyone who might want to send a message (a *public key*). GnuPG uses the private/public key concept.

Cypher
: A cypher is a type of code that conceals the contents of the original message by transforming each character according to a set of rules. Some people spell it 'cipher', but I've always seen it spelled 'cypher'.

Digital Signature
: A digital signature, like a physical signature, is a way of associating your identity with a message. Unlike your physical signature, however, a digital signature cannot be counter-fitted without access to your *private key*. A digital signature is generated via the following steps:

1. Generate a *hash* of the message
2. Encrypt the hash using the private key
3. Attach the encrypted hash to the message

Once the encrypted hash has been attached to the message, it has been digitally signed. Because it was encrypted with your private key, only *your public key* will decrypt it. This means that the recipient will be able to verify that the message came from you and that it has not been intercepted and altered prior to its arrival. An important implication of this feature is *nonrepudiation*, i.e. someone cannot deny sending a message that (s)he signed. In order to falsify your signature, *Mallory* would need access to both your private key and your *passphrase*!

Encryption
: The process of transforming *plaintext* (a legible message or string of data) into *cyphertext* (data that has been altered according to a set of instructions, or algorithm). Its counterpart is *decryption*, which is the process of transforming cyphertext back into plaintext.

Hash
: In this context, it is not something you smoke. Instead, a hash is a mathematical derivation of a message that is computed with a particular algorithm. Any alteration of a message will cause it to produce a completely different hash, so the hash can be used to determine whether a message has been altered.

Keyserver
: A keyserver is a special type of Internet server dedicated to the hosting of *public keys*. GnuPG allows you to upload your public keys to a chosen keyserver directly. Michael Lucas, the author of **PGP & GPG** (see 'Recommended Texts' below), recommends `subkeys.pgp.net`. I personally use `keyserver.ubuntu.com` on port 11371. Your GUI will allow you to select a desired keyserver, or you can specify one on the GnuPG command line. Most of the different keyservers synchronize with each other, so you typically only need to submit your key to one server to distribute it globally.

Mallory
: Mallory is the enemy. (S)he could be Big Brother, The Man, the wormy network administrator who's been hitting on you incessantly, the employer who likes to read all the employee's email messages or the high-profile advertising agency trying to figure out the best way to attack you with promotions for the latest brand of staplers. In short, Mallory is a catch-all term for the sneaky bastards that you're trying to protect your messages from.

Passphrase
: The passphrase adds an additional layer of security to your *private key*. GnuPG encrypts your private key when it stores it, so you need the passphrase to use it. Unlike a password, your passphrase can (and should) be several words long (white spaces are legal), miXed Punctuation, and non-alphanumeric characters (!,",#,$,etc.). "The quick brown fox jumps over the lazy dog" is an example of a very bad passphrase, and some variation of "The descendants of Jopheth: Gomer, Magog, Madai, Javan, Tubal, Meshech, and Tiras. --1Chronicles 1.5, NRSV Anglicized Cross Reference Edition, p. 387" would be an example of a good passphrase, but I just published here, so now it would be a bad phrase to use. The trick to devising a good passphrase is to find something that you can

remember easily (writing down your passphrase is like locking your front door and then hanging the key next to the knocker), but that *Mallory* won't easily guess. Tricks such as mixing languages, substituting symbols for letters and mixing up the spelling all help to keep Mallory guessing.

Private Key
: The private key is the *secret* counterpart to the *public key*. **Never share your private key with anyone! Ever!** Also avoid storing your private key on a public computer. You should make an encrypted backup of your private key (use symmetrical encryption), and store it someplace secure (e.g. a USB drive that you carry with you, a floor safe, a safety deposit box).

Public Key
: The public key is the counterpart to the *private key* that you share online. You can post it on your web site, upload it to a specialized *keyserver* or print it out and hand it out at cocktail parties. If you only want certain parties to have the ability to send you encrypted messages, then provide your public key to only those parties.

Revocation Certificate
: Once you distribute your *public key* (particularly if you send it to a *keyserver*), you're stuck with it. This make sense when you consider how the *asymmetric encryption* system works with a public/private keypair. This means that if you lose access to your *private key* (e.g. you accidentally delete it), other people can send you encrypted messages that you cannot decrypt. Worse, if *Mallory* somehow gains access to your private key and cracks your *passphrase*, (s)he can read all your incoming encrypted messages and can send h(er|is) own messages *with your digital signature*! Therefore, you need some way to revoke your public key should your private key be lost or compromised. The solution is the revocation certificate, which can be used to automatically revoke your public key. You should generate a generic revocation certificate when you generate your key (we'll cover how to do this shortly), and you should keep it someplace safe (consider encrypting it with a *symmetric encryption algorithm*). If Mallory gains access to your revocation certificate, (s)he can revoke your public key on you, forcing you to generate a new keypair (while this is an inconvenience, it is not nearly as bad as the former possibility of Mallory stealing your identity).

Symmetric Algorithm
: This type of algorithm uses the same key for encryption and decryption, such as a substitution *cypher*. This approach only works if the cypher key can be transmitted securely; by definition, anyone who has access to the key can both encrypt and decrypt messages travelling between two parties. The difficulty of sending a symmetric key securely over the Internet led to the popular adoption of the alternative algorithm, the *asymmetric algorithm*.

The Web of Trust

The Web of Trust is a system within the encryption community that allows you to determine whether someone actually is who h(er|is) public key declares (s)he is. You can vouch for someone's identity by attaching your *digital signature* (see above) to h(er|is) public key, and then sending the signed key back to its owner. When you sign a public key, you are declaring that you are absolutely certain that the key you signed

belongs to the person named in the ID, so **never sign a key that you have not verified with its owner *in person*!** When you verify someone's key, you should request an official form of ID (Operator Driver License, Passport, etc.) and verify that the name on the key matches the name on the ID *exactly* (e.g. if the key reads 'Greg', but the ID reads 'Gaylord', then refuse to sign the key unless the key ID is changed to 'Gaylord'). This level of security may seem extreme, but it's necessary to ensure that the Web of Trust remains secure. If this responsibility seems a bit overwhelming, don't worry. You don't need to participate in the Web of Trust to use GnuPG. In fact, I'm not going to say anything more on the topic. You can learn more about keysigning and the Web of Trust, including instructions for keysigning in GnuPG, by consulting the references at the end of this article.

Ingredients

Before you can begin setting things up, you'll need to download the necessary ingredients. If you're using Thunderbid, then all you need is Thunderbird, GnuPG and the Enigmail plugin. If you're using Outlook, then you'll need WinPT (see the software list at the end for an alternative to WinPT), which comes with its own version of GnuPG.

Thunderbird configuration

- Mozilla Thunderbird -- Download page for Linux, Mac and Windows users (alternatively, Ubuntu users can install from the main software repository: `sudo apt-get install thunderbird`)
- Enigmail Plugin -- Download page for Mac, Windows and multiple Linux distros. When you download this, be sure to save the file some place where you'll be able to find it.
- GnuPG -- Download binaries for Mac, Windows and Linux distros, or download source code[2]. The developers recommend that you verify the integrity of the GnuPG application you download before installing it. If you don't know how to do this, then read the instructions provided by the GnuPG developers.

Outlook configuration

- Presumably, you already have Outlook (or Outlook Express) installed. If not, then you should probably install it.
- Gpg4win -- Some clever folks got together and rolled GnuPG, WinPT and GpgOL together into a single package, named Gpg4win. GnuPG can be downloaded without the GUI here. Gpg4win also includes the plugins for Outlook and Outlook Express, though, so you'll need it if you want to send encrypted email

Friendly GUIs

GUIs are available for most platforms that run GnuPG, although they each have their own quirks. While a GUI may help you get started (particularly if you're one of those

people with an irrational phobia of the command-line), you'll probably eventually want to get to know GnuPG on a 'more intimate' level.

- Seahorse -- GNOME front end
- KGPG -- KDE front end
- Cryptophane -- Alternative to WinPT for Windows users. Can be downloaded with GnuPG or separately.
- MacGPG -- Mac front end

Installation

Obviously, the installation instructions depend on the platform you're using. I'll cover Ubuntu Linux and Windows in detail, but I don't know Macs well enough to offer instructions for them. If I find any good tutorials for Mac users, I'll post links to them.

Ubuntu installation

Your installation process is arguably the easiest. Simply type the following at the command line:

```
$ apt-get install thunderbird mozilla-thunderbird-enigmail gnupg2* gnupg-doc
```

* gnupg2 is the package name for GnuPG Version 2, which is the desktop version. Omit the 2 for the standalone version.

Congratulations, everything should now be installed. Move on to the next step, generating key pairs.

Windows installation

The applications you need to install depend on whether you plan to use Thunderbird or Outlook.

Thunderbird + Enigmail

1. First, install GnuPG by executing the file you downloaded. Although GnuPG is command-driven, the installer will provide you with visual instructions. You may want to install Cryptophane or Gpg4win (Windows) or Seahorse (GNOME) or KGPG (KDE) if you would rather not work with GnuPG at the command line.
2. Once you have GnuPG installed, install Thunderbird. This is also done by executing the file you downloaded.
3. Finally, install the Enigmail plug-in. You install this by opening Thunderbird and selecting 'Extensions' from the 'Tools' menu. Once the Extensions screen appears, select the 'Install' button on the left. This will open an Explorer dialogue. Navigate to the place where you stored the Enigmail extension when you downloaded it and select it. Thunderbird will then activate the extension when it restarts.

Outlook (and Express)

The installation of Gpg4win is relatively straightforward. Simply execute the installer that you downloaded and follow the on-screen instructions. You will probably want to keep most of the default settings. If you plan on using Outlook Express, then make sure WinPT is selected. If you plan to use Outlook, then make sure GPGol is selected. GPGee will allow you to encrypt and decrypt files on a local drive.

Outlook and Outlook Express don't allow you to use PGP/MIME protocols, all encryption must be done with inline encoding. Be sure to warn your correspondants of this limitation, or you may start to get annoyed with all the illegible messages you receive (yet another reason to switch over to Thunderbird). You'll also need to adjust some of your Outlook / Outlook Exchange settings to make them compatible with GnuPG:

Outlook Express

1. Select 'Options' from the 'Tools' menu
2. On the 'Send' tab, make sure the option under 'Mail Sending Format' is set to 'Plain Text' so that messages composed in Outlook Express are compatible with inline encoding
3. Make sure that the 'Send Messages Immediately' option is not selected. If you miss-type your passphrase while this option is selected, Express will send your message unsigned and unencrypted. Note that by de-selecting this option, you will need to press the 'Send/Receive' button to send outgoing messages.
4. Before you can go any further with the configuration, you need to generate your keys, which we'll cover in the next section.

Outlook

1. The GPGol plug-in sets more of the options for you than the Outlook Express plugin. All you really need to do is set your mail format to plain text. Do this by selecting 'Options' from the 'Tools' menu, and then selecting the 'Mail Format' tab.
2. Set the Composition options to 'Plain Text'.
3. You should now generate your keys, which we'll cover next

Key generation

Here's where trying to provide a unique set of instructions for each configuration becomes a bit tricky. Each of the different GUIs will have slightly different configurations, and I have neither the time nor the desire to walk through each one individually. Instead, I will cover key generation in GnuPG from the command-line first, so that you'll understand what the options mean. After that, I'll briefly cover the key generation process in WinPT.

Key generation at the command-line

I will assume that Linux users already know how to access the terminal. To access the

command-line in Windows, select 'Run' from the 'Start Menu', and enter '`cmd`' into the dialogue box.

1. Call the command to generate a new keypair[3]:

```
$ gpg2 --gen-key
```

2. GnuPG will ask you to select the algorithm it will use to generate the keypair.DSA and Elgamal are the defaults, and although RSA is more advanced, it cannot be used for encryption in GnuPG (After doing some digging, I learned that you can specify an RSA key for signing and encryption if you add a `--expert` flag to the initial keygen command. I leave the decision of which key type to use up to the user, with the qualifier that RSA is generally considered more 'up-to-date' than DSA). Select `DSA and Elgamal (default)`:

```
Please select what kind of key you want:
   (1) DSA and Elgamal (default)
   (2) DSA (sign only)
   (5) RSA (sign only)
Your selection? 1
```

3. Next, GnuPG will ask you how many bits you want in your ELG key (the DSA part is locked at 1024 bits). More bits do not necessarily translate into a more secure key, and the default value of 2048 should be sufficient for the next several years.

```
DSA keypair will have 1024 bits.
ELG keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048) 2048
Requested keysize is 2048 bits
```

4. Now GnuPG needs to know when your key should expire (i.e. it no longer can be used). Since this is presumably your first key, set it for one year. GnuPG will then tell you exactly when your key will expire, and ask you if that is what you want.

```
Please specify how long the key should be valid.
         0 = key does not expire
        = key expires in n days
      w = key expires in n weeks
      m = key expires in n months
      y = key expires in n years
Key is valid for? (0) 1y
Key expires at Mon 20 Jul 2009 19:41:30 EDT
Is this correct? (y/N) y
```

5. Each key must be associated with a given identity. This identity consists of a name, an email address and additional comments that help to distinguish you from anyone else with the same name. Use your legal name, so that other users can verify your identity against an official ID.

```
You need a user ID to identify your key; the software constructs the user ID
from the Real Name, Comment and E-mail Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Real name: Brian Napoletano
Email address: napzilla@napoletano.net
Comment: Graduate Student, Webmaster, Activist
You selected this USER-ID:
    "Brian Napoletano (Graduate Student, Webmaster, Activist)
     <napzilla@napoletano.net>"
```

```
Change (N)ame, (C)omment, (E)-mail or (O)kay/(Q)uit? o
```

6. After you specify your information, GnuPG will ask you for a passphrase to secure your private key. See the entry above on the passphrase for guidelines on selecting a useful passphrase.
7. Once you enter your passphrase twice, GnuPG will generate your keypair. When it finishes, it will tell you all about your new key.

```
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilise the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilise the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: key 89FAF746 marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   2  signed:   1  trust: 0-, 0q, 0n, 0m, 0f, 2u
gpg: depth: 1  valid:   1  signed:   0  trust: 1-, 0q, 0n, 0m, 0f, 0u
gpg: next trustdb check due at 2009-07-20
pub   1024D/89FAF746 2008-07-20 [expires: 2009-07-20]
      Key fingerprint = 27B6 0D97 2056 132E 5303  A03F 300A 3009 89FA F746
uid          Brian Napoletano (Graduate Student, Webmaster, Activist)
             <napzilla@napoletano.net>
sub   2048g/43F6924F 2008-07-20 [expires: 2009-07-20]
```

   The trust information goes back to the Web of Trust, which I promised not to cover any more. Below that, you'll find the user id and fingerprint for your key. Copy your key fingerprint down somewhere, as it can be used to find your public key on a
8. keyserver.

Key generation in WinPT

Key generation is relatively straightforward in WinPT. Simply select 'New' from the 'Key' menu, and then select either 'Normal' or 'Expert' (I am assuming you're not using a Smartcard). The information provided in this article is probably sufficient to allow you to use the 'Expert' mode, but the decision is yours. Once you make your selection, answer the questions to generate your key.

1. Use your legal name, so that others may verify your identity with an official ID.
2. Enter keywords that help distinguish you from others with the same name in the comment section (e.g. Student, Chemist, Mad Scientist)
3. The email address you enter is the one which you will be sending encrypted messages from and receiving them at.
4. Because this is your first key, you should have it expire in one year. You can always change this later if you become more comfortable with GnuPG.
5. You can decide whether you want to use DSA/Elgamal or RSA keys for encryption and decryption. DSA/Elgamal is typically the default, but RSA is generally considered more advanced. In either case, you should select a key size of 2048 bits (the DSA component of the DSA/Elgamal key will be locked at 1024 bits, which

is fine--just set the ELG part to 2048).
6. Consult the passphrase definition above for guidelines on selecting a secure passphrase.

Once you have finished generating your keypair, you should create a backup copy and a revocation certificate. We'll cover those procedures next.

Key backup and revocation

You want to keep a copy of your revocation certificate around so that you can revoke your public key if something happens to your private key (e.g. you lose it or Mallory gets hold of it). You should also generate a backup of your keys, so that you are less likely to lose them. Of course, such a strategy is most effective if you keep the backup someplace safe, like on a USB drive that you carry with you or in a safety deposit box. I just discovered that WinPT automatically asks you to back up your keys right after you generate them. This puts those of you using WinPT one step ahead of everyone else, so you can skip down to the second step, which tells you how to encrypt your key backup.

Backing up your keyring

In GnuPG, you generate a backup of your keyring by exporting it to a file that you should encrypt with a symmetric algorithm.

1. Export the keyring to a gpg binary file with the `--export` flag. Use the `--output` flag to specify a file to which the key should be written.

   ```
   $ gpg --output backupkey.napzilla@napoletano.net.bak
     --export-secret-keys napzilla@napoletano.net
   ```

   You can specify the key to be exported by using any information unique to that key. If the email address is only assigned to that key, then you can use the email address. Alternatively, you can use the key ID (the last 8 digits of the key's fingerprint, sans spaces).
2. Next, you should encrypt your backup key with a symmetric cypher.

   ```
   $ gpg --symmetric backupkey.napzilla@napoletano.net.bak
   ```

   When you enter the above command, GnuPG will ask you for a passphrase. This is the passphrase that you will use to encrypt and decrypt your backup key, not the passphrase you assigned earlier. Make sure that the passphrase is something that you will be able to remember.

Generating revocation certificates

Deleting a private key does not automatically shut down your public key, which may have been distributed across the Internet already. Instead, you need to revoke both your private and your public keys with a revocation certificate. You can easily generate such a certificate if you have access to your private key, but what if you lose the private key? In such an instance, you will be glad that you took the time to generate a revocation

certificate when you first generated your keypair. Generating such a certificate is relatively simple in GnuPG. Simply use the `--gen-revoke` in conjunction with the `--output` flags. You may also want GnuPG to output the certificate in ASCII format, so that you can read it. Simply add the `-a` flag:

```
$ gpg -a --output revcert.napzilla@napoletano.net.asc.revoke
  --gen-revoke napzilla@napoletano.net
```

As with the previous instance, the text after the `--output` flag is the name of the file you're writing to, and the text after the `--gen-revoke` flag is used to identify the key for which the certificate is to be generated. GnuPG will ask you to select a reason for the revocation, and you should select '`0 No reason specified`'. The program will then allow you to enter an explanation, which you can phrase as something to the effect of "Generic revocation certificate generated at time of key generation".

WinPT allows you to generate a revocation certificate by selecting 'Revoke Cert' from the 'Key' menu. Select '`0 No reason specified`' for your reason, and enter something to the effect of "Generic revocation certificate generated at time of key generation" in the 'Optional description text' space. Then, enter your passphrase and specify an output file for your certificate.

Setting your default email key

Once you have finished generating and backing up your keys, you can specify your new key as your default key for email encryption. I am assuming that you have already set up your email account to send and receive with Thunderbird or one of the Outlooks. If you do not know how to do this, then consult the email client's help documentation, as well as any documentation offered by your email server. Be warned that Yahoo! does not offer POP access on its free email accounts, but Gmail does (you need POP access to run your email account through a client like Thunderbird or Outlook). Once you have your email account configured, you're ready to configure your encryption options.

Thunderbird Configuration

In Thunderbird, you do this by navigating to the email account that you want to associate with your GnuPG key, and selecting 'View settings for this account'. Select the 'OpenPGP security' option, and check the box to enable OpenPGP support for that identity. Then, you can have Enigmail identify the key to use by searching for the key with the matching email address, or you can specify a key from your keyring (if you only have one key, then either option will probably work). You can also specify whether you want your messages to be encrypted, signed or both by default. If you want people who are using a Microsoft® email client to be able to receive signed or encrypted messages, then you should not check the 'Always use PGP/MIME' box. Finally, you can specify whether or not you want your public key ID to appear in the header, and you can specify a URL from which message recipients can download your public key.

Outlook (& Express) Configuration

Unfortunately, I was unable to experiment with the encryption configuration in Outlook or Outlook Express, as I do not have access to a machine with a working version of either client. I have collected the following instructions from the documentation for Gnupg4win and from the text of **PGP & GPG**, by Michael Lucas.

Outlook Express
I did not see any instructions for configuring default keys in Outlook Express. According to the documentation, two new buttons should appear on your email composition screen after you install Gnupg4win. One button is labelled 'Sign', and the other is 'Encrypt'. Outlook Express will then ask you to identify the key it should use (if you have more than one) and to enter your passphrase when you press the 'Send' button. Remember that you should have told Outlook Express *not* to send messages immediately, so you will need to press the 'Send/Receive' button on the main screen to actually deliver your message.

Outlook
Outlook uses a plugin called 'GPGol'. The Gnupg4win documentation includes a section with recommendations for using this plugin, and you should read these recommendations if you plan on using Outlook. Once Gnupg4win is installed, you can reach the GnuPG configuration menu by selecting 'Options' from the 'Tools' menu. As with Thunderbird and Enigmail, this menu should allow you to specify a default key, and whether or not you want your messages signed, encrypted or both by default.

Sending encrypted messages

Any of the email plugins should cause a button that enables encryption and signing to appear when you're composing emails. Encrypting a message means that only the people who's public keys are listed will be able to decrypt the message (this includes you--if you don't list yourself as a recipient, then you won't be able to decrypt your own message). Signing a message attaches a brief hash to the email that can be checked by the recipient to verify that the message came from you and that it hasn't been altered. Because people without GnuPG can still read signed messages, you may wish to always sign your emails, regardless of whether or not you encrypt them.

Also, note that encrypting an email message *does not encrypt the subject line*. Going through all the trouble of encrypting an email containing a secret recipe is rather pointless if you give it a subject like "The secret ingredient is formaldehyde". Even indicating that the message contains a secret recipe may cause Mallory to give it more attention than you'd like. Instead, either leave the subject line blank, or give it an innocuous title, such as "The instructions you requested" or "What are you doing this weekend?". This highlights a more important point, which is that good encryption is not a substitute for common sense. If you encrypt and sign an email to a co-worker complaining about your boss, that co-worker can easily decrypt the message and pass it along to the aforementioned boss. Worse, the fact that you signed the message means that it couldn't have been forged by your co-worker! If you use common sense and encryption together, then you're much more likely to protect your privacy and your job than if you use just one of the two.

Distributing your public key

Now that you have your own keypair and you're ready to begin sending secure emails, you need to distribute your public key so that other people can send you secure messages. As I mentioned previously, you can distribute your public key any way you see fit. To help you do this, I will show you how to export your public key and how to send your key to a keyserver in GnuPG. GUIs like WinPT also offer options to facilitate these processes, so I'll also describe how to do it with WinPT.

Exporting and publishing keys with GnuPG

Exporting your public key in GnuPG is very much like exporting your private key. Instead of the `--export-private-keys` flag, you simply use the `--export` flag (once again, add the `-a` flag to make your public key legible in plain text form).

```
$ gpg -a --output napzilla@napoletano.net.public.asc
  --export napzilla@napoletano.net
```

Sending your key to a keyserver is also fairly straightforward with GnuPG. Specify the name of the key to be sent after the `--send-keys` flag, and specify the server you wish to send your key to after the `--keyserver` flag.

```
$ gpg --send-keys napzilla@napoletano.net --keyserver subkeys.pgp.net
```

In this instance, I instructed GnuPG to send the key with napzilla@napoletano.net in the address to the subkeys.pgp.net server (this is the server that Michael Lucas recommends; I personally prefer keyserver.ubuntu.com).

Exporting and publishing keys with WinPT

You can export your public key simply by selecting the key on the main screen and then pressing the 'Export key to a file' button (the button with the picture of a floppy disc on it). You should probably change the directory that you're saving to, as the default path is inside the directory that you installed Gnupg4win in.

Publishing your key to a keyserver is almost as easy as exporting it. 'Right-click' on your key, and move the mouse down to 'Send to keyserver'. Then, select the desired keyserver from the list and send your key.

Remember, the keyservers only represent one of many media through which you can distribute your public key. If you have a particular friend that you want to communicate with, you can send h(er|im) an email message with a copy of your public key attached, and request that (s)he do the same. You can also print or write down your key's fingerprint and give your friend a copy of it. (S)he can then use that fingerprint to find your key on a keyserver.

Importing public keys

You are almost ready to begin communicating securely via email! You only need to learn one more minor function: importing other peoples' public keys. You can import public keys from a keyserver simply by searching for and downloading them. If you have downloaded a key from an alternative source (e.g. an email message or a web page), then you'll need to tell GnuPG to physically import the key. Here's how you do this with GnuPG and with WinPT:

Importing keys with GnuPG

To import a key from a keyserver, you simply use the `--recv-keys` flag. For instance, if you want to import my actual public key (not the fake one I've been using as an example, but the one I honestly use), then you would enter the following:

```
$ gpg --recv-keys C5B7DC56
```

GnuPG will then query the server for the key with the ID that matches the one specified after the flag. Obviously, this only works if you have the key ID. If you're working with more basic information, such as a name, you would use the `--search-keys` flag:

```
$ gpg --search-keys 'brian napoletano'
```

If you enter this particular line, you'll probably find quite a few extraneous keys on the server. I made a few mistakes while I was learning how to use GnuPG, and my revocations are taking a while to filter through the system. Make sure you enclose the name you're looking for in quotes, or you will confuse GnuPG (it will think that the name after the second space is another option, and last I checked, there was no 'napoletano' option).

If you already have a copy of the key you wish to import on your computer, then you do not need to bother searching the keyserver for it. Instead, use the `--import` flag to import a local file. I have attached a copy of my public key to this article, and you are welcome to download it and to import it into your keyring. After you download the key, open a terminal interface ('Run' 'cmd' for you Windows users) and go to the directory that you saved the key in. Then use the following command to import the key into GnuPG:

```
$ gpg --import bmn.personal_pubkey.asc_.txt
```

I don't know why Drupal insisted on using such a funky filename for the key. Apparently, bmn.personal_pubkey.asc wasn't a valid filename, so it decided to make it more confusing. In any case, GnuPG should add my public key to your keyring after you issue the above command.

Importing keys with WinPT

Fetching keys from a keyserver is even easier in WinPT than in GnuPG. Simply select 'Keyserver' from the main menu bar, and WinPT will open a dialogue box with a list of keyservers and a box to enter a key ID or an email address to search for. Recall that the key ID is the last eight digits of the key fingerprint. Once WinPT returns a list of keys,

select the one you want and press the 'Receive' button. If you would like to search an additional keyserver, 'right-click' on the list of keyservers and select 'Add'.

Importing local keys is also relatively simple. Select 'Import' from the "Key' menu, and then direct WinPT to the location where the key you wish to import is stored. Once you find the key, press the 'Open' button, and WinPT will import it into your local keyring!

What I didn't cover

I chose to omit a few major topics because they weren't necessary to my central objective, which was to provide you with simple, straightforward instructions for securing your email. I do think that you should take the time to investigate some of these topics when you have a moment. These topics include:

- *Encrypting files with your own or someone else's public key*-- You can do this with GnuPG using the `--encrypt` flag. Be sure to add yourself as a recipient if you want to be able to decrypt the file. Some GUIs, such as WinPT, install with a feature that adds an 'Encrypt' option to the context menu (the menu that appears when you 'right-click' on a file in Windows Explorer).
- *Subkeys*-- You can associate more than one email address (or identity) with a single key by generating subkeys. In GnuPG, you access this feature from the key editing menu, which you reach with the `--edit-key` flag. You can also add a photograph to your key in the edit menu.
- *Signing public keys*-- This is part of the Web of Trust, which I said I would not cover. Key signing is a way to verify that a person is who (s)he claims (s)he is in h(er|is) public key. You should only sign a key after verifying the keyholder's identity *in person* with an official form of ID. Once you sign a key, you have to export it and send it to its owner if (s)he is to accrue any benefit from your signature.
- *Disk encryption*-- In addition to individual files, you can encrypt entire disks or disk partitions for added security. This is a bit much for GnuPG to handle, though, so you will probably want to examine alternative programs to handle this (free software can do this).

Conclusion

Congratulations! If you have read this entire article, then you are now ready to send and receive private email messages. My hope is that making this information available will help to engender more respect for online privacy. As our culture becomes increasingly 'digitized', more of each person's private life will become accessible via the computer. I, personally, would like to see as many of those lives as possible remain private, and I don't believe that anyone should be allowed to invade that privacy without good cause and a legal warrant. It is precisely when our society faces threats such as those incurred by terrorism that we need to be the most vigilant about defending our sacred values, because times like that are when they are the most likely to be sacrificed in the name of 'state security'.

Recommended Resources

General Reading

- An Overview of Cryptography by Gary C. Kessler
- Bruce Schneier is the author of, among other works, **Applied Cryptography** and **Beyond Fear**
- Cryptography and Topics in cryptography articles in the Wikipedia
- Cryptography article by Electronic Frontiers Australia
- Encryption Archive by the Electronic Frontier Foundation
- The GNU Privacy Handbook -- *If you only visit one other resource, make it this one*
- Surveillance Self-Defense by the Electronic Frontier Foundation

Linux-Specific Help

- How to Use Gnu Privacy Guard-- Ubuntu Community Documentation
- Privacy and Encryption with PGP : Signing and Encrypting Email / Files-- Ubuntu Tutorial

Windows-Specific Help

Please provide feedback as to whether or not these links helped or links that did help, so that I can improve the resources for other Windows users.

- How to Encrypt Your Email with GnuPG in Windows Part 1 - WinPT-- Posted at TechMalaya.com
- How To: GnuPG (gpg4win) for MS Office Outlook, Exchange, and Others

Mac-Specific Help

Please provide feedback as to whether or not these links helped or links that did help, so that I can improve the resources for other Mac users.

- Using GnuPG encyption with Mac OS X Mail-- Posted at "A Thousand Cups of Tea"
- Configuring GnuPG (Mac OS X)-- Posted at zeitform

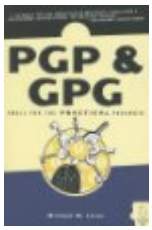
Software Resources

- Enigmail
- The GNU Privacy Guard
- Gpg4win -- An alternative to WinPT that includes Outlook plugins
- OpenPGP Alliance
- Thunderbird
- Windows Privacy Tools (WinPT)

FISA Reading

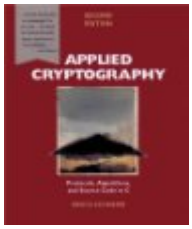
- The text and summary of the FISA legislation
- Role call of U.S. Senate votes on H.R. 6304
- The Case Against Retroactive Amnesty for Telecoms (Electronic Frontier Foundation)
- Senate Approves Bill to Broaden Wiretap Powers (New York Times)
- Senate approves FISA makeover and telco wiretap immunity (The Register)
- Senate OKs FISA Bill, Immunity For Telecom Firms (National Public Radio)
- ISPs to record all emails and calls -- This was in the Guardian on 6 April 2009, and applies to the UK

Recommended Texts



PGP & GPG
Michael W Lucas
Best Price $5.47
or Buy New $18.21
Buy from amazon.com
Privacy Information

I purchased this book from Amazon and read it in a day. It's very concise, easy to read and it provides enough information to get you up and running with GnuPG (or OpenPGP). This book is not too expensive, and I highly recommend it if you're new to the world of cryptography (like me). Much of the information I provide here came from this book.



Applied Cryptography
Bruce Schneier
Best Price $14.25
or Buy New $33.12
Buy from amazon.com
Privacy Information

I recently purchased this text on the recommendation of a close friend and colleague, Jonah Duckles. I now understand why he told me that it is the requisite text for anyone interested in exploring cryptography. In addition to covering the concepts, mathematics and pragmatic applications in detail, this wonderful resource provides the C code to implement many of the algorithms covered in the text.

Footnotes

1. Although GnuPG is rumored to have protect the lives of rebel fighters in some places I have never heard of this being verified.

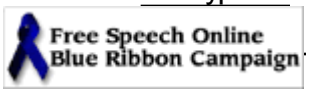
2. GnuPG, by itself, is a command line driven application. However, since many people are still intimidated by the terminal, I'll cover some of the common GUIs. If you'd rather use the

command-line, you can find instructions later in this article or on the GnuPG manual page. Some users may also notice that GnuPG comes in two versions. Version 2 is actually the desktop version (i.e. it relies on external libraries), whereas version 1 is a standalone version. If you're on a desktop, the designers recommend version 2.

3. The $ symbol in this context refers to the prompt at the command-line. In Windows, it would be a > symbol instead.

Attachment	Size
bmn.personal_pubkey.asc_.txt	133.43 KB

encryption GnuPG privacy Computing



Join the Blue Ribbon Online Free Speech Campaign!

09 F9 11 02 9D 74 E3 5B D8 41 56 C5 63 56 88 C0

Source URL (retrieved on *23 Oct 2009 - 17:52*): http://www.acontrario.org/node/352