



OPENPGP EMAIL SECURITY
FOR MOZILLA APPLICATIONS

The Handbook

V 1.8

Adaptations by Ludwig Hügelschäfer
Based on Version 1 by Daniele Raffo
with Patrick Brunschwig
and Robert J. Hansen.

Enigmail Handbook Contents

1. Setup and usage	5
1.1. What do you need?.....	5
1.2. How to install Thunderbird.....	5
1.3. How do I install Enigmail?.....	6
2. Quick start guide	8
2.1. The Setup Wizard.....	9
2.2. Start using Thunderbird with Enigmail.....	28
3. Key management	30
3.1. Operations on your key pair.....	31
3.2. Distributing your public key.....	35
3.3. Importing public keys.....	36
3.4. Validity of public keys.....	37
3.5. Importing an existing key pair.....	42
3.6. Generating your own key pair.....	42
3.7. Revoking your key pair.....	46
4. Signature and encryption	47
4.1. Account settings.....	47
4.2. Signature and verification.....	50
4.3. Encryption and Decryption.....	59
4.4. Handling attachments.....	63
4.5. Practice with “Adele”.....	64
4.6. Permanent Decryption of messages.....	68
4.7. Encrypted Subject?.....	71
5. Preferences	72
5.1. Setting the preferences.....	72
5.2. Per-recipient rules.....	81
6. Troubleshooting	85
7. FAQ	89
8. Notes, Tips & Tricks	93
8.1. How to choose a good passphrase.....	93
8.2. Protection of the local computer.....	95
8.3. Keeping your key pair in a safer place.....	97
8.4. Passphrase Handling.....	100
8.5. Key verification procedure.....	102
9. Advanced operations	103
9.1. Manually installing GnuPG.....	103
9.2. Manually editing the preferences.....	105
9.3. XML format of per-recipient rules.....	116
9.4. Available languages.....	117
10. Acknowledgements	119
11. The Enigmail team	120
12. Support	121
13. History	122

WHAT IS THIS ALL ABOUT?

Enigmail is an extension for Thunderbird and other Mozilla based mail clients. It allows you to encrypt and digital sign emails using the OpenPGP standard.

This handbook covers all aspects of using Enigmail. It consists of two parts. The first part will show you how to setup and use Enigmail, the second part will answer frequent questions, provide historical background, technical details and references, as well as pointers to support pages. During reading the first part, you will encounter references to the second part. Reading the referenced paragraph is not necessary for understanding. You can do this later, e.g. for getting background information.

Parts of this handbook are not (and can not be) written in a linear way, and while reading you may want to jump between sections, chapters and pages. As far as possible we have placed clickable cross references. Use them!

Most of the screenshots show Enigmail 1.8 on Thunderbird 31 on Mac OS X. If your installation is different concerning mail client, or operating system, there will be minor differences in the visual appearance but functionalities will be similar if not the same.

HOW DOES THIS ENCRYPTION WORK?

There are two main branches of cryptography: symmetric cryptography and asymmetric cryptography.

Symmetric cryptography is the first type of cryptography invented, dating back to 2000 years ago, and the only one most people know. In symmetric cryptography, a *cipher* (cryptographic algorithm) is used in conjunction with a single *key*, for instance a password, to encrypt a message. The message can then be decrypted using the same key.

Symmetric cryptography poses a problem concerning the delivery of secure messages. The sender can encrypt a message and send it to the recipient, but has to provide the recipient the key to decrypt it. The key cannot obviously be sent with the message, and must be communicated through a secure channel. Encryption provides a secure channel for the delivery of messages but, in order to make it usable, the sender must first deliver the key to the recipient.

This catch-22 problem was solved only thirty years ago with the birth of *asymmetric cryptography*, also called *public key cryptography*. See paragraph 13 at page 122 to read more about the history.

Public key cryptography is much more interesting and useful. It does not operate with a single key but with a *key pair*, composed of a *public key* and a *private key* (also called *secret key*). Public and secret key are created together at the same time using a special algorithm.

Let's show how public cryptography works by taking as an example two people, Alice and Bob, that want to exchange secure messages.

Alice generates her own key pair in advance. Then she makes the public key available to anyone, for instance by publishing the key in a public directory, and carefully keeps for herself the secret key. This is perfectly safe, because it is practically impossible (or, as computer scientists prefer to say, *computationally infeasible*) to derive a private key from its companion public key alone. Bob does the same: generates a key pair, publishes his public key and keeps undisclosed his secret key.

When Bob wants to send a confidential message to Alice, he first retrieves Alice's public key from the directory. Then he encrypts the message with her public key and sends the message. Alice decrypts the message with her private key and is able to read it.

Public key cryptography is not only employed for confidentiality (ensure that the message can be read only by the intended recipient), but also for authentication (ensure that the message really comes from the intended sender) and integrity (ensure that the message has not been altered in transit). Authentication and integrity are enforced by appending a *digital signature* to the message. A digital signature is generated by an algorithm that uses a *hash function* in conjunction with a key. A hash function is a function that takes in input a message of any length, and outputs a string of fixed small length called *digest* which is a distillate of the message fed in input. Notable features of hash functions include that is practically impossible to derive the input from the output, and that changing just one bit of the input results in a completely different output.

Hence Bob writes the message, generates the digital signature for the message using a predetermined hash function and his private key, appends the signature to the message, and sends to Alice the whole lot. Alice receives the message and verifies the signature using the same hash function and Bob's public key. If the signature is valid, then the sender is authenticated, because only the owner of the private key, Bob, could have signed the message. This guarantees also the integrity of the message, because had the message been altered in transit, it would resolve to a different digest and the signature would not match.

1. SETUP AND USAGE

1.1. What do you need?

This chapter illustrates how to get Enigmail up and running.

As a first step, if you don't already use it, you need Mozilla Thunderbird.

Note: For better readability, we always refer to “Thunderbird” in our documentation, although technically correct would be “a Mozilla based mail client such as Thunderbird, SeaMonkey or Icedove”. Instructions for Thunderbird should apply to all derivatives based on recent Mozilla code.

Second, you need the Enigmail extension. How to install these two will be covered by the next two paragraphs.

As a third and fundamental requirement you need GnuPG (Gnu Privacy Guard) which will be used to perform all cryptographic operations. If it is not installed on your system (or Enigmail cannot access it), the setup wizard will offer to download and install it for you.

1.2. How to install Thunderbird

You can download Thunderbird from <https://www.mozilla.org/en-US/thunderbird/>

Many Linux distributions ship with their own customized version of Thunderbird. If you use your distribution's version of Thunderbird, you should use your distribution's version of Enigmail. The package manager of your distribution takes care of the installation and you can skip the rest of this paragraph and proceed directly with the Quick start guide at page 8.

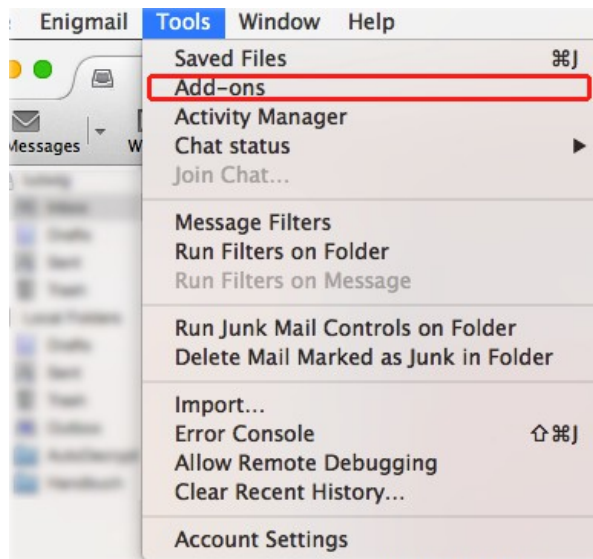
If you get Thunderbird from the mozilla site and install it yourself, though, you can use the official Enigmail releases provided by the Enigmail Project.

Thunderbird has their excellent documentation and user group support, hence we'll skip the steps to install and configure them.

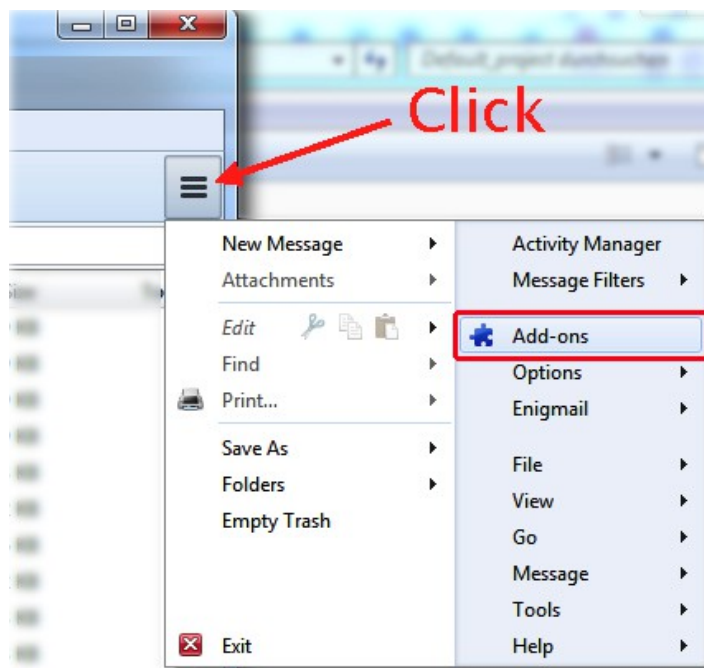
In every case you should have Thunderbird and your email account fully configured before proceeding to the installation of Enigmail.

1.3. How do I install Enigmail?

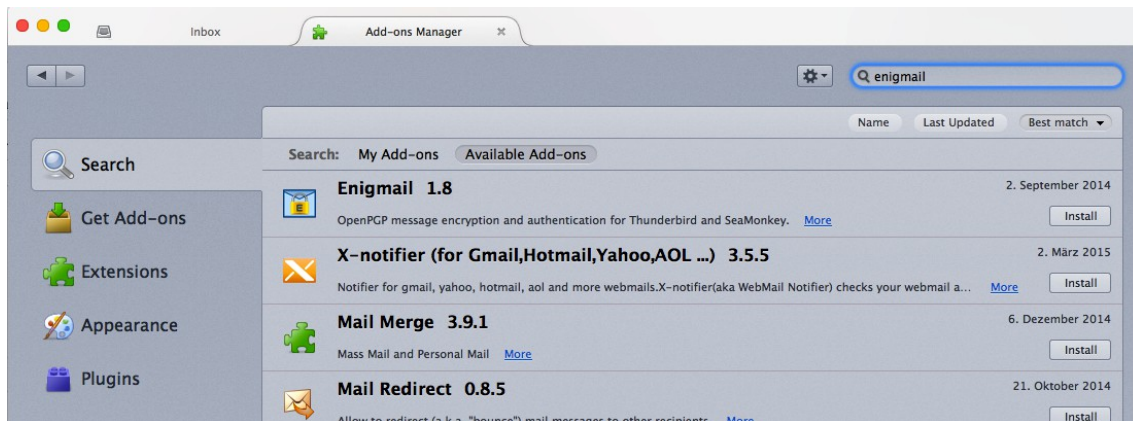
Enigmail is a “Mozilla extension”. Extension can be installed (and removed) through Thunderbird’s “Add-ons-Manager”. Start it like shown:



Note: The Thunderbird menu may be quite difficult to reach on Windows using the standard layout. Click on the shown icon to open the menu and select “Add-ons”:



Enigmail can be selected (“Featured Extensions”) or searched and installed:



You will get a confirmation dialog, to not download software from unknown sources. However, the Mozilla site can be trusted and you can confirm.

After installing the extension, you must restart Thunderbird to activate it.

Linux users: If you install all your software packages from a repository, you may find that it contains an older (but security-patched) Thunderbird version. Please use the (matching) Enigmail package from that repository.

1.3.1. Manual Install

If Enigmail is not shown in the Extensions list in when searching for it in the Add-ons-Manager, then either Enigmail is already installed or your Thunderbird version or platform may not be officially supported (= available through <https://addons.mozilla.org/thunderbird/addon/71>).

In such case you may use your favorite browser to search for a user contributed build suitable on <https://www.enigmail.net/download/>.

There are also current development snapshots available from <http://www.enigmail.net/download/nightly.php> that you may test drive. Warning: Depending on the state of development, these builds may be unstable!

To install a downloaded XPI file, drag-and-drop it to the Extensions tab of Thunderbird’s Add-ons-Manager or use its “Install Add-on from file ...”:

2. QUICK START GUIDE

In chapter 1 we covered how to install Enigmail. This chapter assumes that Thunderbird and Enigmail is installed by now. If that is not the case, please head back to chapter 1.

In particular, the Quick Start Guide assumes that

- Thunderbird is installed and
 - all email accounts and identities are set up
 - you can receive and send (unencrypted/unsigned) e-mail
- Enigmail is installed
 - but not configured yet (default values are set)
 - you did not restart Thunderbird since installing Enigmail
- GNU Privacy Guard
 - may or may not be installed yet and
 - keys may already exist in your keyring or not

Even if you already did modify Enigmail settings or have existing keys, you should be able to follow this guide. Adapt for your situation, like rather choosing an existing key instead of creating a new key.

Throughout this “Quick Start Guide” we use the default “Basic” settings. They are intended for new users and perfectly fine for everyday usage.

There also is an “Expert” mode that gives you more options and control. We will enable this at the end of this guide because the other chapters of this handbook also explain those expert settings.

But let’s start now:

Restart Thunderbird – and note the Setup Wizard window. This wizard will guide you through all steps to configure Enigmail and have it ready to use. This is intended for new users.

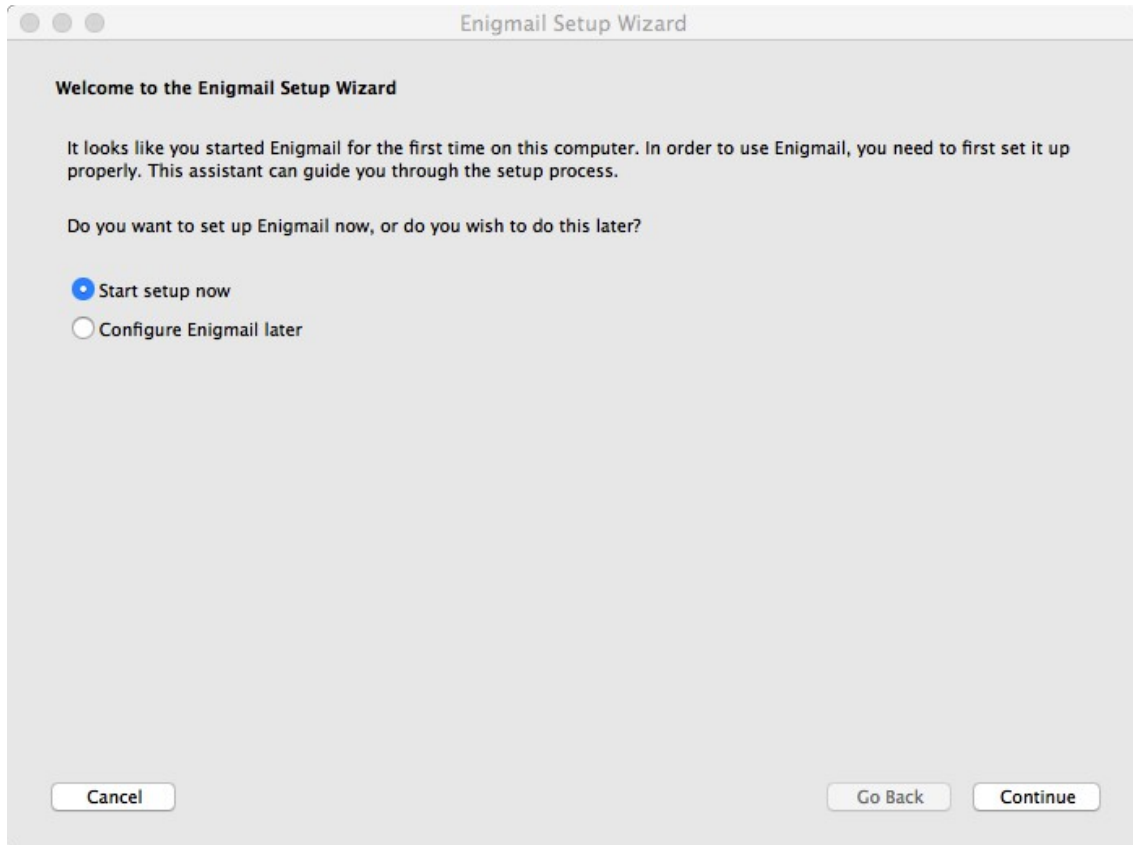
Using the Setup Wizard is not necessary. Experienced users may as well configure Enigmail by hand, which will grant a deeper knowledge of the mechanisms of Enigmail.

The rest of this chapter is a step-by-step guide to the Setup Wizard followed by a very basic explanation of the signing and encryption functions. If you decide not to use the Setup Wizard, you can go directly to the next chapter.

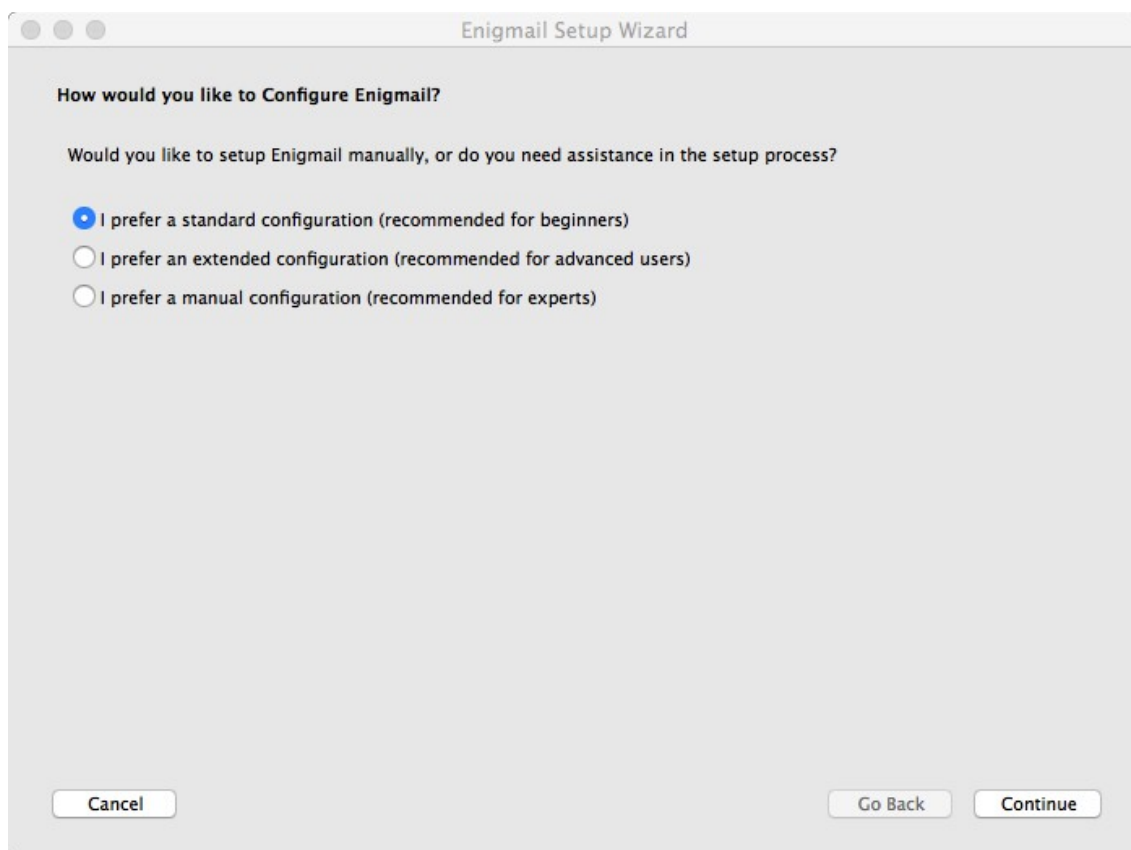
2.1. The Setup Wizard

The Setup Wizard starts automatically when restarting Thunderbird after installing Enigmail. The following dialogs will guide you through a basic setup. You may cancel this wizard any time and (re)start it from the menu.

The first screen asks you whether you want to set up everything using the wizard:



The next screen will show a selection for your experience level:



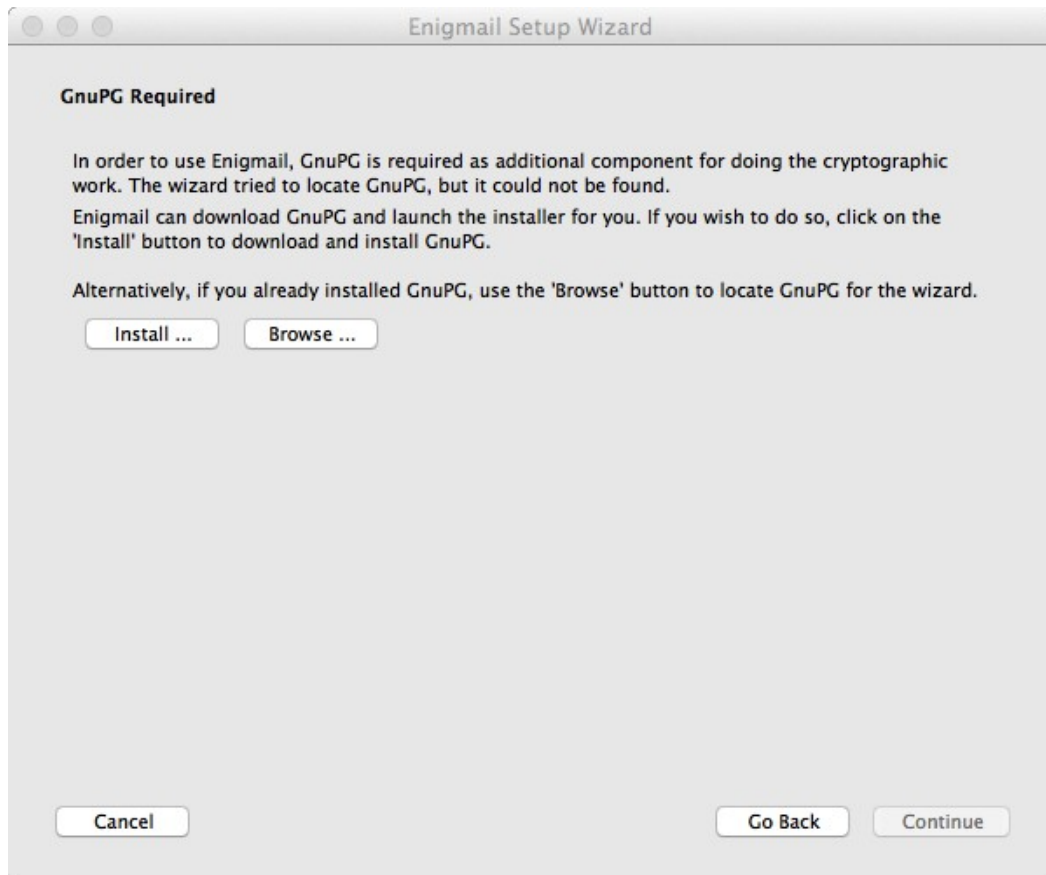
In most cases you will select “I prefer a standard configuration (recommended for beginners)”.

Advanced users, e.g. previous users of OpenPGP encryption may want to select the second option.

Expert users can use the third option. This is intended for users having good experience of Enigmail.

After you have made your choice, click *Continue*.

For every choice, the next step will download and install the Gnu Privacy Guard (GnuPG), if the Enigmail Setup Wizard cannot find it on your computer.



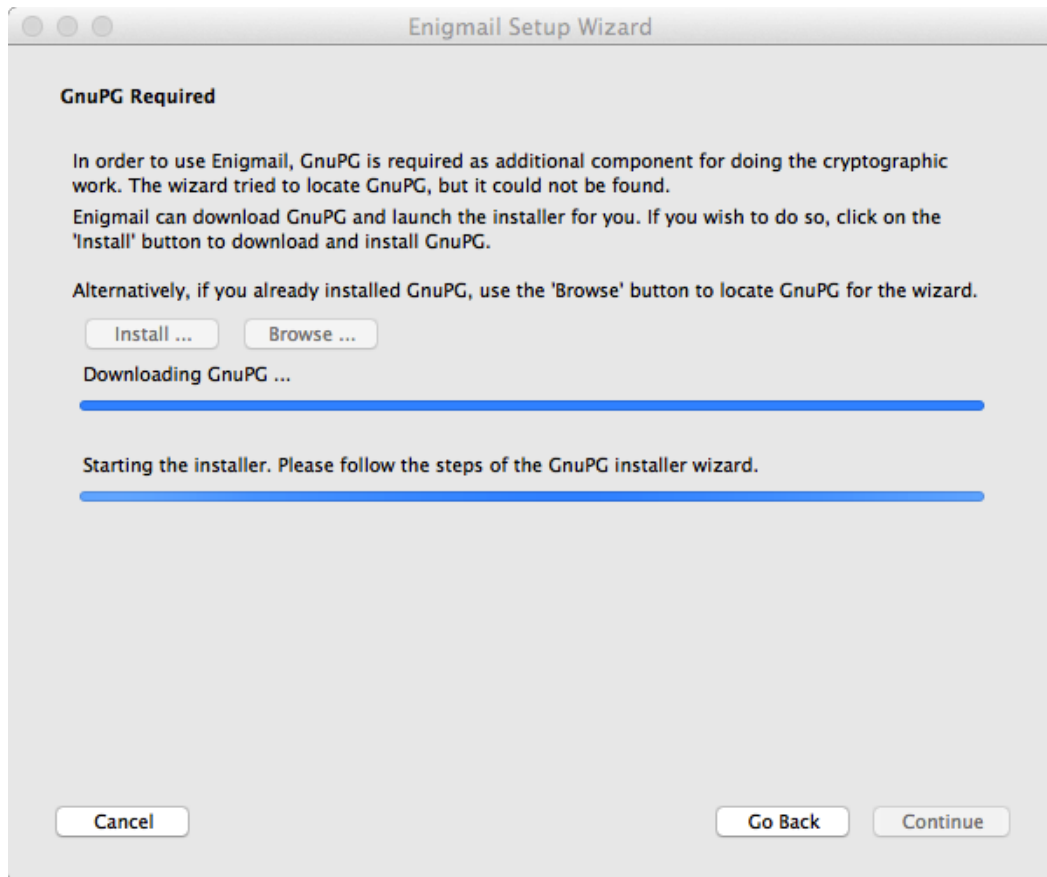
The Enigmail Setup Wizard couldn't find GnuPG on your computer.

If you are certain that GnuPG is available on your computer, please point Enigmail to the installed GnuPG program by clicking on *Browse* to select the GnuPG program folder.

Otherwise, please click on *Install*. Then Enigmail setup wizard will then download the required package (different for Windows and Mac OS X) and start the Installer of that package.

On Linux there is almost always a usable version of GnuPG, so this step is not needed there.

If in any case you need to install GnuPG manually, please refer to section 9.1.



This screen is shown during download and run of the GnuPG installer.

Note: The package is downloaded through a SSL secured connection. Furthermore, its checksum will be verified by Enigmail so the package is trustworthy.

The next screenshots show the installation of the windows version of GnuPG, provided by the Gpg4Win project. Just in case you want to do this manually: This is the Gpg4Win “vanilla” installer.

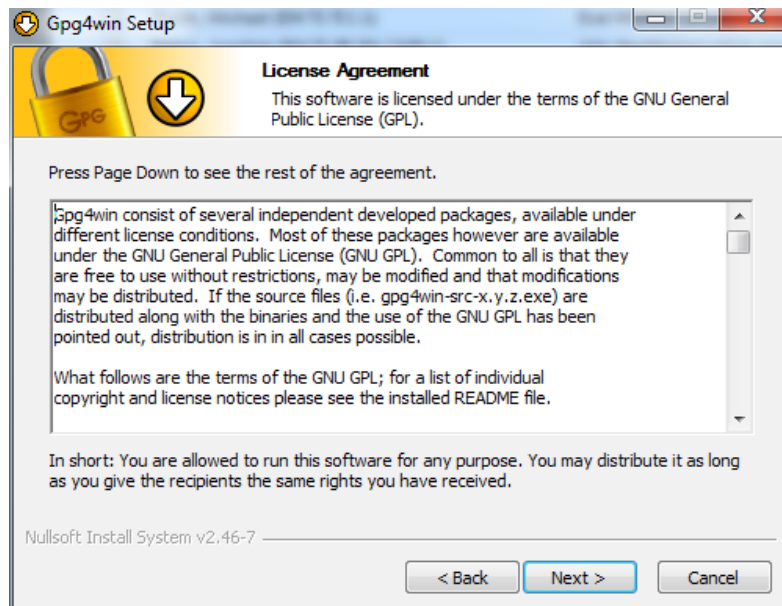
Mac OS X users may want to continue reading at page 16 to skip the windows specific pages.



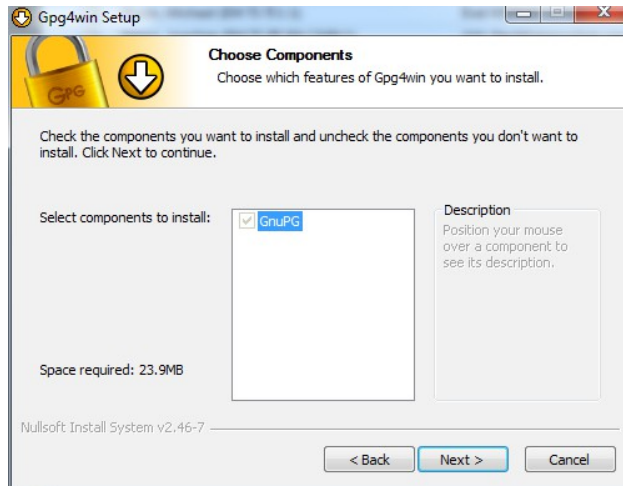
First selection: Choice of language.



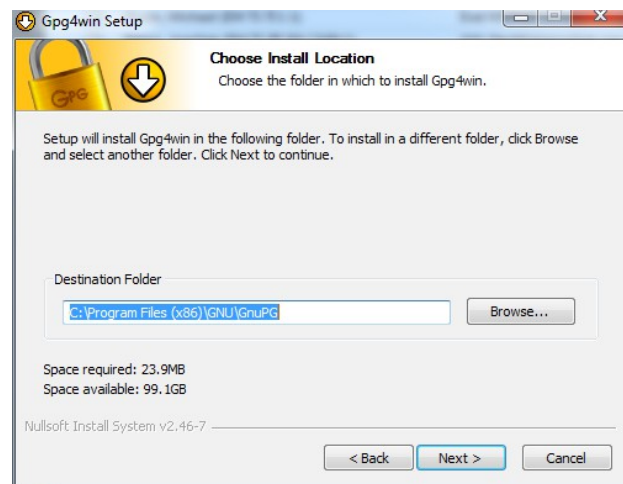
Second screen: Welcome message, click Next.



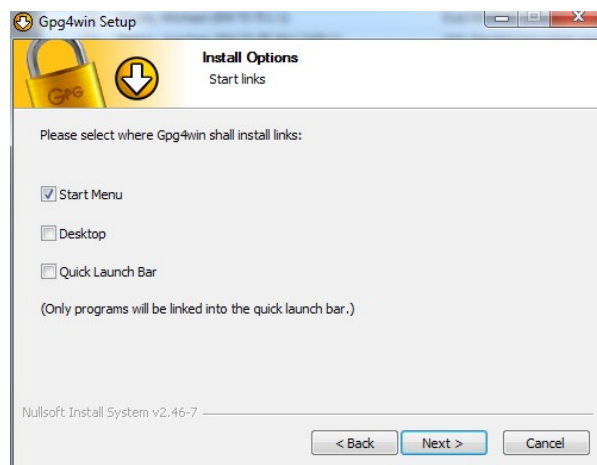
License agreement. Click Next.



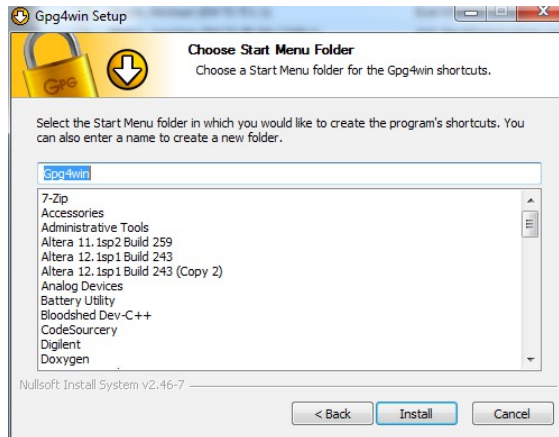
Selection of components (there's only one and compulsory component). Click Next.



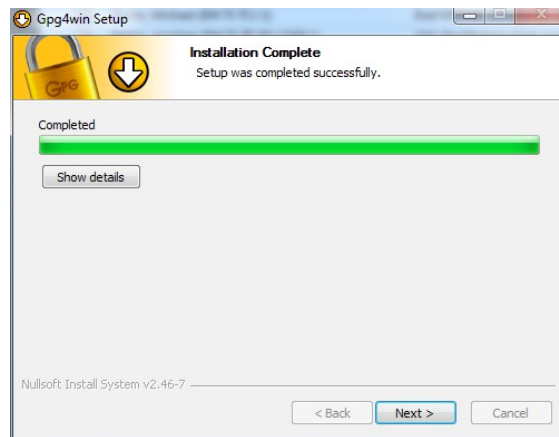
Select the folder where the program shall be installed (leave default if you're not sure). Click Next.



Select where you want links to the program be placed. Click Next.



If you checked “Start Menu” in the previous screen, select the Start Menu folder where the shortcut shall be placed in.



Now the installer performs its work showing a progress bar. Click Next when finished.



Now the installer has finished. Check the “Show README” file if you want to read it. Click Finish.

Please continue reading at page 17 to skip the following pages, they cover the Mac OS X GnuPG installer.

The following screenshots show the installation of the Mac OS X version of GnuPG, provided by gpgtools.org:



Please double click on “Install.pkg” to proceed.



Please consider carefully what to install. The only required component for the use within Enigmail is MacGPG2. The others are not necessary for the use with Enigmail:

GPGMail is a plugin for Apple Mail.

GPG Keychain Access is a key administration utility outside of Enigmail

GPGServices is a tool to en/decrypt files on your hard disk

GPGPreferences is a tool to administer the GnuPG configuration file.

Please only select those components if you want to use them.

Congratulations, you now have successfully installed the required GnuPG package for your platform!

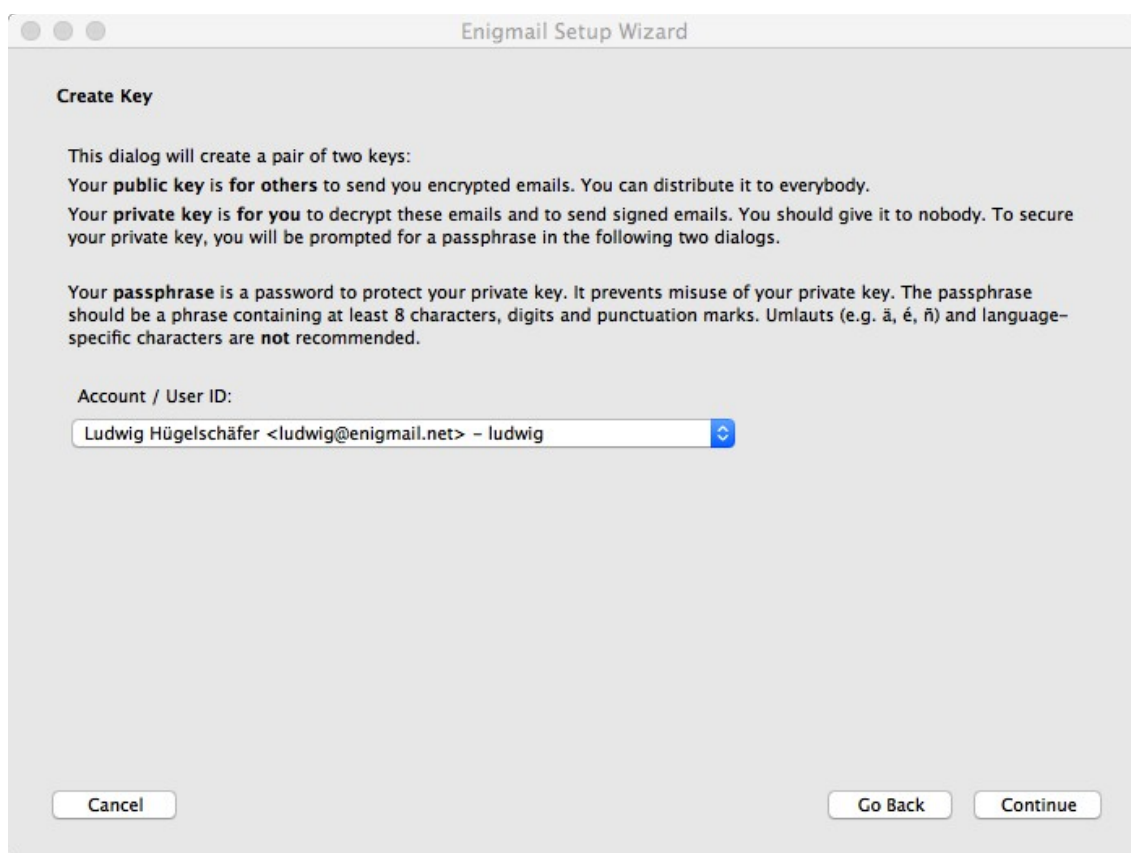
The Setup Wizard will now continue.

If you did select “Extended configuration” on the first wizard screen, please read on in chapter 2.1.2 at page 23.

If you did select “Manual configuration”, please read on in chapter 2.1.3 at page 27.

2.1.1. Enigmail Setup Wizard for beginners

The next screen shows the wizard, if you did select “standard configuration”:



Select the account/User ID you want to use for use with Enigmail. Don't worry, you can later add other accounts/User IDs if you wish to do so.

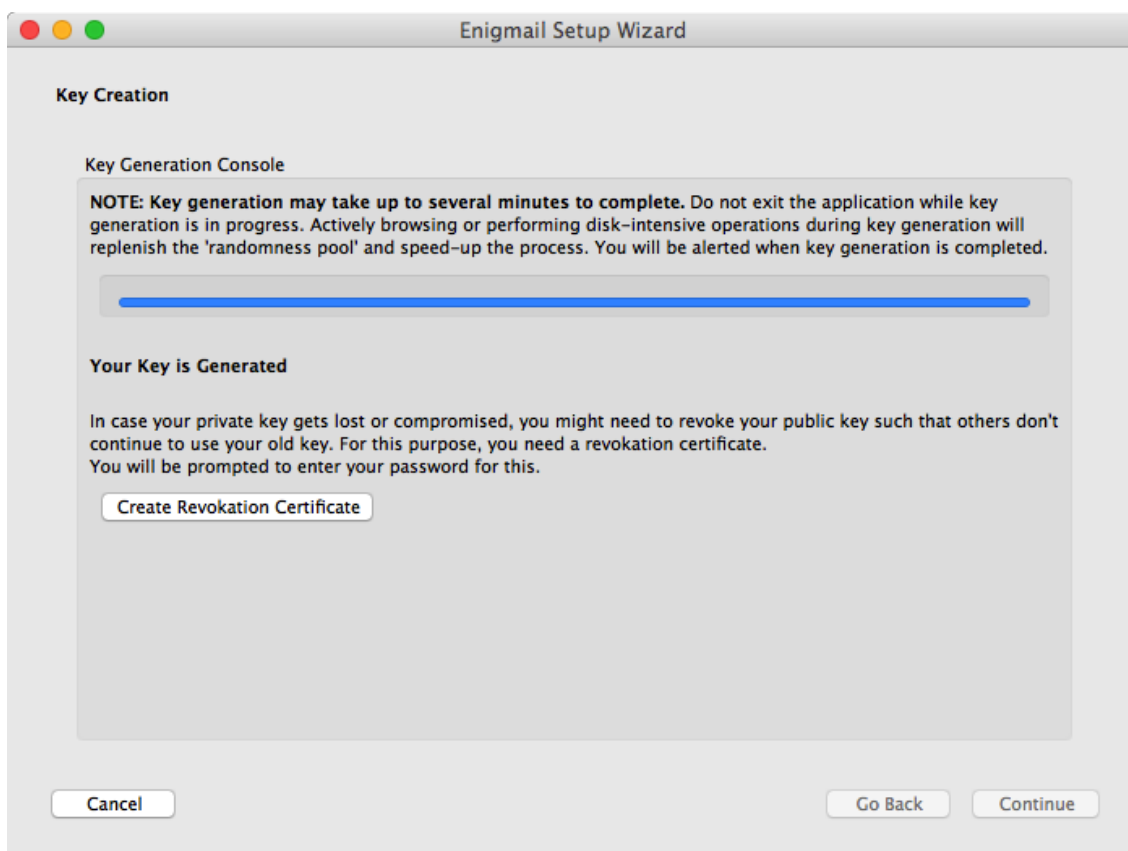
Click *Continue*.

The Wizard will now help you to get your key pair. There are several ways to obtain it:

1. Generating a new key pair
2. Using an already existing key pair

The wizard will guide you through the possibilities. If there is an already existing key pair on your computer, Enigmail will detect it and offer it to use.

More likely, as this is the first time you use Enigmail, you will need to generate a new key pair. The wizard will do this automatically for you:



You will be asked to choose a passphrase to protect your key pair: you will need that passphrase for signing or decryption of a message. Please repeat the passphrase (just to be sure there's no typo).

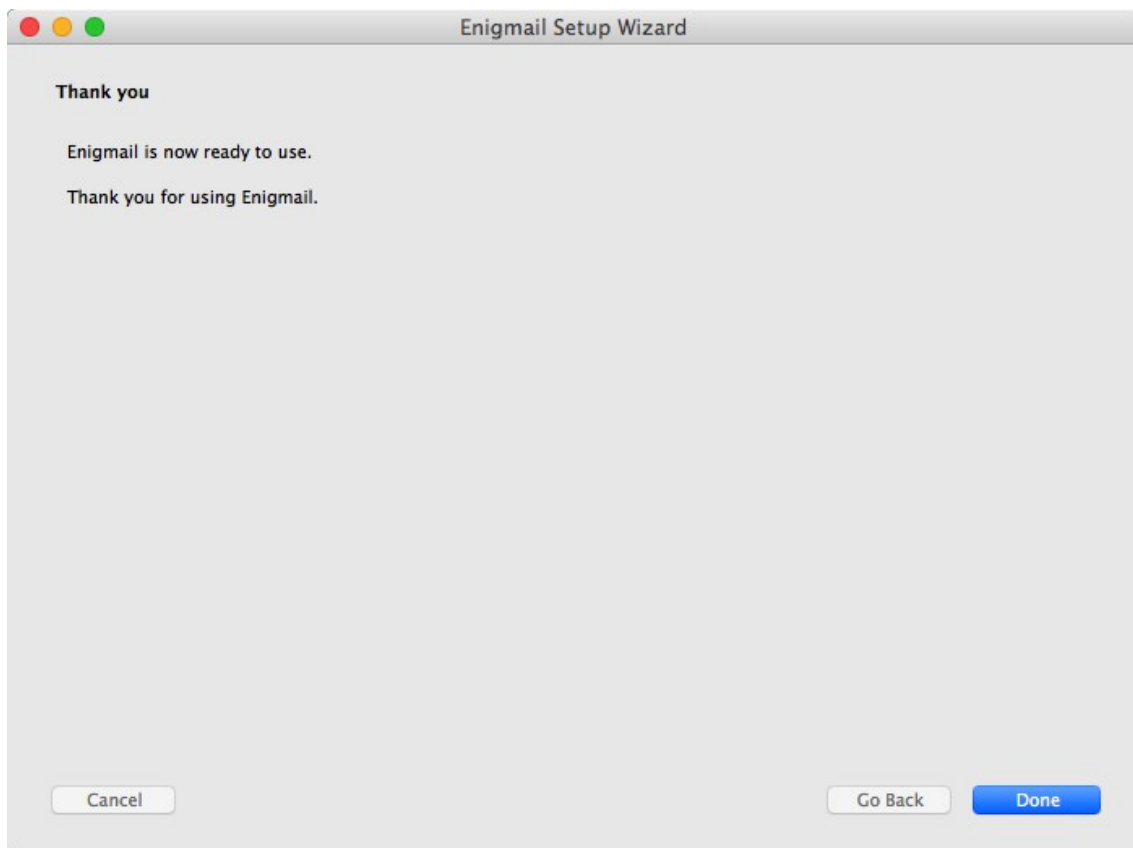
Important: Please make sure, that you remember that passphrase! If you forget it, you will not be able to use your key pair any longer! Everything encrypted for this key pair will be lost. Hints for selecting a good passphrase can be found in chapter 8.1 at page 93.

Once the key has been generated, the wizard will display a confirmation together with the button “Create Revocation Certificate”. Doing this is absolutely

necessary! If you lose your key or forget your passphrase, you can use this revocation certificate to revoke your key. Click on this button, and you will be prompted for a location on where to save the revocation certificate. Please save it in a safe location outside of your computer! This can be done afterwards.

If you like to know more about key generation, read Section 3.6 at page 42.

Close the dialog clicking on “Continue”.

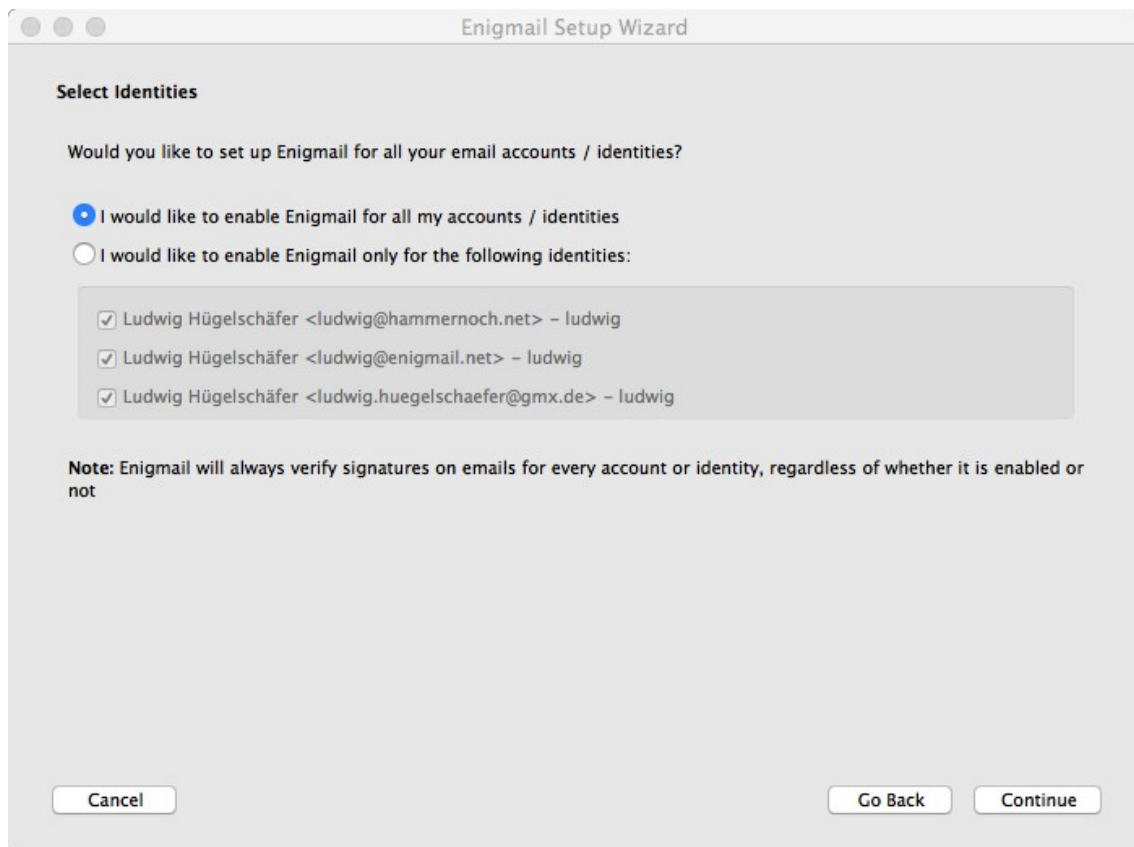


The Setup Wizard has done its job, and Enigmail is now ready to use!

Please read on at page 28.

2.1.2. Enigmail Setup Wizard for advanced users

This chapter will show the steps, if you selected an extended configuration at the first Setup Wizard page. If you did this accidentally or feeling unsure whether this was the right choice, click on “Go Back”.



Here you can choose whether to have Enigmail configured to work on all your email accounts and identities, or for some only. This can be changed at any time later, so this is not a permanent setting.

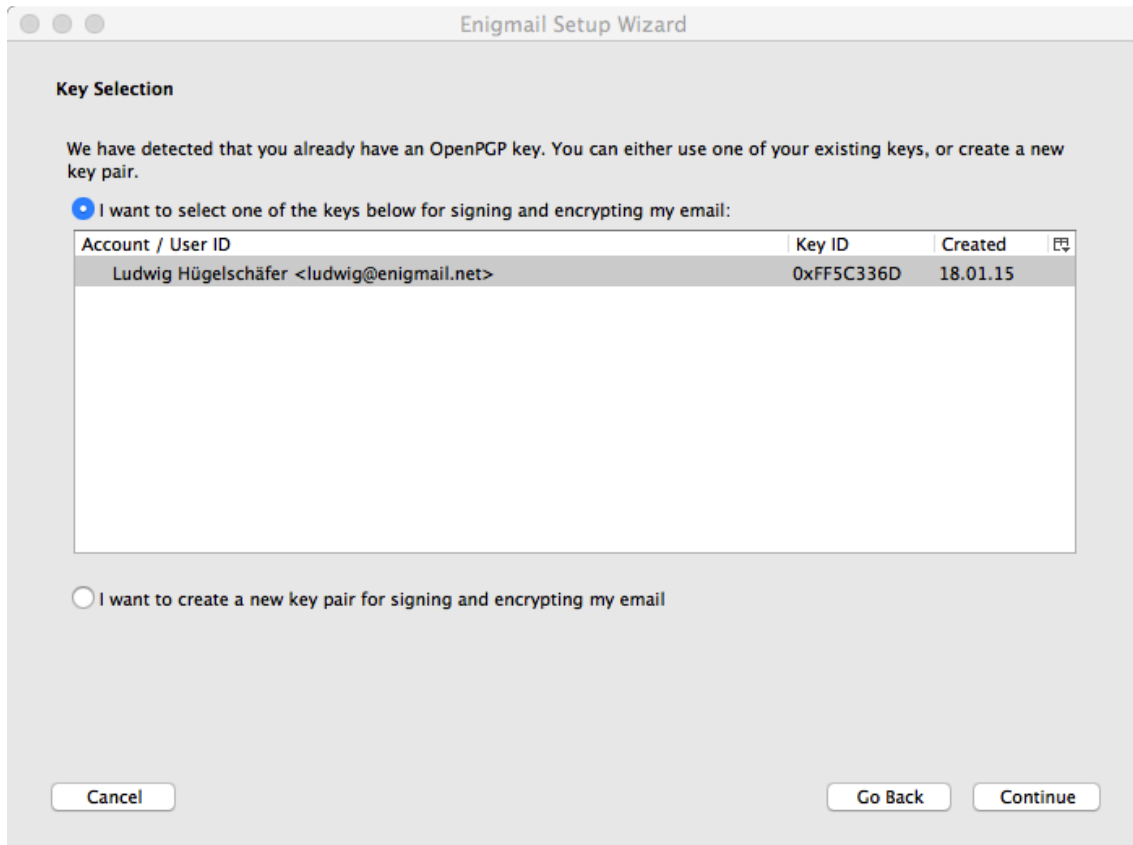
How to set up an account to use it with Enigmail is explained in Section 4.1 at page 47.

Click *Continue*.

The Wizard will now help you to get your key pair. There are several ways to obtain it:

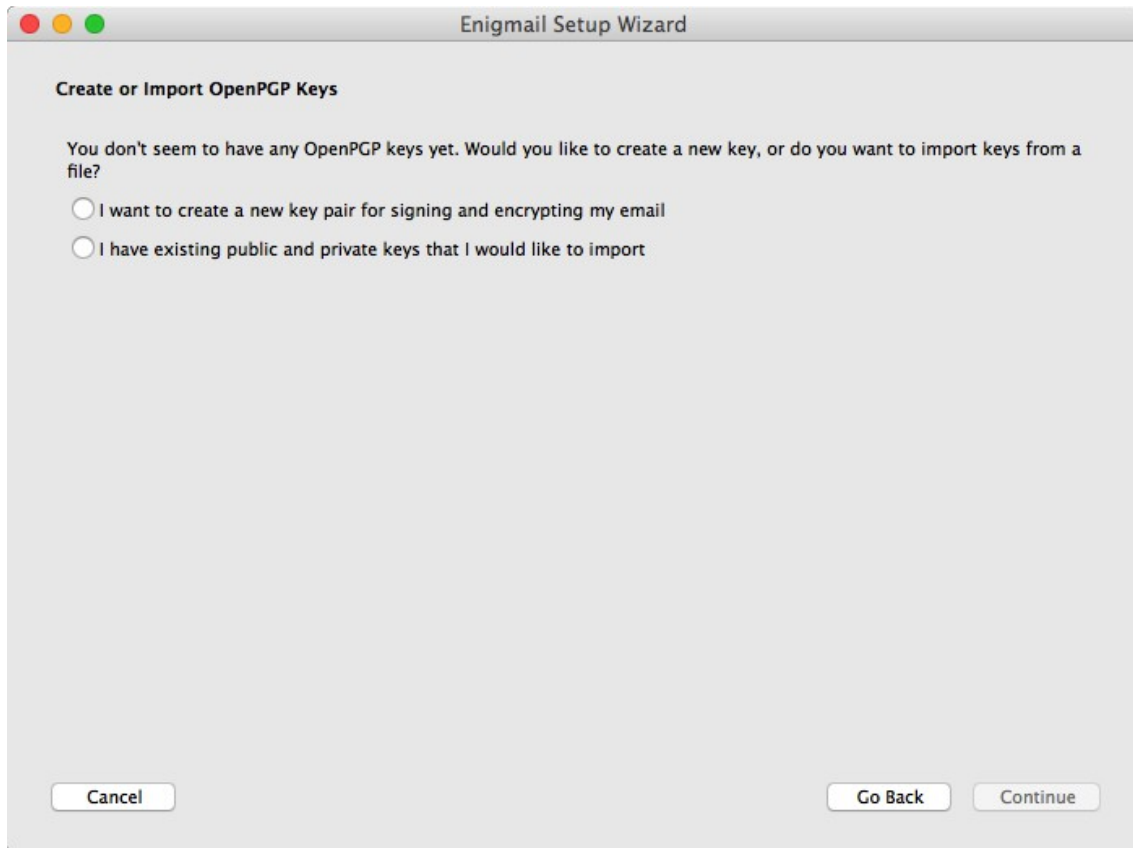
1. Using an already existing key pair
2. Generating a new key pair
3. Importing an existing key pair

The wizard will guide you through the possibilities. If there is an already existing key pair on your computer, Enigmail will detect it and offer it to use.



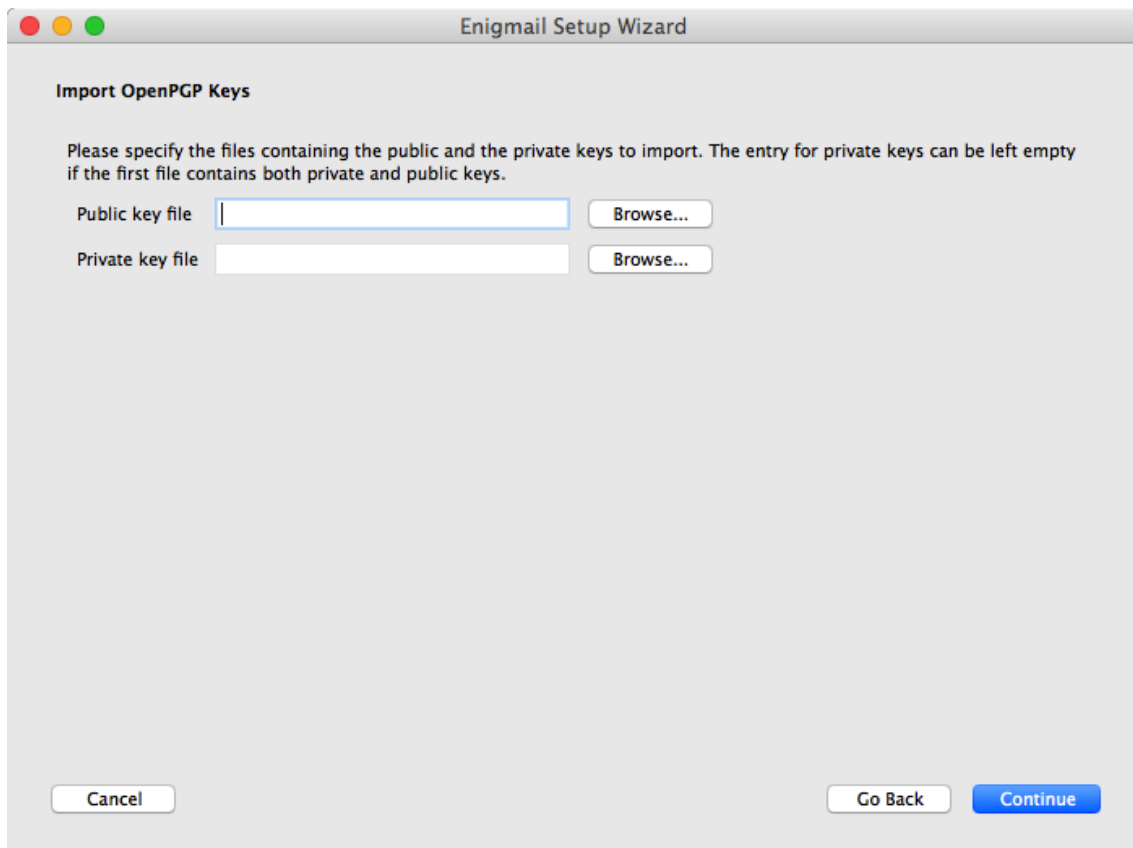
Select the presented key (or the one you want to use if there is more than one) and click *Continue*.

If no key pair is found, you are offered to either create a new key pair or to import one:



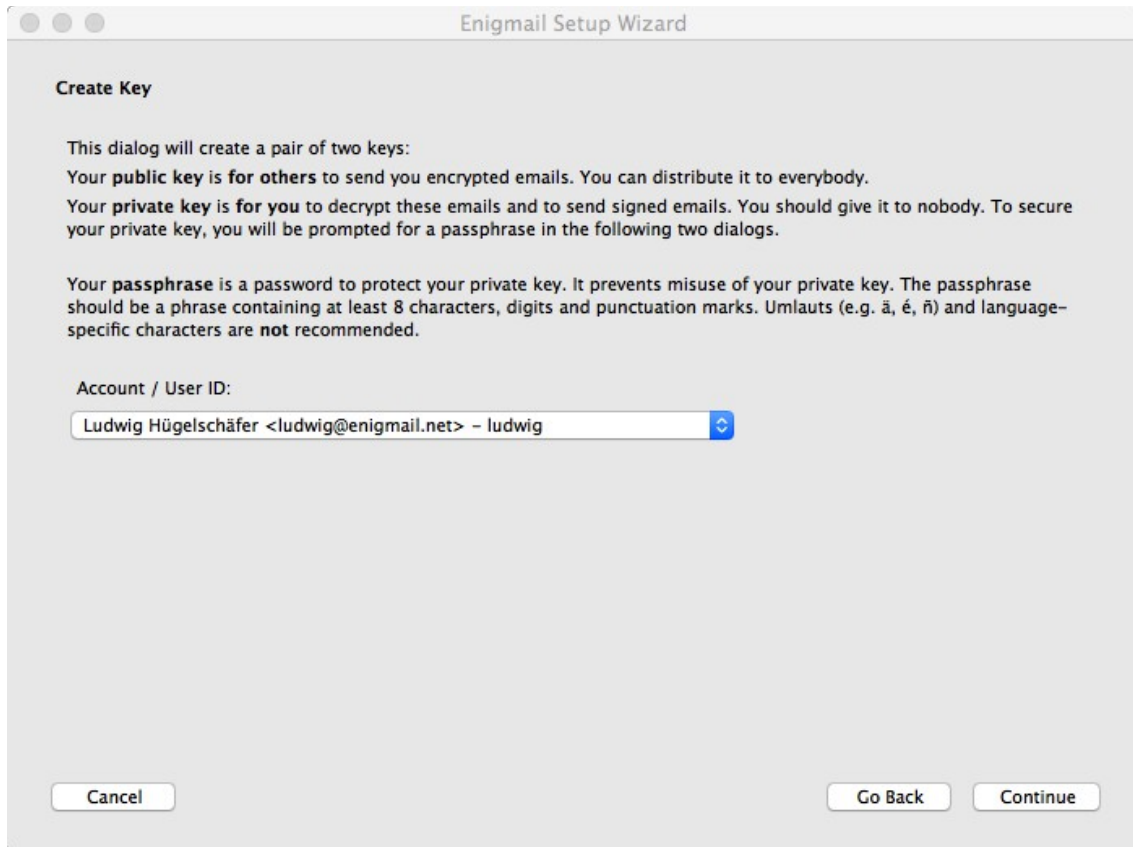
Select your option and click *Continue*.

If you decide to import a key pair the following dialog allows browsing for the suiting file:

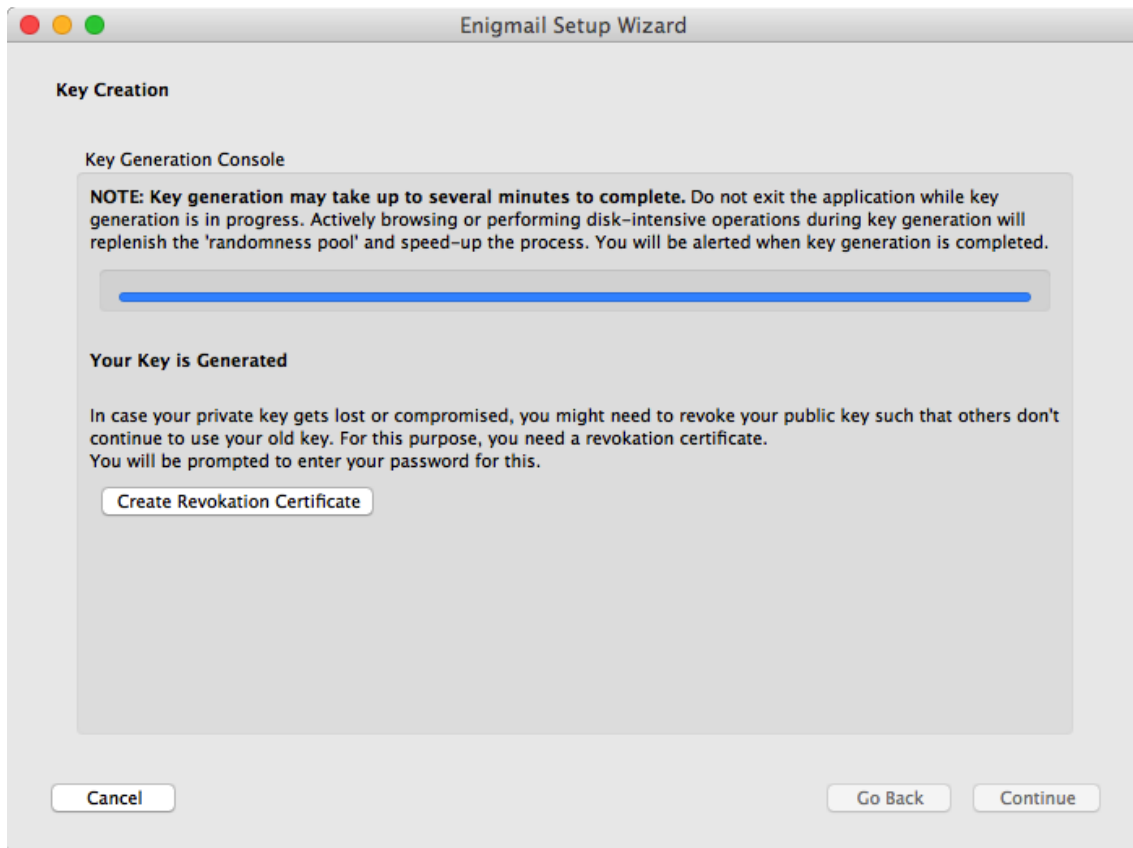


Please select the file or both files and click *Continue*.

If you decided to create a new key pair, the following dialog will be shown:



The name and email address you select here (called “user id”) will be associated with the new generated key pair. You can associate more than one user id with a key. Adding another user id to the new generated key can be done easily later. How to do that is shown in chapter 3.1.2 at page 33.



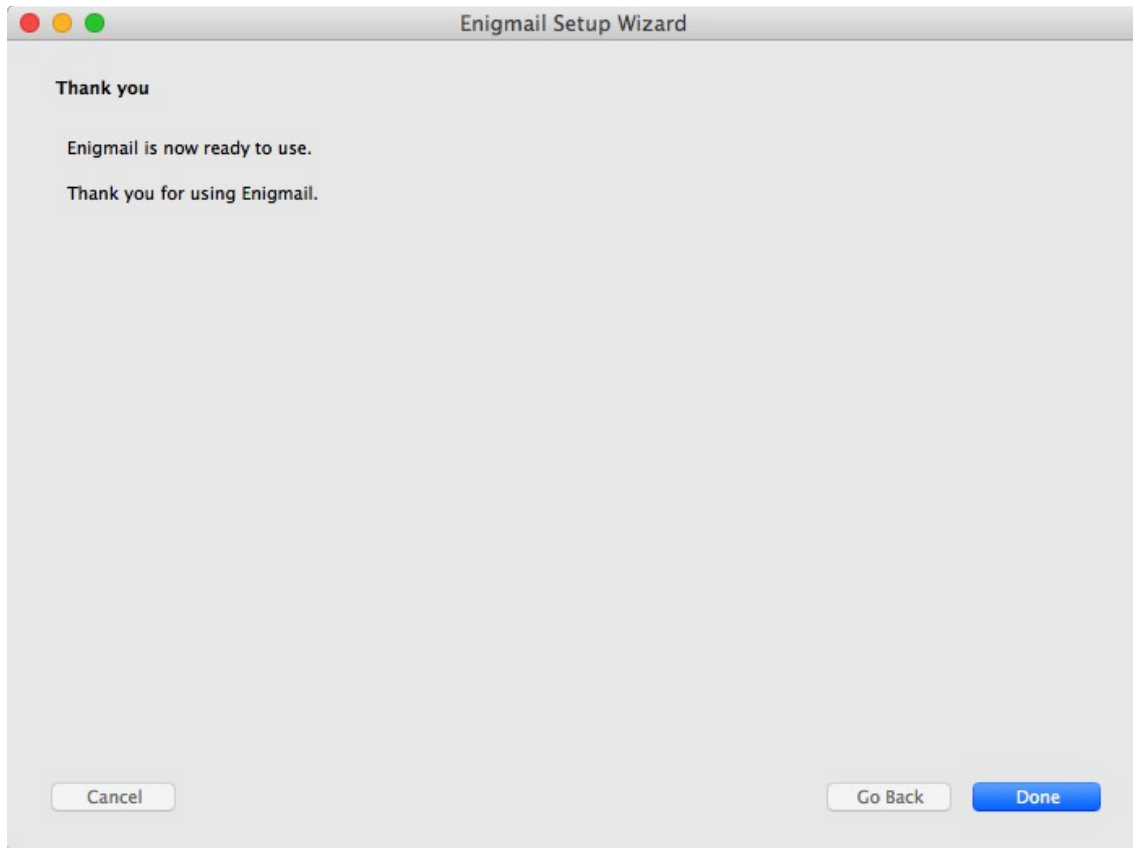
You will be asked to choose a passphrase to protect your key pair: you will need that passphrase for signing or decryption of a message. Please repeat the passphrase (just to be sure there's no typo).

Important: Please make sure, that you remember that passphrase! If you forget it, you will not be able to use your key pair any longer! Everything encrypted for this key pair will be lost. Hints for selecting a good passphrase can be found in chapter 8.1 at page 93.

Once the key has been generated, the wizard will display a confirmation together with the button “Create Revocation Certificate”. Doing this is absolutely necessary! If you lose your key or forget your passphrase, you can use this revocation certificate to revoke your key. Click on this button, and you will be prompted for a location on where to save the revocation certificate. Please save it in a safe location outside of your computer! This can be done afterwards.

If you like to know more about key generation, read Section 3.6 at page 42.

Close the dialog clicking on “Continue”.



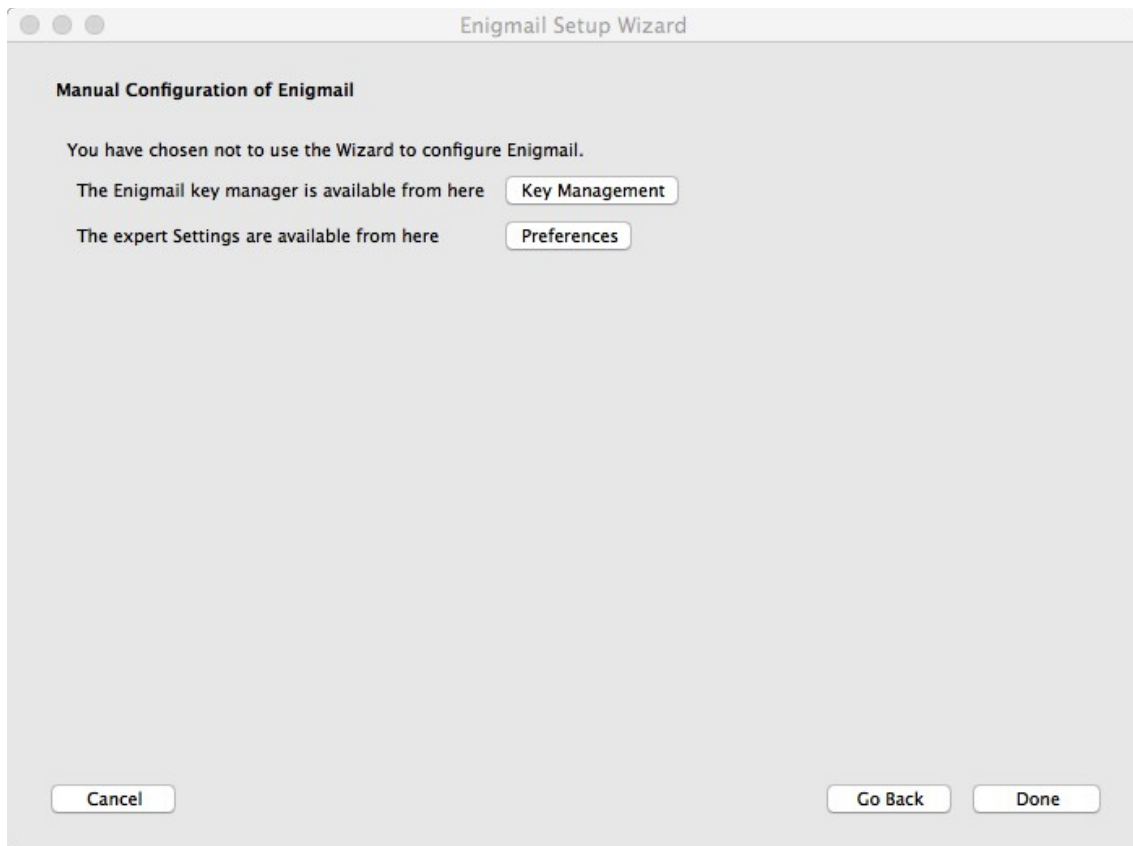
The Setup Wizard has done its job, and Enigmail is now ready to use!

Please read on at page 28.

2.1.3. Enigmail Setup Wizard for expert users

This screen will show up, if you selected an extended configuration at the first Setup Wizard page. If you did this accidentally or feeling unsure whether this was the right choice, click on *Go Back*.

There is no real wizardry here, in fact this dialog shows where to find everything needed for manual configuration.



Clicking the button *Key Management* you can directly jump to the Key Management of Enigmail, where you can manually generate a key (refer to section 3.6 at page 42).

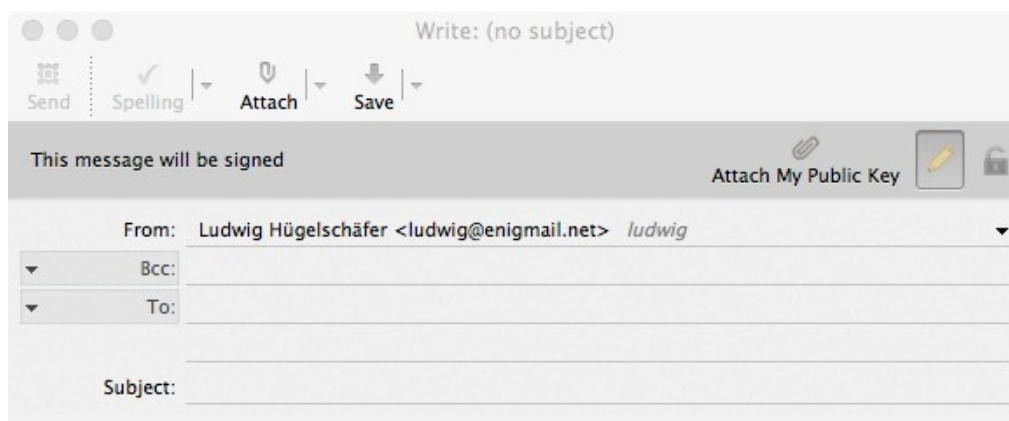
Also you should setup your preferences (expert view is already activated) using the *Preferences* button.

Once you've finished, click on *Done*.

2.2. Start using Thunderbird with Enigmail

When you start writing a mail, you will now notice a new *Enigmail* toolbar below the normal toolbar of the Compose window. This toolbar allows you to sign and/or encrypt the message using a single click on the shown icons. On the right end of this extra toolbar, a pen and/or a key icon are displayed, to show if signing and/or encryption is enabled.

If you miss some of the icons, you can configure this toolbar like the normal toolbar above: Right click (or CTRL-click on Mac OS X) and select “Customize” from the appearing pop-up menu. You can now drag icons into the toolbar or remove them as you like.



You can immediately send signed mail to anyone. However, in order to allow someone to verify your signature or to send you an encrypted message, you must provide him with your public key. You can send your public key as an attachment either by clicking the “Attach My Public Key” button in the Enigmail toolbar or by choosing *Enigmail* → *Attach Public key...* in the Compose window, and then by selecting your key in the Key Selection window that will appear.

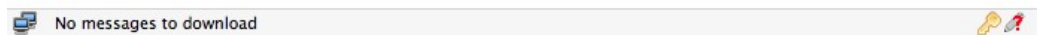
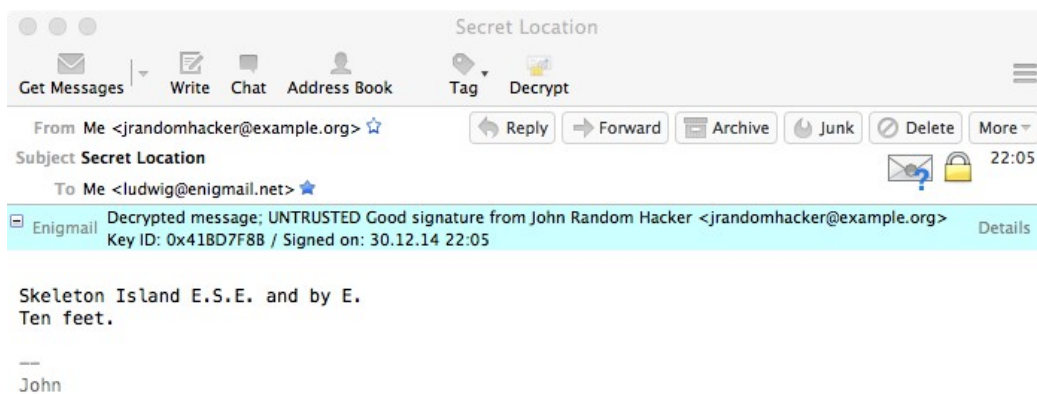
All your stored keys (your own key pair, and other people's public keys you have acquired) can be seen in the Key Management, via the menu command *Enigmail* → *Key Management*.

To send encrypted mail, you need to have the public key of the recipient. You

can acquire it in one of the following ways:

- ask him to email you his public key as an attachment; then right-click on the attachment and choose *Import OpenPGP Key*;
- retrieve his public key from a keyserver via *Keyserver* → *Search for keys* from Key Management.
- download his public key from his web site as an ASC file, then import it via *File* → *Import Keys From File* from Key Management;

When you receive a mail message that has been OpenPGP-secured (signed and/or encrypted), it will appear as such:



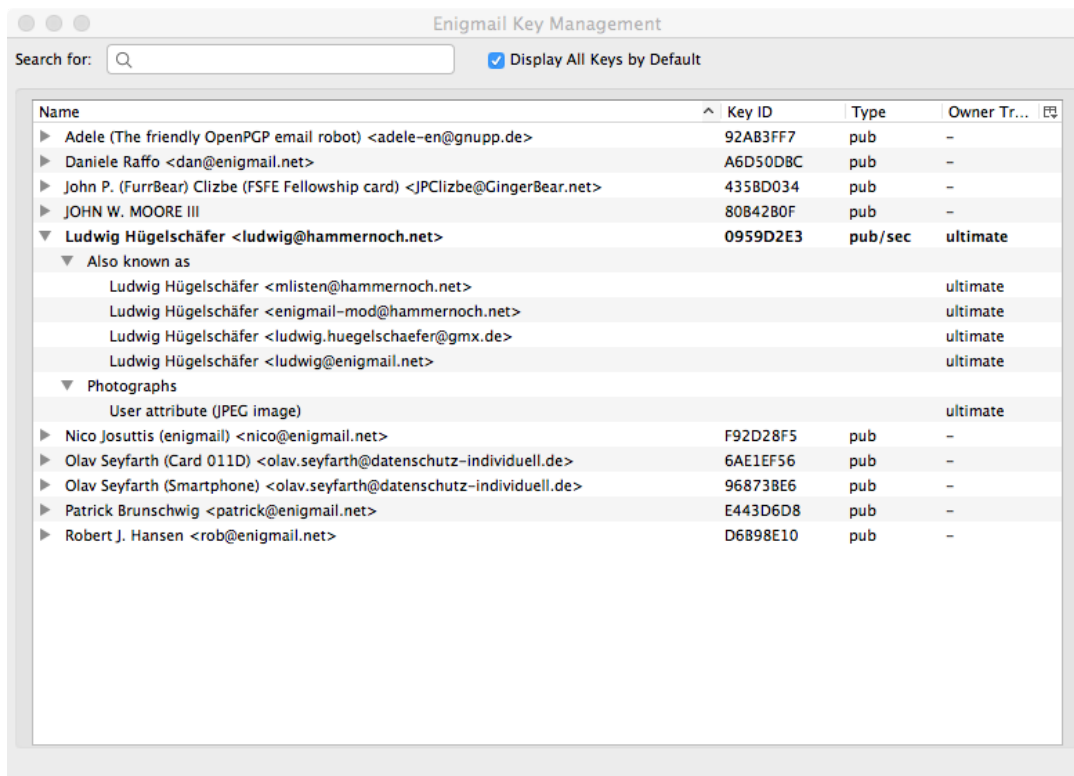
The message in the figure has been both signed and encrypted, as shown in the Enigmail status bar. **Hint:** if it doesn't show you the full message, click on the little [-] box and it will expand.

Thank you for using Enigmail! These are the basics of it. You can read about all topics in detail by perusing the rest of this handbook.

3. KEY MANAGEMENT

Once you have Enigmail and GnuPG on your system, you need to populate it with keys: it's pretty useless without them. You need to have your own key pair in order to sign messages and allow anybody to send encrypted messages to you. Furthermore, you will need the public key of a person in order to send that person an encrypted message, or to verify the signature in a message sent by that person.

Select *Enigmail* → *Key Management* to open the Key Management window.



The Key Management window shows all keys (yours and other people's) you have stored on your computer; this is called your *keyring*. The set of all public keys that you have collected is often called your *public keyring*.

Key pairs (i.e. a bundle of public key and its companion secret key) are shown in bold. Revoked or expired keys are shown in italic.

If you have run the Setup Wizard and generated a new key pair, or imported an existing one, it will be shown here. (If it doesn't, tick the option *Display All Keys by Default*.) Otherwise, the window will be empty.

By clicking the expand gadget at the left of each key, you can see the key's additional user IDs and PhotoID, if present. The columns (*Key ID*, *Type*, *Key Validity*, *Owner Trust*, *Expiry*, and *Fingerprint*) show a number of other key properties: you can choose which columns you want to see by selecting them in the rightmost gadget in the column header bar. In the rest of this chapter we'll explain what these properties mean.

The menu bar of the Key Management window allows you to operate on the keys of your keyring. To do that, select a key, and then choose a menu item. You also have most of these menu items in a pop-up menu that shows up when you right-click on a key. Some menu items will be disabled (greyed out) if the operation is not possible on the key you selected. Some operations on a key require that you have the companion private key, so you can accomplish them on your key pair but not on other people's public keys.

3.1. Operations on your key pair

Once generated, your key pair should show up in the Key Management window. (If it doesn't, tick the option *Display all keys by default*.)

3.1.1. Examining the key properties

Select your key pair and choose *View* → *Key Properties*. A new window will pop up, showing all the key's properties. As you've seen, most of these properties can also be viewed directly from the Key Management window.

The screenshot shows a window titled 'Key Properties' with the following fields and tables:

- Primary User ID: Ludwig Hügelschäfer <ludwig@hammernoch.net>
- Key ID: 0x00A1E6FE
- Type: key pair
- Key validity: ultimate
- Owner trust: ultimate
- Fingerprint: E160 206F B140 E1A6 95C4 99D8 3ADB FA6D 00A1 E6FE

Additional User ID	Valid
Ludwig Hügelschäfer <ludwig.huegelschaefer@gmx.de>	ultimate
Ludwig Hügelschäfer <ludwig@enigmail.net>	ultimate
Ludwig Hügelschäfer <mlisten@hammernoch.net>	ultimate

Key Part	ID	Algorithm	Size	Created	Expiry	Usage
primar...	0x00A1E6FE	RSA	4096	14.01.15	13.01.20	Sign, Certify
subkey	0x0D5096B5	RSA	4096	14.01.15	13.01.20	Encrypt

At the bottom left, there is a 'Select action ...' dropdown menu. At the bottom right, there is a 'Close' button.

- The *Primary User ID* is the name and email address associated with the key i.e. the key owner's. These match the name and address of the email account you generated the key for.
- The *Additional User ID* field show the other user IDs (name and email address) associated with the key, if any. By now your key pair has no additional user ID: should you want to add a user ID, how to do that is explained in Section 3.1.2 at page 33
- The *Type* of the key shows “key pair” (“pub/sec” in Key Management window) to mean that this is a key pair. For other people's public keys that you have imported, this field will show “public” (“pub” in Key Management window).
- *Key validity* indicates the validity of the key. Key validity will show you whether a key has expired or has been revoked. For the key pair that you just created, validity should display “ultimate”. This field is directly related to the Web of Trust model, explained in Section 3.4 at page 37, and is related to public keys purporting to other people.
- *Owner trust* indicates the trust in the key's owner on verifying other keys carefully before signing them. For your own key pair, it should display “ultimate”. For other keys it will display what you have entered. The “Owner trust” property is not exportable.
- The *Fingerprint* is a unique 40-digit hexadecimal number automatically generated by applying a hash algorithm to the key. The fingerprint unambiguously identifies the key worldwide. This means that no key will have the same fingerprint of another key ever created in the whole world.
- The last 8 digits of the fingerprint make the *short Key ID* which, being much shorter, is used to locally identify the key for all practical purposes. While the fingerprint is unique for each key ever created worldwide, you can safely assume the key ID unique for each key in your keyring. (You would need to collect dozens of thousands of public keys in your keyring before there is a good possibility that two keys share the same key ID.)
- The bottom field shows all key components i.e. primary key and subkey(s) along with their properties: the key ID, the algorithm used for the key, the key size in bits, the creation date, and the expiration date (if any) and the key capabilities (possible values: Sign, Certify, Encrypt and Authenticate).

Key ID and fingerprint are often prepended by the characters 0x, which is the prefix indicating a hexadecimal number.

A drop-down menu named *Select action* at the lower left of the Key Properties window allows you to perform different operations on the key. The operations you can perform on a key depend whether you have or not the companion private key, so you can accomplish some operations on your key pair only and not on other people's public keys; Enigmail shows only menu items relevant to the permitted operations.

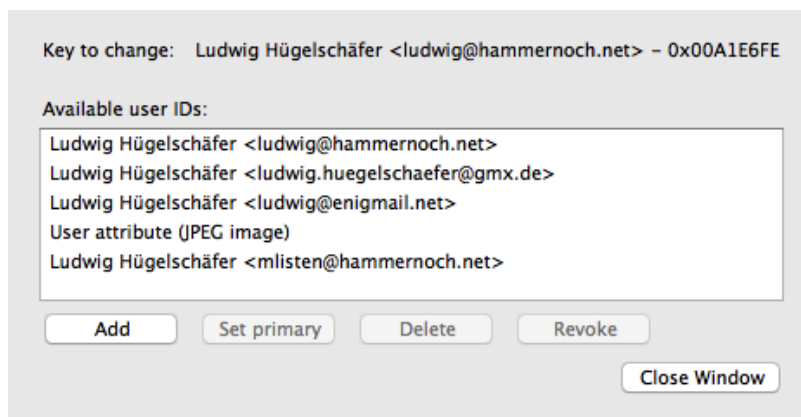
The same operations are available from the menu bar within the Key Management window as well.

Now hit the *Close* button to close the Key Properties window and go back to Key Management.

3.1.2. Specifying additional user IDs

You might desire to use more than one email address to send secure email from. In this case, you do not need to generate one key pair for each address: you may simply associate multiple email addresses to your key pair. This will save you from the burden of managing multiple key pairs.

Select your key pair and choose *Edit* → *Manage User IDs* from Key Management, or choose *Manage User IDs* from the Key Properties of your key. A window will pop up to show you a list of all user IDs (primary user ID and all additional user IDs) that are currently associated with the key. If this is the first time you perform this operation, your key will have only a primary user ID.



The *Add* and *Delete* buttons add and delete other user IDs. A user ID is composed of a name and email address; it is also possible to put an optional comment.

The *Set primary* button sets the selected user ID as primary, and as a consequence the previous primary user ID is relegated to the role of additional user ID.

The *Revoke* button revokes the selected user ID, which is then greyed out and deactivated. The difference with deletion is that a revoked user ID is still associated to the key pair but no longer usable.

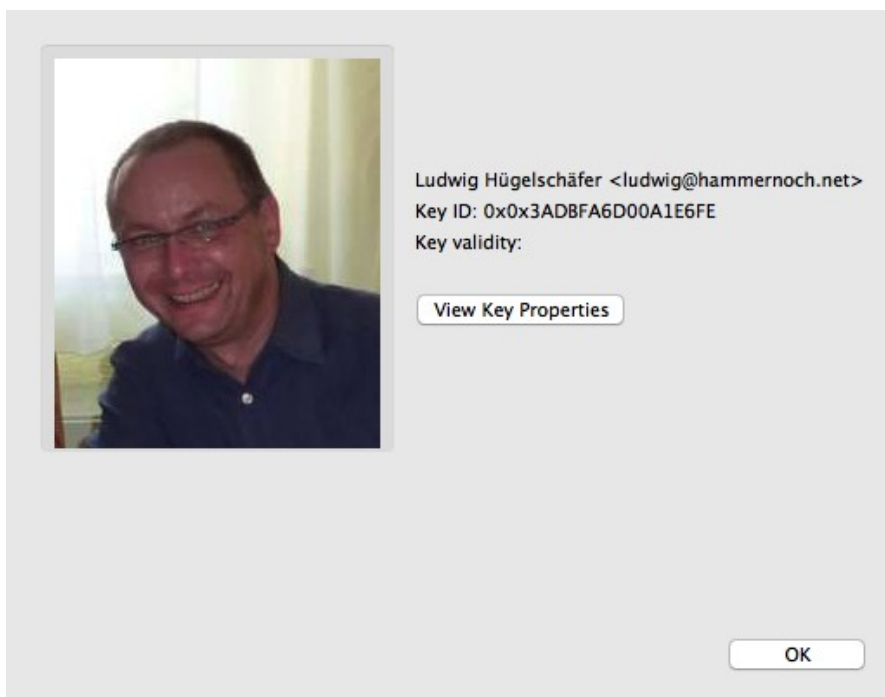
Click on *Close window* to save the changes you have made to the key and to return to Key Management.

All user IDs are included in the public key when it is sent or exported. There is

no mechanism in OpenPGP for selecting which user IDs are included and which not. This also applies to the PhotoID, which is a special type of user ID.

3.1.3. Adding a PhotoID

Public keys may have an embedded PhotoID i.e. the owner's photograph. To view the PhotoID, select the key and choose *View* → *PhotoID* from Key Management, or click the *View PhotoID* button from the Key Properties. If the menu item or button are greyed out, the key carries no PhotoID.



To add a photo from yourself to your key, click on your key and select “Add Photo” from the popup-menu. Then enter the path of your image file. Suggested image sizes are 240x288 pixels (GnuPG) or 120x144 pixels (PGP).

The image will be included in your public key as an additional user ID. For this reason, GnuPG and PGP recommend to use JPEG files of maximum 7 Kb, otherwise your public key may become very big in size.

3.1.4. Changing the passphrase

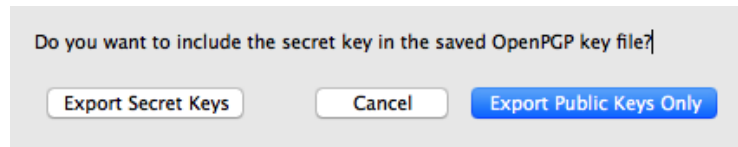
To change the passphrase that protects your private key, click on your key and choose *Edit* → *Change Passphrase* from Key Management, or click the *Change Passphrase* button from Key Properties. GPG-Agent will ask you twice for the new passphrase – just to be sure that there are no typos.

3.1.5. Making a backup of your key pair

Once you've generated your key pair, verified its properties, it's a very good

idea to make a backup copy of it.

Select your key pair from Key Management and click on *File* → *Export Keys to File*. You will be asked whether you want to include your secret key in the saved OpenPGP key file:



If you now click on *Export Secret Keys*, the exported file will contain your whole key pair (secret key and public key). If you click on *Export Public Keys Only* instead, the exported file will contain your public key only.



Be very careful on how you export your key pair!

When backing it up, you **must** include the secret key, or the backup will be useless. When exporting in order to distribute your public key (as will be explained in Section 3.2.1), make sure you do **not** include the secret key, or you'll compromise your key pair.

Since you're making a backup, click on *Export Secret Keys*, then choose where to save the file. Your key pair will be saved as an ASC file named like that:

```
Firstname Lastname you@domain.com (0x89ABCDEF) pub-sec.asc.
```

As you see, the filename is composed of the primary user ID of the key, the key ID, and the word “pub-sec” indicating that this is a key pair. Store this file in an external hard disk, thumb drive or similar, keep it in a safe place and don't pass it to anybody.

3.2. Distributing your public key

Now that you have a key pair, you must distribute your public key around. This is necessary so people can use it to send you encrypted messages and verify the signature on messages you send. You can do this manually and/or by the means of a public keyserver.

3.2.1. Share your public key manually

On the Key Management window, select your key pair and click on *File* → *Export Keys to File*. Again, you will be asked whether you want to include your secret key in the saved OpenPGP key file: make sure you click on *Export Public Keys Only* this time. Save the ASC file, which will have a name like

```
Firstname Lastname you@domain.com (0x89ABCDEF) pub.asc.
```

This is a copy of your public key; notice the “pub” word at the end of the filename. You can now put the file on your website for people to download it, or carry it around in a USB drive to distribute it to people, or send it via email as an

attachment.

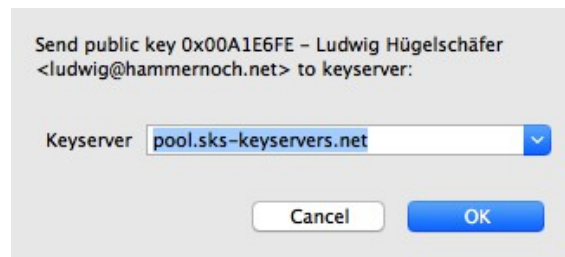
Concerning this last option, there is a simple Enigmail shortcut for it: when writing your mail in the Message Composition window, simply click on “Attach My Public Key” or choose from the menu bar *Enigmail* → *Attach My Public Key* and Enigmail will attach it upon sending.

3.2.2. Publish your public key on a keyserver

By far, the easiest way to let the world know your public key is to publish it on the public keyserver network, a global database of keys.

Some people believe it's a bad idea to use a keyserver, because spammers search them for email addresses. While this is true, the amount of spam you risk to receive by doing this is quite small: you receive roughly ten times more spam when you put your email address on a web page or post a message on the Usenet.

In order to publish your key, click on your key and select *Keyserver* → *Upload public keys*. You will be given a choice of keyservers; *pool.sks-keyservers.net* is the one Enigmail uses by default. Major keyservers periodically synchronize themselves.



Click on *Ok* and wait for your key to be uploaded. Your key is now published on the Internet for anyone to find.

3.3. Importing public keys

You need someone's public key to accomplish two things: encrypt a message destined to this person, or verify the signature of a message coming from this person.

You might obtain the public key as an ASC file from the person himself, either as an email attachment or by hand. In this case save the file on your computer, then from Key Management select *File* → *Import keys from file*. Choose the ASC file you just saved. Enigmail will add this public key to your keyring.

Another way to find someone's public key is to download it from a keyserver.

Select *Keyserver* → *Search for keys* and insert as the search term a part of the name or email address of this person. You may also search for key IDs. The keyserver will return a list of public keys that match. Tick the checkboxes on the left of any key you would like to import and Enigmail automatically will do it for you.

Later, you can also refresh your public keyring in part or as a whole by the menu commands *Keyserver* → *Refresh Selected Public Keys* and *Keyserver* → *Refresh All Public Keys*.

You might also find a public key published as raw text, for instance in a personal web page. It will appear like this:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.4.9 (MingW32)

mQGIBeGn3cERBAC+2os2hoACip4EiKxEzv+iVHOWaznOJIGZY9zY4y8C0BhUP++q
ccgO9vgNO1vIXApYvJctwX9HFJieNdlsBrWOR69hPBAAbDo+3BbOKwJFYgq8akYnv
tBCOdCNfOFwQs/8XdOH25/Oig+UjKhgxwKjkdd1UCj7shdGioXOvjO13xwCgzrGa
k2oA9Bne3hW+jUPjJlU4UbKD/i7mbQfFwTgxcXfRfsVDnkmPc+QvKe00ajfRP31o
(... 30 lines omitted...)
3bRLiE8EGBECAA8FAkgN3cECGwwFCQlmAYAACgkQW9ZLGa9iatRGlACeIyyIBTGj
uhrZRubL8gSdI3HyPzEAnAqWD1g+DvEHyEOZ0/2rUcsj7CT2
=3RyK
-----END PGP PUBLIC KEY BLOCK-----
```

This is also what you see if you open a public key ASC file with a text editor.

Select the whole block of text, including the lines `-----BEGIN PGP PUBLIC KEY BLOCK-----` and `-----END PGP PUBLIC KEY BLOCK-----` and copy it to the clipboard (*Ctrl+C* under Windows). Then choose *Edit* → *Import keys from Clipboard* to import this public key into your keyring.

You can search and add public keys to your keyring at any time. Enigmail will also offer to do so automatically when you receive a signed message from someone and you do not have his public key; we'll see this in Section 4.2.3. If you followed all instructions up to here, by now your keyring should contain your own key pair and a number of public keys purporting to other people.

3.4. Validity of public keys

Importing a public key from a keyserver is quick and easy, but it does not guarantee that the key really purports to the person specified as the user ID. After all, anybody could have generated and uploaded that key. Fake key on keyserver are a fact.

Furthermore, if you received someone's public key via email, you should reflect on the fact that there is an inherent security problem in using the same channel (e-mail) both for key distribution and for the exchange of messages secured by that key. Theoretically, an attacker that is able to compromise the channel can

replace the public key in transit with a rogue public key of a key pair he created himself (*man-in-the-middle attack*). The attacker can now intercept the message that was encrypted with the rogue public key, and decrypt it since he owns the companion private key.

3.4.1. Verification of public keys

A solution to this problem is to check the public key's fingerprint with the key owner through a different channel. You may phone the key owner and have him read the key fingerprint to you. If the fingerprint does not match, you both know that the key was replaced in transit or you were otherwise given a fake key.

However, the best way to verify other keys is to meet the person face to face, check his identity (by his passport, e.g.), receive the fingerprint of his key and let him state which mail addresses are associated with this key. How to do this properly is shown in detail in chapter 8.5 at page 102.

This procedure is safe but can be very difficult to do. Imagine, your mail correspondent lives very far away and you cannot meet in person. This problem was therefore addressed in OpenPGP by developing a trust delegation model called *Web of Trust*. But before this is explained, there is another important thing:

3.4.2. Signing other keys

Finally, if you have verified a public key, you should tell it to the OpenPGP system. This is done by signing your mail partner's key. Signing is needed to make this key valid and useable for you.

In key management, select the particular key and then either using the menu item *Edit* → *Sign Key* or right click (Mac OS ctrl-click) and select *Sign Key* from the appearing popup menu. Then the signing dialog will open:

The screenshot shows a dialog box for signing a key. It contains the following information:

- Key to be signed: John Random Hacker <jrandomhacker@example.org> - 0x0ADF0DE8
- Fingerprint: A188 5C6D 89B1 1869 39B7 529E 047B 52F4 0ADF 0DE8
- Key for signing: Ludwig Hügelschäfer <ludwig@hammernoch.net> - 0x00A1E6FE
- How carefully have you verified that the key you are about to sign actually belongs to the person(s) named above?
 - I will not answer
 - I have not checked at all
 - I have done casual checking
 - I have done very careful checking
- Local signature (cannot be exported)
- Buttons: Cancel, OK

The key to be signed is shown with his user Id (name and mail address), followed by his short key Id.

The next line shows the fingerprint of his key. You should check it again!

The third line shows the secret key you are signing with. In most cases, there will only be one, otherwise you can select by a drop down menu.

You are asked how carefully you have verified that the key you are about to sign actually belongs to the person named in the first line.

- I will not answer (this is the default)
 - Select this, if you don't want to tell anybody how you did verify this key
- I have not checked at all
 - You should only do this, if you want to keep your signature local
- I have done casual checking
 - This option may be selected if you have e.g. received the fingerprint by phone and not met the person yourself
- I have done very careful checking
 - Only select this option if you have carefully checked the person's passport or a similar ID document.

Then there's a checkbox at the bottom: *Local signature (cannot be exported)*. Check this if you do not want to become your signature public. This is extremely helpful when you have not met the person in real and only want to get that key into "valid" state.

If you later – after meeting face to face and thorough verification – decide to issue a public (exportable) signature, you can then repeat this step without checking the "Local signature" box. The exportable signature then replaces the local signature.

Click on *Ok* to perform this operation. Your signature will be attached to that public key; if the key was already signed by other people, your signature will be added to the list. When a key is exported, the list of signatures is exported with it.

Once you have signed a public key, you should return it (for instance in a signed email message) to the owner so he can redistribute it and upload it again on a keyserver.

Note that you can upload your public keyring to a keyserver by the means of the menu command *Keyserver* → *Upload Public Keys*, but doing this without consent of the key owner is not considered good netiquette.

3.4.3. The Web of Trust

Now that we have discussed how you can establish trust by verifying keys from people you know, let's see how you can extend this zone. In the Web of Trust model, responsibility for key validation is delegated to other people you trust.

For instance, Alice would use her key 0xAAAAAAAA to sign Bob's public key 0xBBBBBBBB to certify that the particular public key is belonging to the individual called Bob. Bob has signed Carol's public key 0xCCCCCCCC. From this, Alice can infer that Carol's public key is valid (i.e. public key 0xCCCCCCCC purports to the individual called Carol) because there is a path of valid signatures from her public key to Carol's.

The *View* → *Signatures* menu item in Key Management, or the *View Signatures* button in Key Properties, allow you to view the signatures attached on a key i.e. by whom this key has been signed.

Participation to the Web of Trust is completely voluntary: you do not need to publicly sign other people's keys or to publish your key to successfully use GnuPG or Enigmail.

3.4.4. Owner Trust

A quick note in advance: “Owner trust” is one of the most misunderstood things in the *Web of Trust* context. In fact “Owner Trust” DOES NOT tell anything about the owners key or if it should be treated as valid or not.

The parameter “Owner trust” tells OpenPGP in how it should respect other keys signed by this owners key.

You alone decide how much you trust the practice of particular key signer. In the previous example, Alice could decide that she does not trust the keys signed by Bobs key, because she knows that Bob is happily signing any public key he gets his hands on without caring to verify the owner's identity. In this case she sets the owner trust level of key 0xBBBBBBBB (the key Bob uses to sign other people's public keys) to “None”. Carol's public key 0xCCCCCCCC will therefore not be calculated as “valid” in Alice's installation, if there is no other trust path.

Here, “Owner Trust” refers solely as Bob's capacity to properly validate public keys. It does not infer anything else concerning Bob as a person, such as his trustworthiness, his being a law-abiding citizen, or any of his moral qualities. It does not concern, neither, the content of Bob's messages being truthful or not.

You only should set “Owner Trust” for a person, if you have watched her/him in real life how careful she/he is checking the identity, the fingerprint and the mail address(es) of other key holders. The best reference to judge is how she/he did check YOUR identity.

There are five levels of Owner Trust:

- **Unknown.** Nothing can be said about the owner's judgement in key signing. This is the trust level initially associated to other people's public keys in your keyring.
- **None.** The owner is known to improperly sign keys.
- **Marginal.** The owner is known to properly sign keys.
- **Full.** The owner is known to put great care in key signing.
- **Ultimate.** The owner is known to put great care in key signing, and is allowed to make trust decisions for you.

You can set the level of trust of a particular key by selecting that key and choosing the option *Set Owner Trust* from Key Management, or from Key Properties itself.

You alone decide which level of trust to assign to a key, and the trust is assigned only locally. This is considered private information: it is not included with the key when it is exported, and is stored in a separate place from your keyring.

The trust level of your own key pair should be set to the maximum (*I trust ultimately*), if not – this can happen if you imported a key pair – you can correct this via the above procedure.

3.4.5. Criteria for Key validity

Keys signed by your key are automatically regarded as valid. This is the gold standard.

If you didn't sign a particular key, the *Web of Trust* decides whether a key is regarded valid or not. Within this model, the Owner Trust you set on the keys which signed a particular key is used to calculate their validity.

A side note: The owner trust level is your individual setting; as a consequence the calculated validity of a key is unique for your local installation of OpenPGP. That is, a key you consider valid might be considered invalid by someone else.

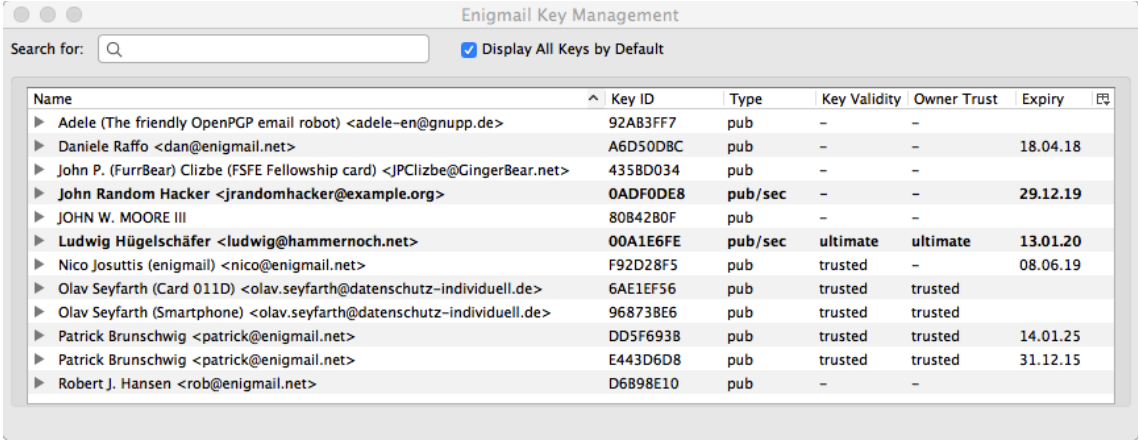
A key is considered valid if it has been signed by at least one key set to full owner trust, or by at least 3 keys with marginal owner trust setting. In addition to that, the path leading from that key back to your own key must be 5 steps or shorter.

This explanation is quite abstract, so here is an example taken from the real world:

I have signed the key of Patrick Brunschwig, our project head. I didn't meet Nic Josuttis, another member of our team in person. I've seen that Patrick carefully verifies other keys before signing, so I set the owner trust on his key to "full". Now, as he also signed Nico's key, Nico's key is regarded as "valid" in my keyring.

Ultimate owner trust bypasses any restrictions on how many fully trusted signatures are needed to make a key valid: any key signed by an ultimately trusted key is always valid. You can set ultimate trust to a particular key you want to allow to make trust decisions for you. This should only be done with extreme caution.

You can see the calculated validity and your assigned owner trust value in the Key Management or Key Properties window, in the fields *Key validity* and *Owner trust*. You may have to expand all columns to get a similar view:



The screenshot shows the Enigmail Key Management window. At the top, there is a search bar and a checkbox labeled "Display All Keys by Default" which is checked. Below this is a table with the following columns: Name, Key ID, Type, Key Validity, Owner Trust, and Expiry. The table contains several entries, including Adele (The friendly OpenPGP email robot), Daniele Raffo, John P. (FurrBear) Clizbe, John Random Hacker, JOHN W. MOORE III, Ludwig Hügelschäfer, Nico Josuttis, Olav Seyfarth (Card 011D), Olav Seyfarth (Smartphone), Patrick Brunschwig, and Robert J. Hansen.

Name	Key ID	Type	Key Validity	Owner Trust	Expiry
▶ Adele (The friendly OpenPGP email robot) <adele-en@gnupp.de>	92AB3FF7	pub	-	-	
▶ Daniele Raffo <dan@enigmail.net>	A6D50DBC	pub	-	-	18.04.18
▶ John P. (FurrBear) Clizbe (FSFE Fellowship card) <JPClizbe@GingerBear.net>	435BD034	pub	-	-	
▶ John Random Hacker <jrandomhacker@example.org>	0ADF0DE8	pub/sec	-	-	29.12.19
▶ JOHN W. MOORE III	80B4280F	pub	-	-	
▶ Ludwig Hügelschäfer <ludwig@hammernoch.net>	00A1E6FE	pub/sec	ultimate	ultimate	13.01.20
▶ Nico Josuttis (enigmail) <nico@enigmail.net>	F92D28F5	pub	trusted	-	08.06.19
▶ Olav Seyfarth (Card 011D) <olav.seyfarth@datenschutz-individuell.de>	6AE1EF56	pub	trusted	trusted	
▶ Olav Seyfarth (Smartphone) <olav.seyfarth@datenschutz-individuell.de>	96873BE6	pub	trusted	trusted	
▶ Patrick Brunschwig <patrick@enigmail.net>	DD5F693B	pub	trusted	trusted	14.01.25
▶ Patrick Brunschwig <patrick@enigmail.net>	E443D6D8	pub	trusted	trusted	31.12.15
▶ Robert J. Hansen <rob@enigmail.net>	D6B98E10	pub	-	-	

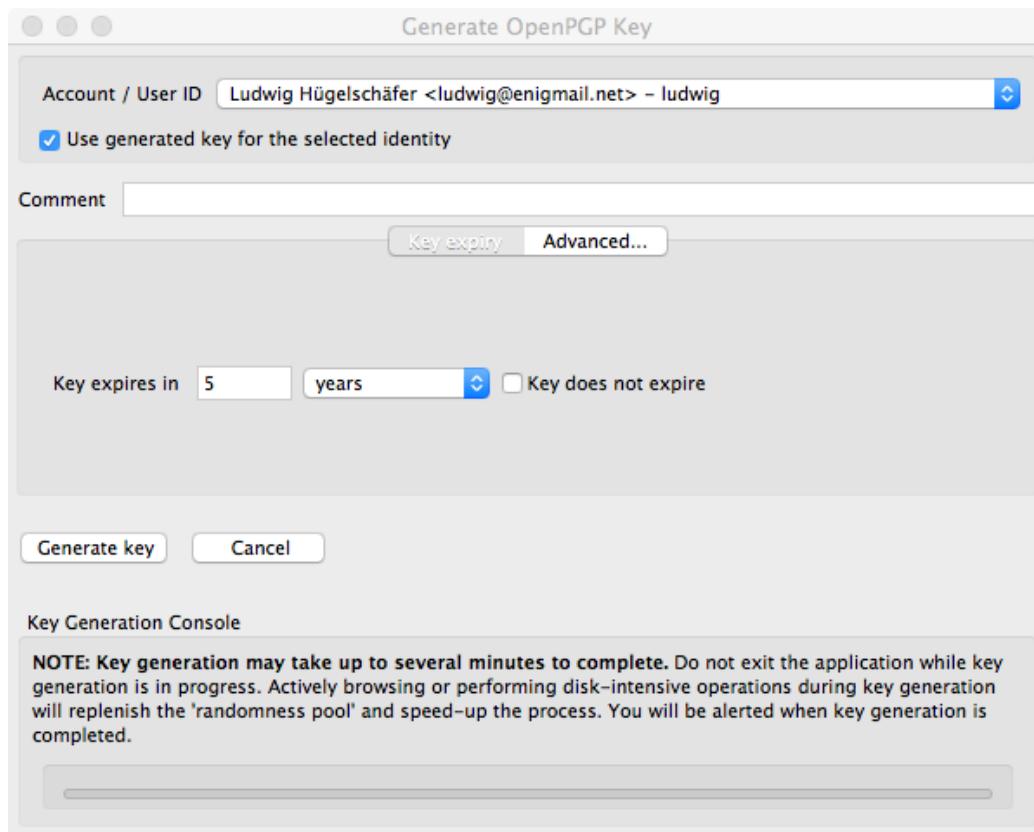
3.5. Importing an existing key pair

Perhaps you have already used GnuPG, Enigmail, or any other OpenPGP software in the past, and you generated a key pair in that occasion. If you didn't do so already using the Enigmail Setup Wizard you might want to use your existing key pair in this installation of Enigmail. Make sure you remember the passphrase of your old key pair, otherwise you will be unable to use it.

To import your key pair, choose *File* → *Import Keys From File* and select the file containing your key pair. Your key pair will be imported for you to use it.

3.6. Generating your own key pair

You need to own a key pair to join the community that communicates securely using GnuPG. If you already have one, you can directly jump to chapter 3.7 at page 46. Otherwise, you can create one at any moment by selecting *Generate* → *New Key Pair*. A window will open:



To successfully generate a key pair, you must proceed through the following steps.

3.6.1. Tell Enigmail which account to use

The *Account/User ID* drop-down menu shows all your email accounts and identities you have configured in Thunderbird: you must select one to associate to this key pair. Select whichever account will be receiving encrypted mail, and make sure the option *Use generated key for the selected identity* is ticked. Should you decide later that you want to change the email address associated to the key pair, or use the same key pair for multiple email addresses, you can do this easily; how to do that is explained in Section 3.1.2 at page 33.

In the *Comment* field you can write something about this key pair. This will help you distinguish keys for different purposes. The comment is optional and, if set, it will appear next to the user ID. Note that the comment will also appear in the exported public key that you distribute to other people.

3.6.2. Choose the expiry of the key

It may happen that some day you lose your private key or your passphrase, and therefore are unable to use your key pair. It may also happen that your private key gets compromised e.g. an intruder manages to have access to your computer and steal your key pair, or you could send someone your private key by mistake.

Furthermore, there might be some breakthrough in cryptanalysis or simply the discover of a weakness in a cryptographic algorithm, although this possibility is more remote. In this case, the cipher and key size you chose for your key pair many years ago could be unfit for offering valuable security in today's standards.

In all these unfortunate cases, it is a good idea to have defined a key pair that is valid only for a limited period of time.

The *Key Expiry* tab allows you to limit the lifespan of your key pair. After the allotted time, the public key will automatically be marked as invalid (expired), and people will be prevented to use it. The messages previously encrypted and signed with that key pair will still be decryptable and verifiable, though. Once your key pair has expired you can either extend the expiry time or you can generate another one, and distribute around your new public key. A good time lapse for key expiration is between 3 and 5 years. You may also choose to have a key that never expires, although we do not recommend to do so.

It is recommended, to create a revocation certificate that you can use at any time to mark the key as invalid; this will be explained further on.

3.6.3. Choose the key type and size

By clicking the *Advanced* tab you can choose some properties used for the generation of your key pair: the *Key size* (1024, 2048 or 4096 bits) and the *Key type*, which defines which cipher will be used (RSA or DSA & ElGamal). Basically, the larger the key size, the stronger the key, the greater the processing power requested for encryption/decryption. If you are going to send a lot of messages that will be decrypted on old computers or on mobile devices, you should probably choose a 2048-bit key. Otherwise, you don't really need to ponder your choice: the default option (4096-bit RSA key) offers excellent security coupled with a good usability, and will work fine. See also FAQ entry at page 89.

3.6.4. Start key generation

Once you made all these choices, simply click on the button *Generate key*.

3.6.5. Choose a passphrase

Upon clicking on "Generate Key" you will be asked by the GPG-agent two times what the secret passphrase should be for your new key pair. Wait a moment and read why you should take the passphrase seriously:

Your private key is all you need to send signed messages and decrypt messages that you receive on your selected email account. Should the private key fall in other hands, this would allow someone else to sign messages on your behalf and decrypt messages that were supposed for your eyes only.

Luckily, GnuPG uses an additional layer of protection: the private key is protected with a secret passphrase. Choose something for your passphrase that is easy to remember but very hard for someone to guess. Read Section 8.1 at page 93 for some suggestions on how to choose a good passphrase.



If you forget your passphrase, you will be unable to use your private key. There is no way to recover the passphrase. Your only hope is to try to remember what the passphrase was. This is a security feature of GnuPG and cannot be circumvented.

After entering your passphrase twice, the key will be generated.

3.6.6. Key generation process

Your computer makes use of a great quantity of random numbers in order to create the key. You may speed up the process by using the computer in the meanwhile, or simply by randomly typing on the keyboard or wiggling the mouse around to generate more randomness. On a modern computer, a standard 4096-bit RSA key pair takes no more than a dozen seconds to generate.

Congratulations! You just created your first key pair.

3.6.7. Generate the revocation certificate

As we just said, it may happen that your private key gets lost or compromised. In this case it is of crucial importance to have a revocation certificate for that specific key pair.

Once it finishes generating your key, Enigmail will offer you to also generate such a certificate. If you accept (which we recommend), click on *Generate certificate*. You will be asked where to save the revocation certificate file. The revocation certificate file has an ASC extension to indicate it's in ASCII-armored format. This is the common file format for exported certificates, key pairs, public keys, and all exported GnuPG output in general.

Keep the revocation certificate in a safe place and not on the same computer you installed Enigmail, so if you even lose your key pair (e.g. due to a hard drive crash) you don't lose the revocation certificate with it. An external hard drive, where you also put a backup copy of your key pair, will be fine.

The revocation certificate can be used to invalidate your key pair at any time. You do not need to type your passphrase or physically have in your keyring the key pair you are revoking, so this will work great even if you forgot your passphrase or accidentally deleted your key pair. However, keep in mind that anyone having access to the revocation certificate can revoke your key pair! You may also generate the revocation certificate at any later time by selecting your key pair and choosing *Generate* → *Revocation certificate*.

3.7. Revoking your key pair

In the unfortunate event that you lose your key pair or feel it has been compromised, you should revoke it. This can happen in many ways:

- Somebody not authorized got access to your computer
- Somebody not authorized got access to your key backup
- Your backup media is lost
- The used key algorithm has been found to be cryptographically broken

You can use the revocation certificate that you generated in advance (or at least you should have done so) to invalidate the key pair. Select *File* → *Import keys from file* from the Key Management window and choose the ASC file containing your revocation certificate.

If you do not have your keys any longer but the revocation certificate, then you must first get your public key, e.g. from a keyserver or a mail correspondent. Then, as a second step you can import your revocation certificate via *File* → *Import keys from file* from the Key Management window and choose the ASC file containing your revocation certificate.

If you did not generate the revocation certificate in advance, you can revoke the key on the fly, provided that you still have your key pair and you remember your passphrase. To do so, select the key pair you want to revoke and click on *Edit* → *Revoke key*. This effectively creates a revocation certificate and imports it in one shot.

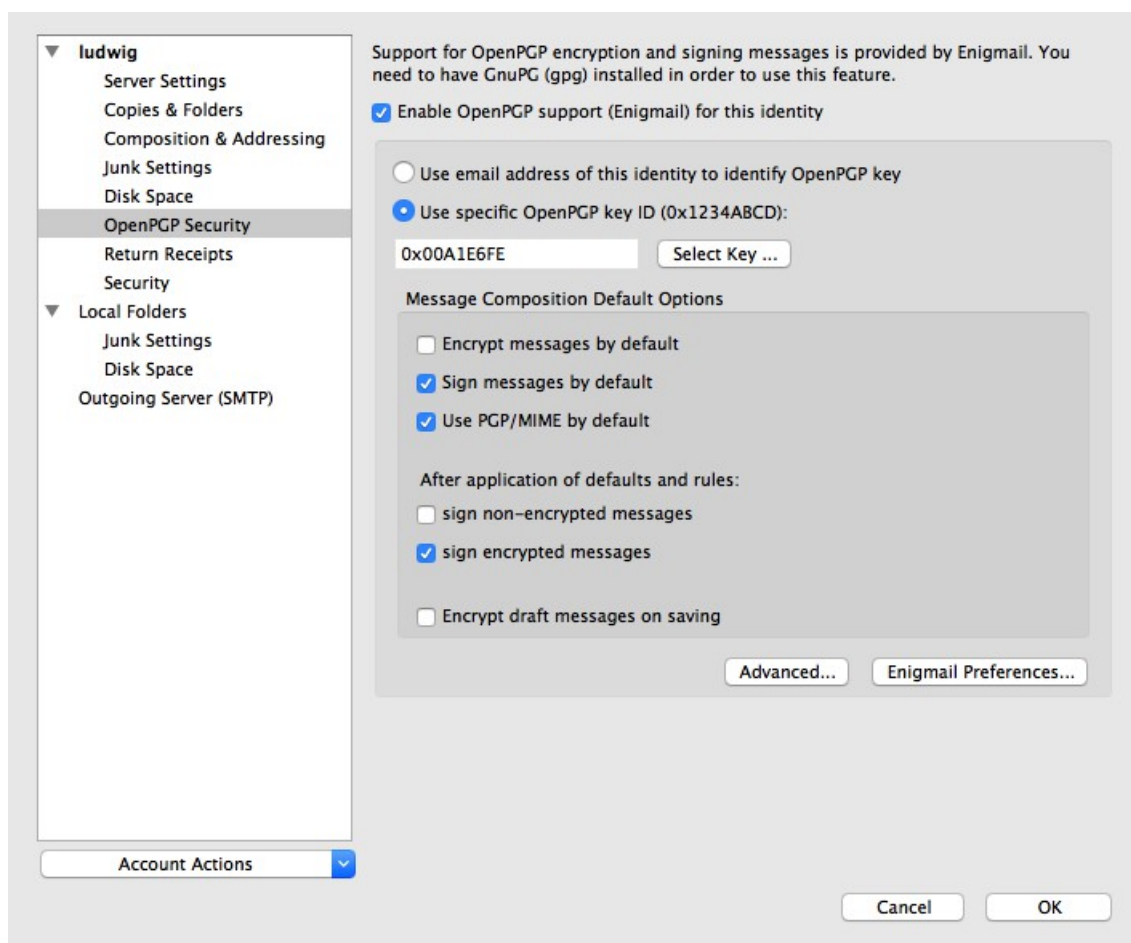
Send the revoked key to your contacts to warn them not to use it any more. If you published your public key on a keyserver, remember to upload again the revoked key to it.

4. SIGNATURE AND ENCRYPTION

You have generated your own key pair and have imported other people's public keys, so you are now able to exchange secure emails with them. But first, you must ensure that your account is correctly set up to use Enigmail capabilities.

4.1. Account settings

Launch Thunderbird and choose *Tools* → *Account Settings...*, and then click on *OpenPGP Security* from the email account settings:



If you have multiple identities enabled, you can (and should) set these OpenPGP options for each identity. You will do this from the Identity Settings window *<account name>* → *Manage Identities...*, which after the Enigmail installation will have a new *OpenPGP Security* tab with the same options as

above.

When configuring the OpenPGP options for your email account, first make sure the option *Enable OpenPGP support (Enigmail) for this identity* is turned on. This is necessary in order to send signed or encrypted email on behalf of this account, and to configure Enigmail. This does not disable signature verification and decryption, which is account-independent.

You need to let Enigmail know which key to use with this account. By choosing *Use email address of this identity to identify OpenPGP key*, Enigmail will automatically select the key pair which lists amongst its user IDs the email address associated with this account. Do not select this option if you have more than one key pair with the same user ID.

The recommended and failsafe method is to explicitly specify a key pair by choosing *Use specific OpenPGP key ID (0x1234ABCD)*: then enter in the field the ID (preended by 0x) of the key pair you want to use, or simply click on *Select Key...* and select the desired key pair.

You can then set some default options when composing a message from this email account:

- *Encrypt messages by default*: always tries to encrypt your messages.
- *Sign messages by default*: always tries to signs your messages.
- *Use PGP/MIME by default*: always tries to use the PGP/MIME format for your mails

Starting with this defaults, the per-recipient rules (see chapter 5.2 at page 81) will be processed. If you're a new user, there will be no per-recipient rules yet, so you start by this.

Having the results from the rules which may have modified the above defaults, you can set some final orders to enigmail:

- *Sign non-encrypted messages* enables signing automatically if encryption is not active at the same time.
- *Sign encrypted messages* activates signing automatically if encryption is active at the same time.

These will be the default options, unless modified manually. If you change the identity while composing a message, signing/encryption will be activated or deactivated according to the above options for the chosen identity, unless you have modified the signing or encryption status manually.

By default, Enigmail uses a standard called inline PGP for signed and/or encrypted messages. Checking the option *Use PGP/MIME by default* tells Enigmail to use the PGP/MIME standard instead of inline PGP. PGP/MIME

offers additional features such as attachments encryption together with the message body, and support for encryption of messages that use HTML format and special character sets. Unfortunately PGP/MIME is not supported by all mail clients; those that currently are compatible with it are Enigmail, Apple Mail, Becky, Evolution, KMail, Mulberry, Claws Mail, The Bat!, Balsa, Gnus, Whiteout Mail and many more.

You can click on the *Advanced...* button to set some advanced options. The first two options add a new mail header in the message, containing OpenPGP information about your key:

- *Send OpenPGP key ID* adds the mail header `OpenPGP: id=key_id` which mentions the key ID specified in *Use specific OpenPGP key ID*.
- *Send URL for key retrieval* adds the mail header `OpenPGP: url=url` which mentions the URL from where your public key can be retrieved. If you enable this option, you must also specify the URL by typing it into the field. This header actually is not used for any purpose of key retrieval, e.g. even if it is set Enigmail won't fetch the public key at the specified URL.

Finally, *Attach my public key to messages* automatically attaches your public key to any message you send.

While you're in Account Settings, this is a good opportunity to talk about some problems with HTML formatted mail in conjunction with OpenPGP secured mails.

HTML messages cannot be handled well, if you are using the OpenPGP inline format. The PGP/MIME format, however, can deal flawlessly with HTML messages. Unfortunately, not every mail client can interpret the PGP/MIME format.

Therefore, for beginners the safest and recommended way to deal with this situation is to disable HTML format for outgoing messages. You can check that by choosing *Composition & Addressing* and make sure *Compose messages in HTML format* is unchecked.

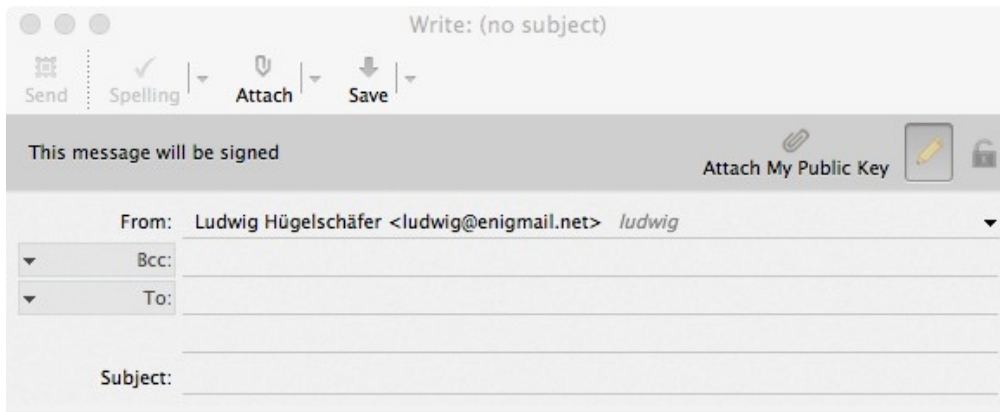
However, if you are sure that your recipient(s) can deal with the PGP/MIME format, you can leave HTML composition as default, but ensure that PGP/MIME is always activated for those messages.

Remember to disable HTML by hand every time you want to sign or encrypt the message you're writing in inline format. In Thunderbird if you have HTML enabled by default you can do this for the current message by holding down the Shift key when clicking on the *Write* or *Reply* button used for message composition.

4.2. Signature and verification

4.2.1. Signing a message

You are now ready to write your first digitally signed email message. From Thunderbird, click on the *Write* button as you normally would do. You will notice two new gadgets on the Write window: an additional toolbar below the standard toolbar, containing icons of a pen and a lock on the right side.



You can select the same options from the *Enigmail* menu item. As you already know by now, you can send a message signed, encrypted, or both encrypted and signed.






The pen icon (which means signature) and/or the lock icon (which means encryption) lights up to signal that the relevant option is on.

If you're opening a fresh Write window, the icons reflect the defaults for your account. As soon as you enter recipients, the icon state will be refreshed, as Enigmail processes the per-recipient rules and checks key availability in the background and displays the result in the icon state.






The icons can also be modified manually, that is, you can also click directly on the pen and the lock icons to respectively toggle signature and encryption. They will contain an exclamation mark to remind you that you changed the state manually.

Here you can see all possible icon states:

Signing:

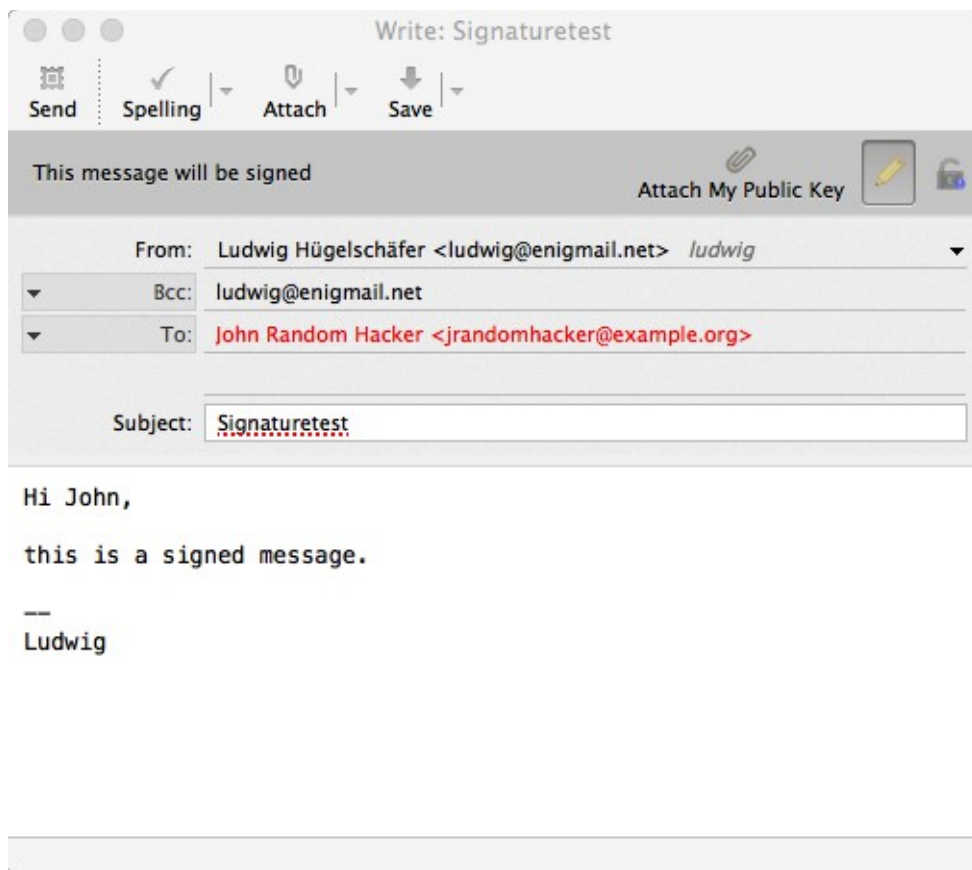
	Signing is active (according to defaults and rules)
	Signing is active (manually forced)
	Signing is disabled (account is not enabled for OpenPGP)
	Signing is inactive (according to defaults and rules)
	Signing is inactive (manually forced)

Encryption:

	Encryption is active (according to defaults and rules)
	Encryption is active (manually forced)
	Encryption is disabled (account is not enabled for OpenPGP)
	Encryption is inactive (according to defaults and rules)
	Encryption is inactive (manually forced)

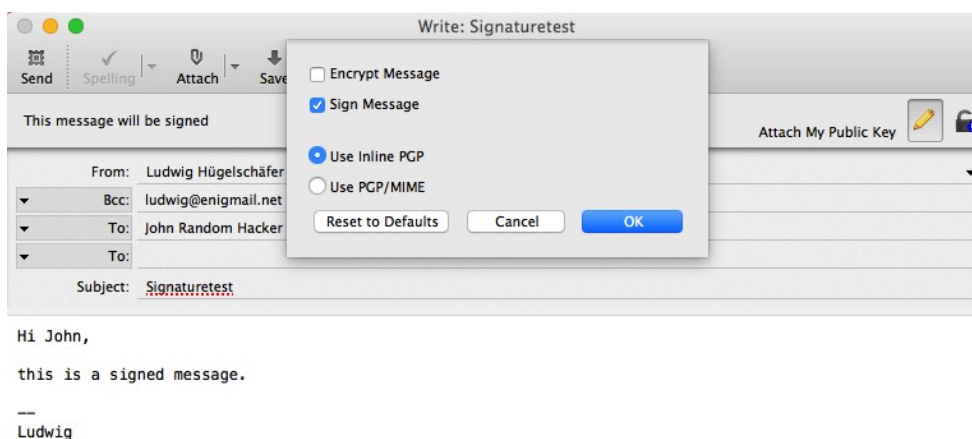
The icons have tooltips. Move the mouse pointer over them to get an explanation why Enigmail shows the displayed state.

The following figure shows the composition of a signed message:



Activate signing by either activating the pen icon in the Enigmail Toolbar or by selecting *Enigmail* → *Signing*,

Note: instead of clicking on the pen or lock icons, you may also click on the *This message will be signed/encrypted* and you are offered a dialog, where you also can change the signature/encryption and format status:



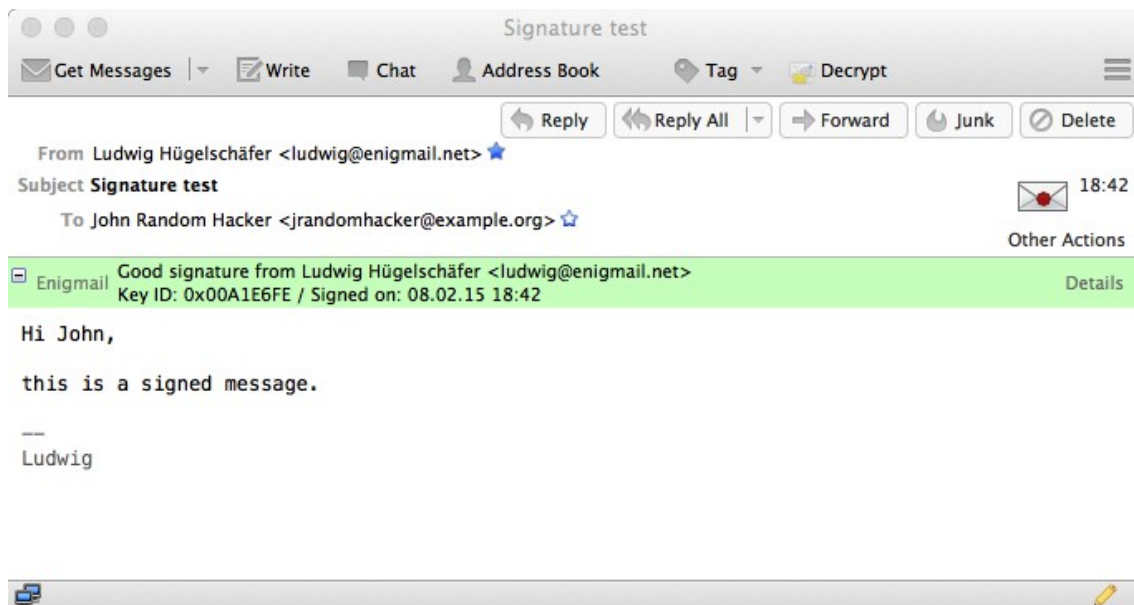
To finish, click *Send*. The message will be signed with the key specified in the Account Settings for the account you're currently using (the one shown in the

From: drop-down menu).

You will be asked for your passphrase, which is necessary for all operations concerning your private key such as signing messages, decrypting messages, and revoking or modifying properties of your key pair. It is also possible to cache your passphrase for a chosen amount of minutes so you won't have to type it every time: this can be set from *Enigmail* → *Preferences* → *Basic* → *Passphrase settings* and is explained in detail in Section 5.1.1 at page 72.

4.2.2. Verifying a signature

Now, if Thunderbird is set up so that a copy of outgoing emails is automatically saved in the Sent folder, it is possible to have a look at how the signed message looks like:



In this Message window, the Enigmail status bar shows up to indicate that the message is secured with OpenPGP.

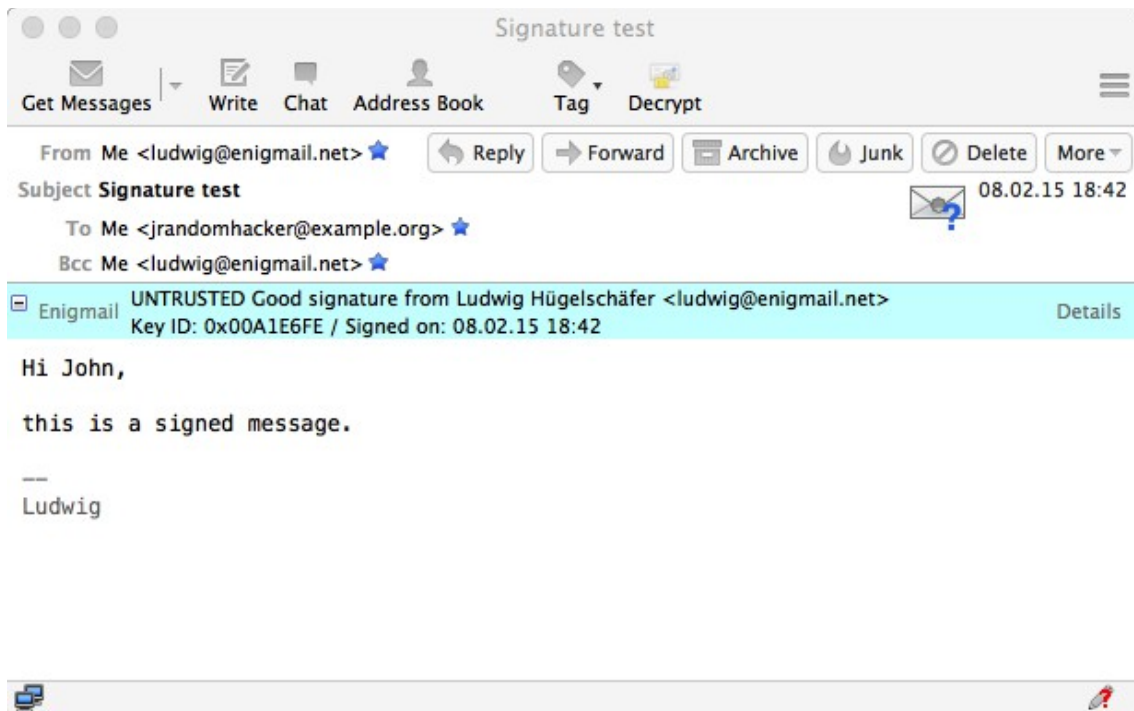
The status bar says that the message was correctly signed, and gives information about the sender's key, i.e. the key that was used to sign. This information is the user ID (the identity of the signer i.e. name and email address) and the key ID. The status bar also reports the date and time of the signature. You can expand or shrink the status bar by clicking on the expand gadget on the top left.

The Enigmail status bar is green to indicate that the sender's key is valid. In fact, in this case the sender's key is my own key, which has ultimate validity in my own Enigmail environment. Accordingly, a picture of a signing pen is shown in the status line at the bottom and a sealed envelope near the headers.

You can have more details about the signing key (in this case, the fingerprint of

the signing key) by selecting *Details* → *Enigmail Security Info...* or simply by clicking on the picture of the sealed envelope. *Details* → *Copy Enigmail Security Info* copies the security information to the Clipboard instead.

Now let's have a look at a signed message I received from `jrandomhacker@example.com`, assuming I have his public key:



The Enigmail status bar tells that the signature is correct, although untrusted. Note that the colour of the Enigmail status bar is now blue. Accordingly, a question mark is shown on the pen in the bottom right corner and the sealed envelope in the header section.

This means that John Random Hacker's public key is not fully valid in my public keyring, which is the default for freshly imported keys. Please see section 3.4.5, on how to get full validity.

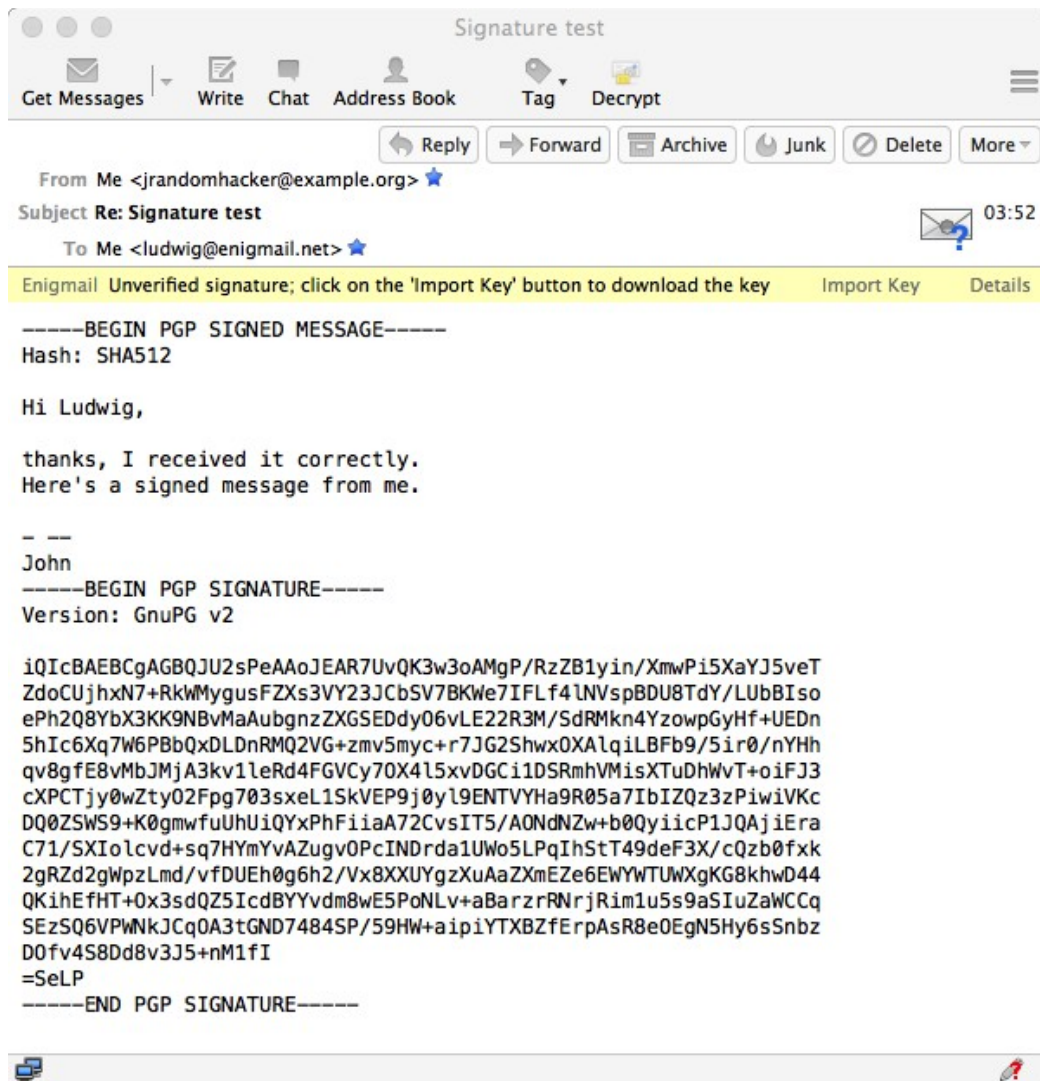
The most important point here is that it is verified that the message has been correctly signed with the specified public key.

From the *Details* menu you can operate directly on the sender's key:

- *View Key properties* shows all key details
- *View OpenPGP PhotoID* allows you to see the PhotoID, if any;
- *Sign Sender's Key...* allows you to sign the sender's key;
- *Set Owner Trust of Sender's Key...* allows you to set the "Owner trust" of sender's key. Please see section 3.4.4.

These are just shortcuts, you can do the same operations from Key Management as well.

What if I haven't had John Random Hacker's public key? In this case, the message would appear as such:



The message is signed, but the signature cannot be verified at all. This is also how a recipient that does not use Enigmail, nor any other OpenPGP software, will see the message. As expected, the original text is still readable: signature ensures authentication of the sender, not confidentiality of the message.

Note how OpenPGP manipulates the mail when signing. The original message is prepended by a line

-----BEGIN PGP SIGNED MESSAGE-----

and then it is specified which hash function has been used. Then there is the original message. Finally, the digital signature, embedded within two lines

-----BEGIN PGP SIGNATURE-----

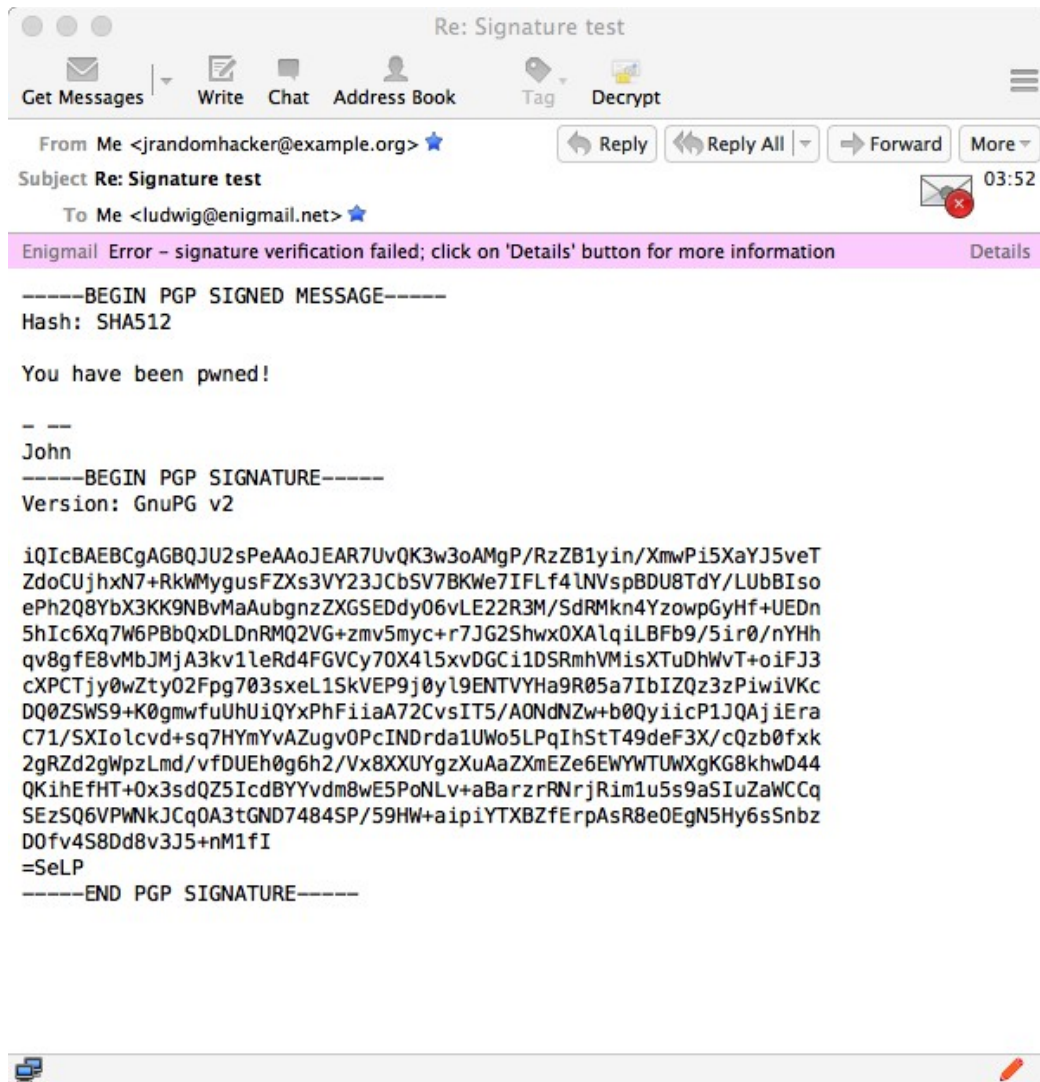
and

-----END PGP SIGNATURE-----

is appended to the message. Within the signature there may be line that

specifies the version of GnuPG used. It is also possible to put an additional comment line after the version line; this will be explained in Section 5.1.4 The lines starting with ----- are called *PGP headers*.

Finally, you might receive a mail that Enigmail shows as such:



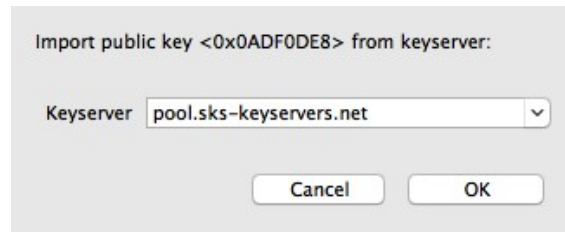
Note the purple status bar, the image of a red pen, and that the envelope icon is marked with a red "X".

This signature is invalid, which means that the message has been altered in transit. This alteration may even have happened within invisible characters like line breaks, spaces or tabs. Things like this can e.g. happen during the sending process or by improperly working mail servers.

In cases of an invalid signature, nothing can be said about the originality of the mail text. It may be unchanged or not, and you are advised to take it with caution. A good practice would be to ask the sender by encrypted mail for a statement about the contents.

4.2.3. Retrieving the key that signed the message

A nice feature of Enigmail is that it can import automatically the public key needed to verify a message. If you received a message for which you don't have the sender's public key, as shown in the figure at page 56, simply click on the *Import Public Key* button shown in yellow the status bar, and Enigmail will offer to download the key that was used for signing from a keyserver:



Just click on *Ok* and Enigmail will do that for you. The imported key will be added to your public keyring.

More often, you will receive someone's public key as an ASC file attached to the email message. In this case, importing the public key is just as easy: you only have to right-click on the attachment and choose *Import OpenPGP Key*.

Someone might also send you his public key embedded in the message text.

4.3. Encryption and Decryption

Now comes the interesting part: exchanging encrypted messages.

4.3.1. Encrypting a message

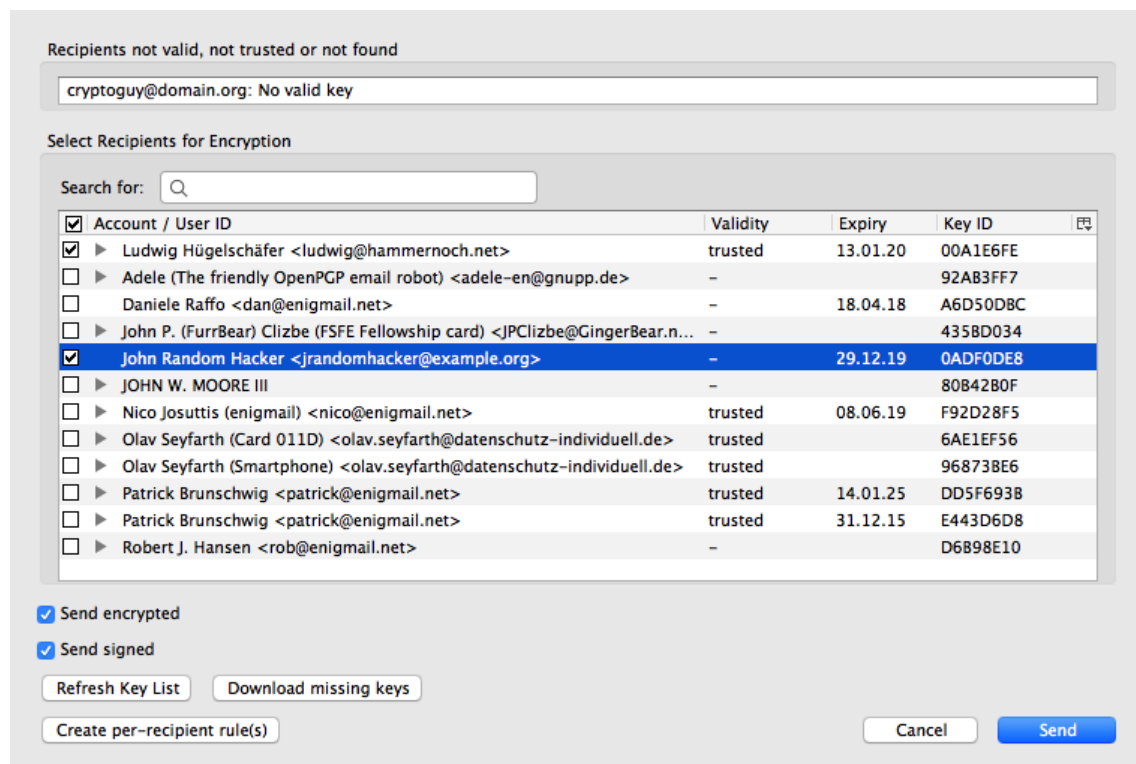
To encrypt a message, select the option *Encrypt Message* before sending, and make sure the lock icon in the Enigmail Status Bar is lit. It is common practice to also sign a message you're encrypting.

To send an encrypted message to someone, you need to have his public key. If you have it, the key is automatically selected: Enigmail searches your keyring and selects the public key that has a user ID that matches the recipient's address. (Note: If you have set per-recipient rules, these will be looked up first. You will learn about per-recipient rules in Section 5.2 at page 81)

This is done for each recipient. Recipient addresses are all those specified in the mail headers To:, Cc:, and Bcc:.

Additionally, the message is also automatically encrypted with your own public key, to allow you to read (from the Sent folder) the messages you sent.

As you see, this is pretty straightforward. But what happens if Enigmail is unable to select a public key for a recipient, for instance because you don't have it? In this case, Enigmail pops up the Key Selection window to ask you to select the key(s) by hand:



In the figure, I was trying to send an encrypted email to `cryptoguy@domain.org`, which let's imagine is set as an alias and forwards all mails to `jrandomhacker@example.com`. In this case, I would select John Random Hacker's public key, as shown in the figure, and click *Send*. The message would then be sent to `cryptoguy@domain.org` encrypted with John Random Hacker's public key.

If I had to send mail to `cryptoguy@domain.org` often, it would be worth creating a per-recipient rule that says "Encrypt all mail that is sent to the address `cryptoguy@domain.org` with the public key associated with address `jrandomhacker@example.com`". This can be done directly from the Key Selection window by clicking the *Create per-recipient rule(s)* button.

Alternatively, if John Random Hacker intends to use often his alias address, he should add the user ID `cryptoguy@domain.org` to his public key, and redistribute the updated key.

As you have learnt, a message can be encrypted with more than one public key. In fact, it is usually encrypted with at least two public keys: yours (to let you be able to read a copy of the message) and the recipient's.

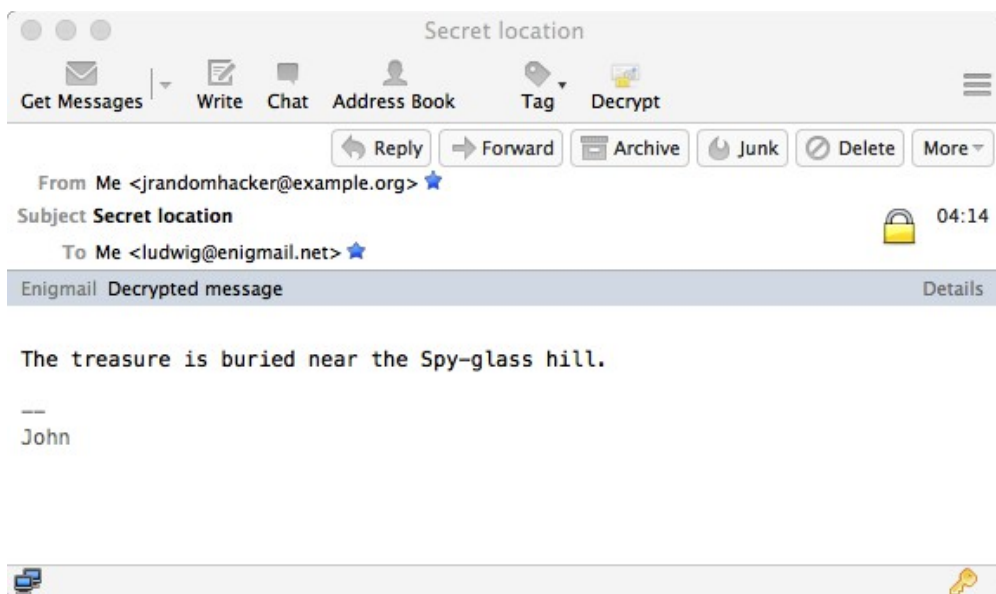
To be more precise, OpenPGP uses hybrid encryption. First it generates a random session key, and encrypts the message with the session key using a symmetric algorithm; then, for each intended recipient, it encrypts the session key with the recipient's public key and adds each encrypted session key to the encrypted message. It then internally builds an OpenPGP block, which includes a header containing the key IDs and user IDs of any public key the message has been encrypted with. Each recipient then receives the same OpenPGP block.

As a consequence, it is not possible to send to multiple recipients a message that is encrypted for some recipients and unencrypted for others. The message is sent out either encrypted or unencrypted for the whole list of recipients.

That being stated, you should not send encrypted messages to Bcc: recipients, because from the OpenPGP block each recipient is able to tell the identities of the others – hence defeating the purpose of the Bcc: field. While Enigmail is able to do some workaround to hide the Bcc: recipients from the header, as a side-effect this could block users of other products (e.g. PGP Corp.) from being able to decrypt the message.

4.3.2. Decrypting an encrypted message

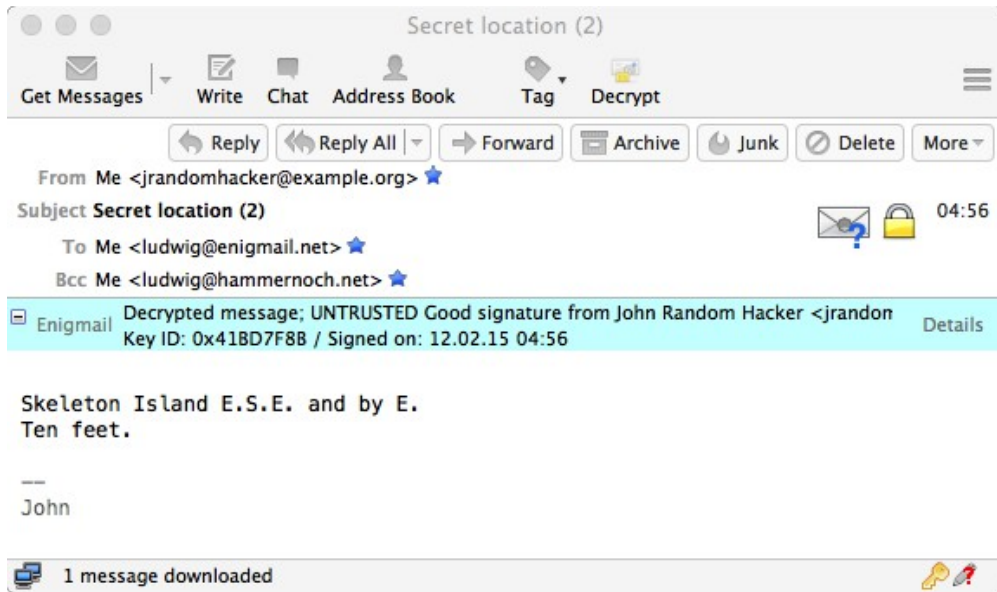
This is a message that John Random Hacker sent encrypted to me:



The Enigmail status bar is grey, saying “Decrypted message”, the lock in the headers bar, and the yellow key icon in the corner all indicate that the message was correctly decrypted.

By default, the message is automatically decrypted as it is opened. If you ever want to change this setting, deselect the option *Enigmail* → *Automatically Decrypt/Verify Messages*; then you can decrypt messages by hand by clicking the *Decrypt* button in the toolbar.

The previous message was encrypted but not signed. Here's how a message that is both signed and encrypted appears for you:

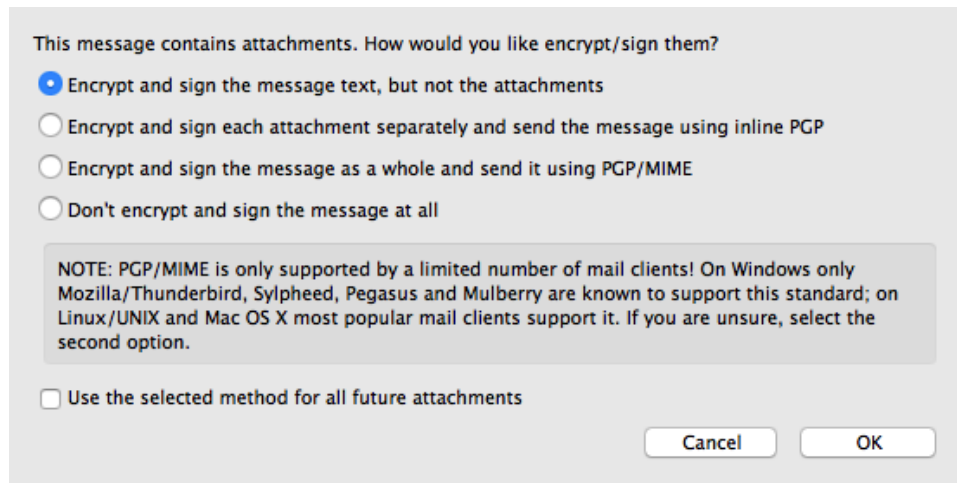


Note the additional (compared with a signed only message) text in the Enigmail status bar: "Decrypted message".

The colour of the status bar is blue and it says "Untrusted good signature". This means that the signature verifies, but John Random Hacker's public key is not fully valid in my public keyring, which is the default for freshly imported keys. Please see section 3.4.5 at page 41, on how to get full validity.

4.4. Handling attachments

When sending an encrypted or signed email message that has attachments, you will be given the choice how to encrypt/sign the attachments:



There are three mutually exclusive options:

- *Just encrypt/sign the message text, but not the attachments.* This will only protect the message text.
- *Encrypt/sign each attachment separately and send the message using inline PGP.* In this case, the signature for each attachment `filename.ext` will be stored in an additional attached file `filename.ext.sig`. If the attachment must be encrypted, it will be renamed as `filename.ext.pgp`. This is the safest method and most recommended.
- *Encrypt/sign the message as a whole and send it using PGP/MIME.* As said previously, the PGP/MIME standard is not supported by all mail clients, so the risk here is that the recipient could be unable to read the message.
- *Don't encrypt and sign the message at all:* Choose this, if you erroneously selected encryption.

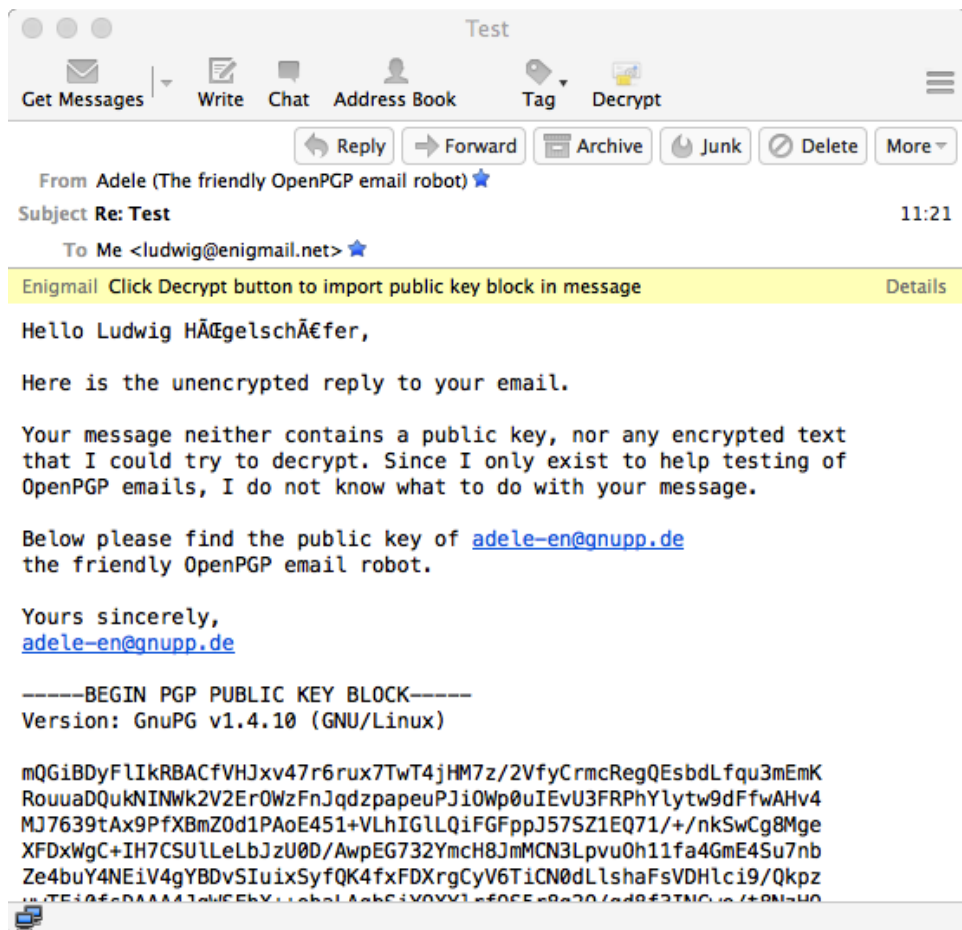
If you tick the option *Use the selected method for all future attachments*, Enigmail will remember your choice and won't ask you any more. To have Enigmail show you the options again, go to *Enigmail* → *Preferences* → *Advanced* and click on the *Reset Warnings* button.

When you receive an encrypted attachment, you can view it or save it simply by right-clicking on it and selecting, respectively, *Decrypt and Open* or *Decrypt and Save As...*

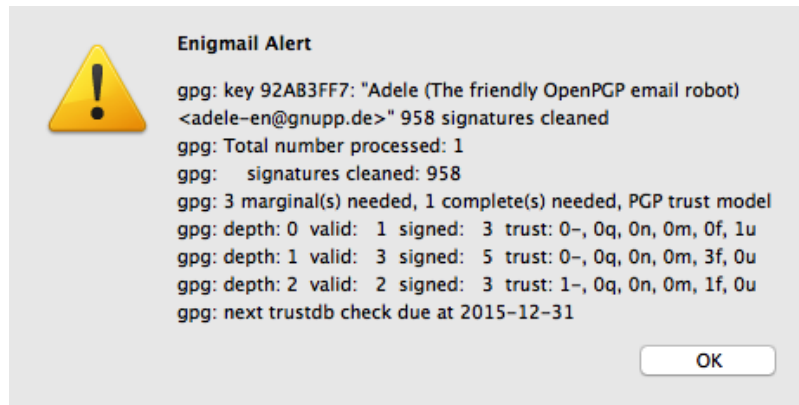
4.5. Practice with “Adele”

If you want to do some signature and encryption tests yourself, then you'll find a very patient correspondent in Adele, “The Friendly OpenPGP Email Robot”. Adele can be contacted at adele-en@gnupp.de and is an automated program that is able to receive and understand OpenPGP messages, and to reply to them accordingly in a very short time.

I sent a simple cleartext mail (unsigned, unencrypted) to Adele, and here's how she replied to it:

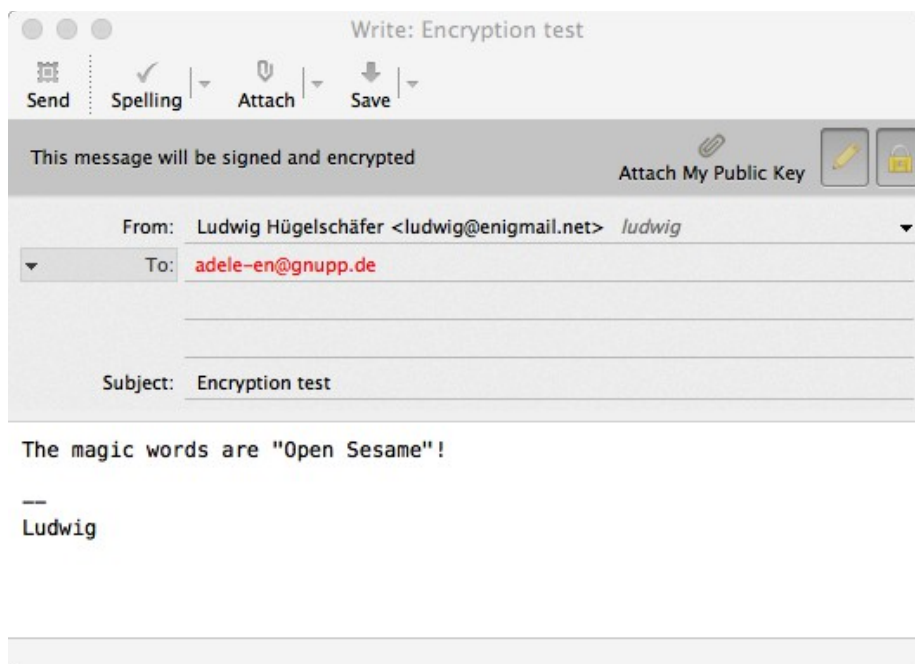


Here Adele complains that there was no public key attached to my message, so she doesn't know what to do with it. However, she provided me with her public key embedded in the message; note the OpenPGP key block in the mail body, and the yellow Enigmail status bar. Clicking on the *Decrypt* button and then confirming will import the public key into my public keyring:

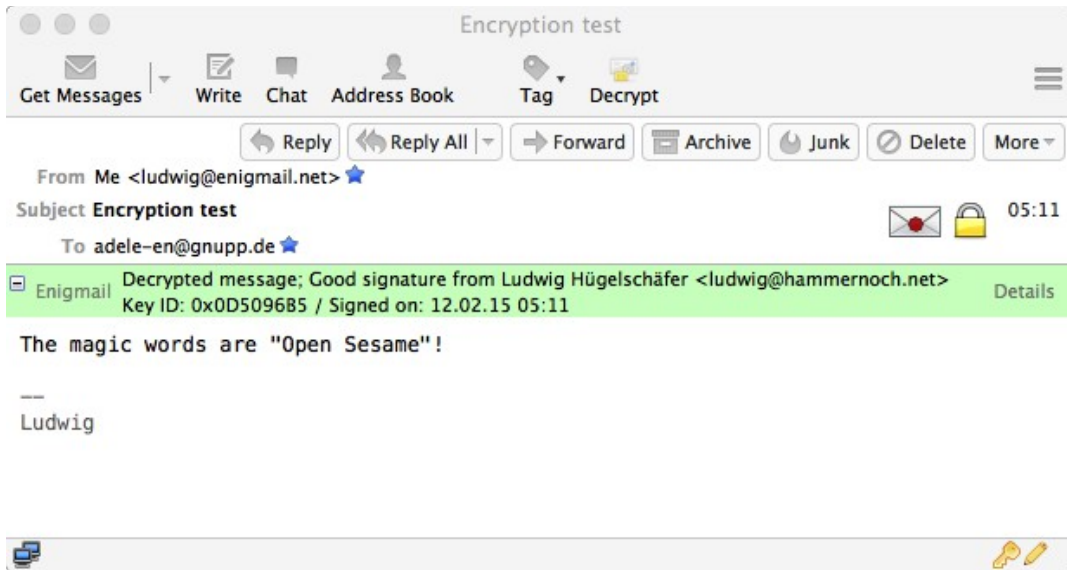


Adele's public key is now in my public keyring.

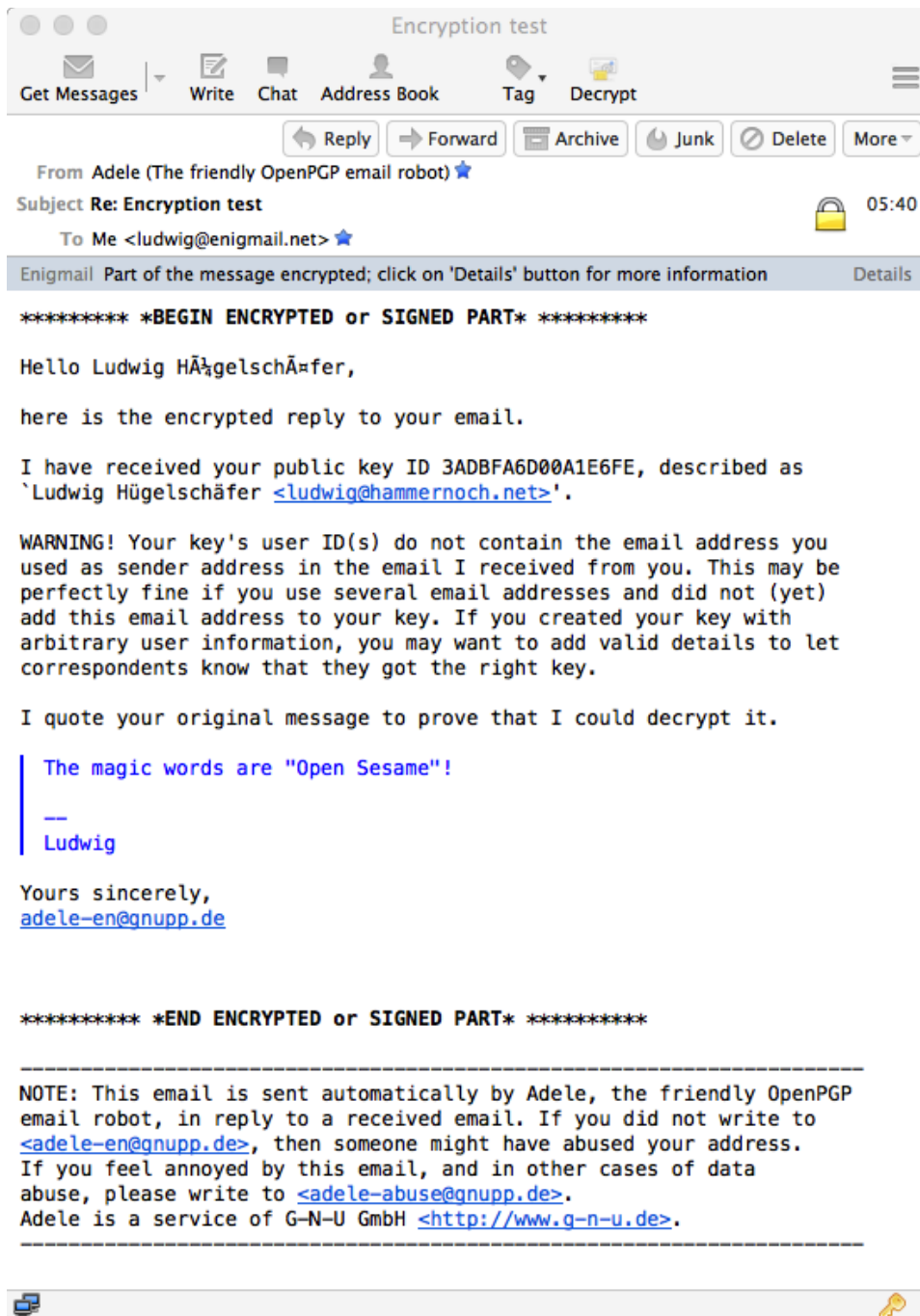
I can use Adele's services to test that my messages are encrypted and decrypted correctly. As you remember, I have imported Adele's public key in my keyring, and I am therefore able to send her an encrypted message:



If I look in my Sent folder there is my message, automatically decrypted as I open it. I am able to read it only because Enigmail, by default, encrypts any outgoing message with the sender's public key too. Shouldn't Enigmail do that, the message would look gibberish to me – even if I was the creator of the message. The next figure shows my own message, correctly decrypted:



A short time later, I receive Adele's reply:

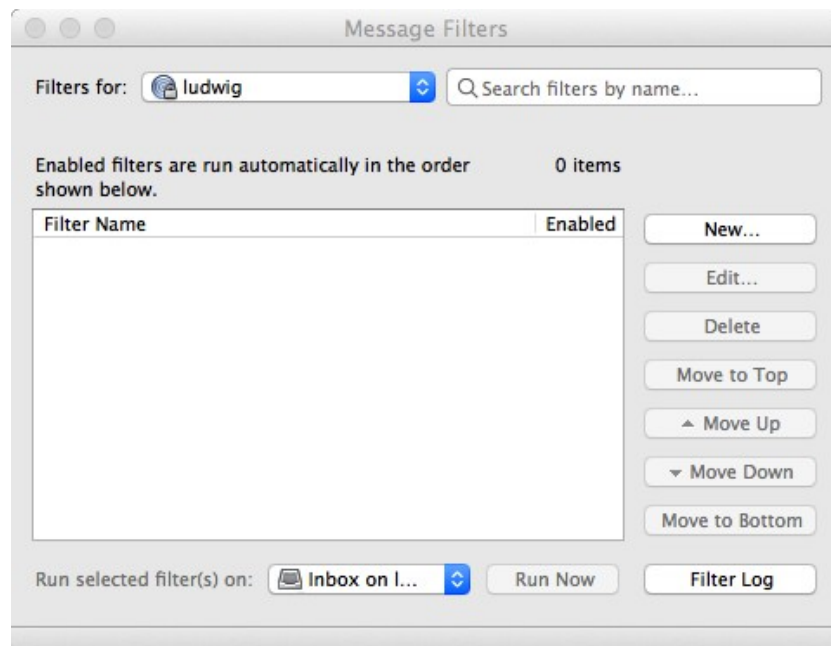


Please note that the Enigmail status bar warns that the mail body is only partly encrypted: Adele's message is, while the note at the bottom isn't. Enigmail marks the secured, i.e. signed and/or encrypted part, so that you can easily distinguish between them.

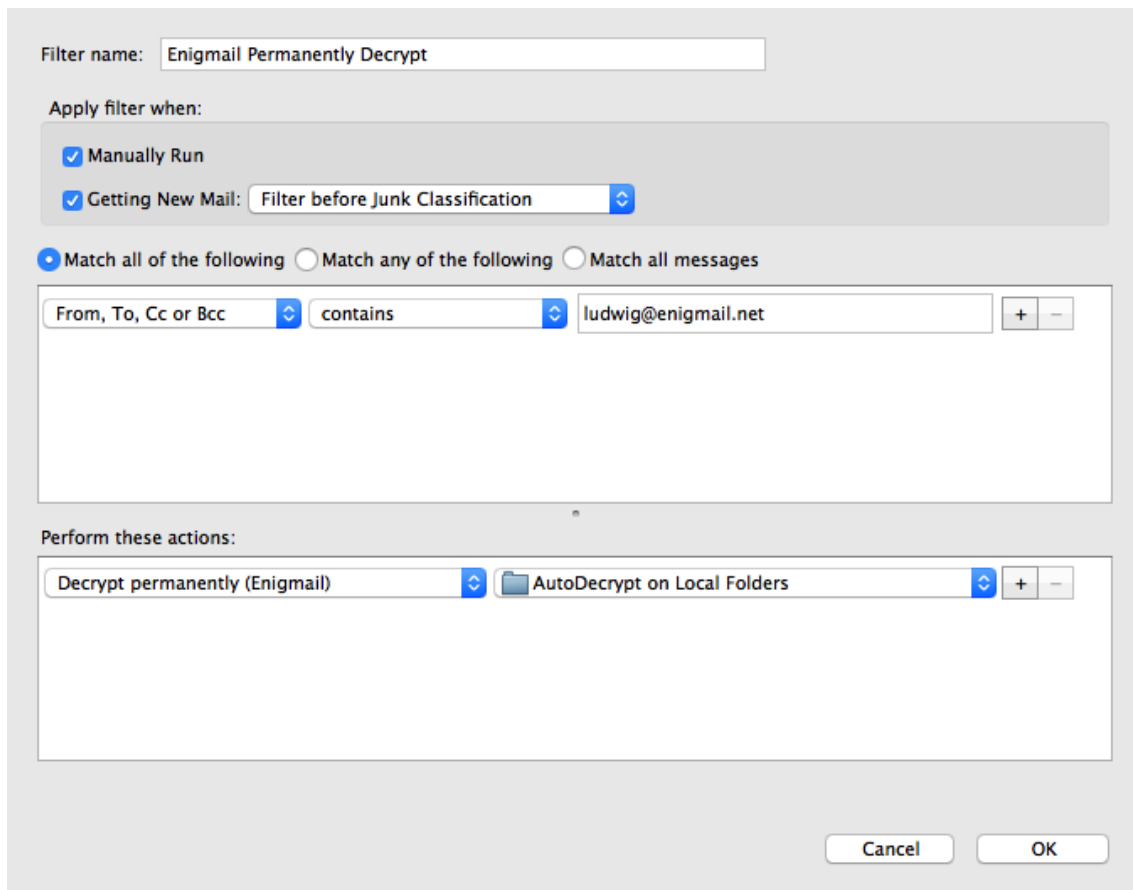
4.6. Permanent Decryption of messages

This feature was a long standing wish from users, but quite difficult to implement. Enigmail 1.8 now offers a way to permanently decrypt messages. It works when receiving messages from the mail server using the Thunderbird mail filter scheme.

To activate this feature, click on your account and then *Manage message filters*. The following window will open:



Click on *New* to create a new message filter:



The top line is a freely choosable name, so that you can distinguish it from other filters.

You can leave the *Apply filter when:* as it is set by default.

In the next section of the dialog you can enter the conditions under when the filter shall trigger. I have entered only one condition, naming my mail address in “contains” “From, To, Cc or BCC”, to apply this filter on every mail I receive under this address.

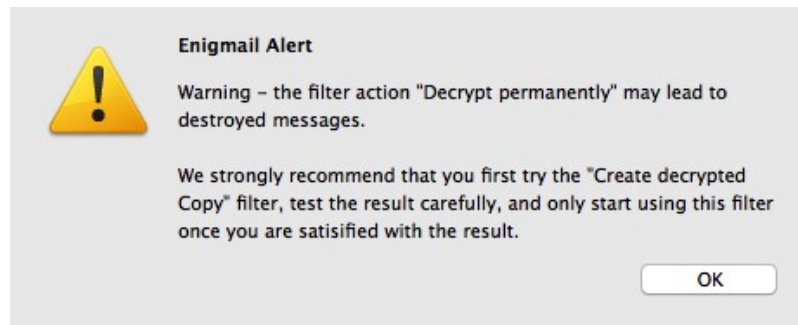
Up to now, this is like setting up a Thunderbird standard message filter.

Enigmail provides for two action types, which can be selected using the *Perform these actions* drop down menu:

- *Decrypt permanently (Enigmail)*
- *Create decrypted Copy (Enigmail)*

The first one decrypts the message and moves the resulting message in the folder selected using the drop down menu on the right side.

Clicking ok, Enigmail presents a warning:



This is because if there is any failure during decryption (e.g. messages that are encrypted on odd ways like S/MIME+PGP combined), the message is lost or corrupted. It is better to select *Create decrypted Copy (Enigmail)* and test the behaviour for some days or weeks. If anything goes wrong, you still have the original message and can decrypt it manually. If it works flawless, you can later change it to *Decrypt permanently (Enigmail)*.

4.7. Encrypted Subject?



Mail headers cannot be encrypted, nor included in the signature computation. Do not write any sensitive information in the Subject when sending an encrypted message.

One important point concerns mail header security. Signature and encryption applies to the mail body only – and also to attachments, if you chose so. That is, when you sign a message, no mail header (such as the Subject, Date, all Received headers, etc.) can be included in the signature. Also, when you encrypt a message, mail headers are not encrypted.

Most mail headers (e.g. the Date) are added by the Mail User Agent (mail client) only after that Enigmail processes the payload; other mail headers (e.g. the Received) are subsequently added by the Mail Transfer Agents at each hop of the way from sender to receiver; finally, other mail headers are added by the Mail Delivery Agent at the endpoint. Even mail headers that are user-set at the time the message is composed (e.g. the Subject) may be legitimately modified by anti spam or antivirus applications on the destination server.

This is to say that mail headers change in transit (therefore they cannot be signed), and they must be in cleartext (therefore they cannot be encrypted) in order for the mail message to be processed by intermediary routers and delivered.

This is not a limitation of Enigmail, neither of GnuPG. The OpenPGP standard does not describe a way to sign or encrypt the Subject or, for this purpose, any other mail header, and there is no other standard yet, which will allow this.

Thus, please remember that today mail headers aren't and cannot be secured, and they could be snooped and forged in transit like any cleartext mail message.

Note that, for the same reason, Enigmail will not sign and/or encrypt a blank message: the message body is empty, so there's nothing Enigmail can process.

5. PREFERENCES

Enigmail can be fine-tuned to tailor your needs. This chapter illustrates the many configuration options of Enigmail.

If you use GnuPG and configured it manually, please note that these preferences will override any similar entry in the GnuPG configuration file `gpg.conf`.

5.1. Setting the preferences

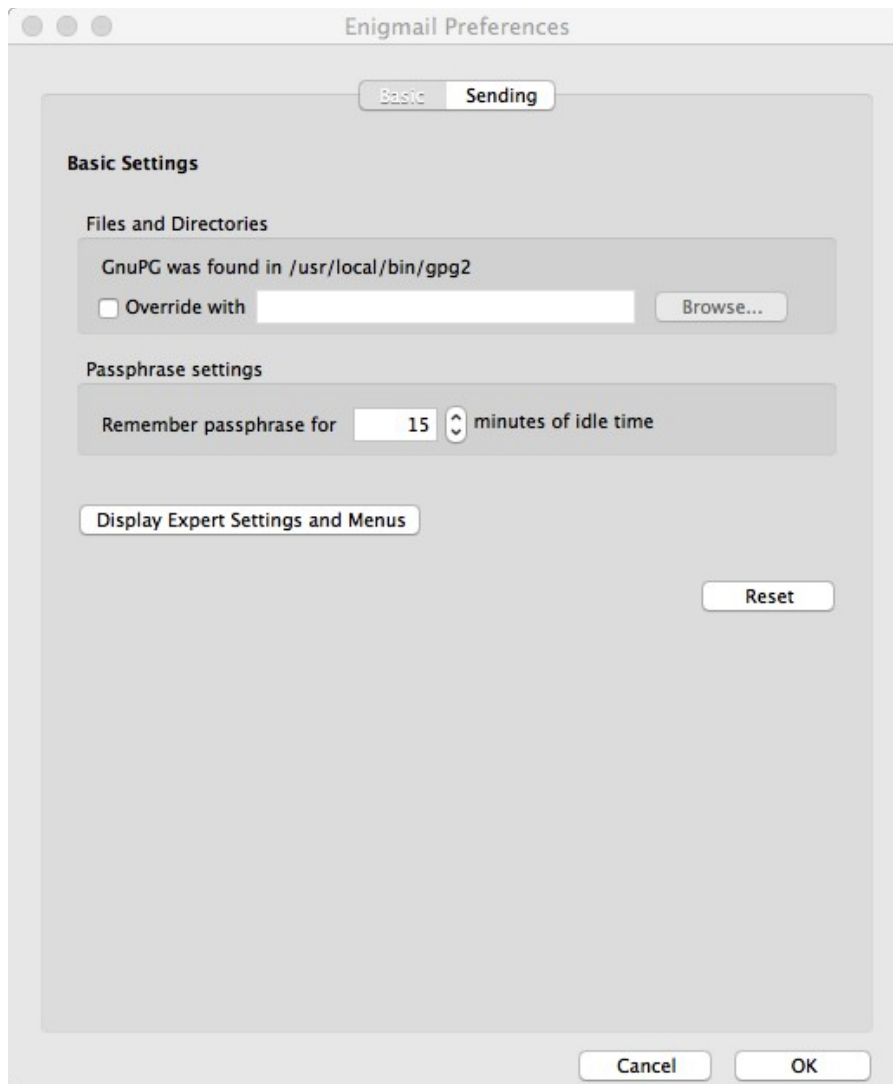
5.1.1. Basic

To access the Enigmail preferences, select *Enigmail* → *Preferences* from the menu bar. This will initially bring up the Basic preferences, which control the basic functioning of Enigmail.

In *Files and Directories*, it is shown where GnuPG was found. Enigmail tries to locate automatically the GnuPG executable file upon its start. Typical locations are `C:\Program Files (x86)\GNU\GnuPG\gpg2.exe` for Windows, and `/usr/local/bin/gpg2` for Linux and Mac OS X.

If however Enigmail can't manage to find GnuPG, or you want to specify that location manually, tick *Override with* and enter the path to the GnuPG executable file.

You will be asked for your passphrase every time it needs to access your private key, for instance whenever you sign, decrypt, or change your key pair properties. It is often cumbersome to have to type the passphrase all the time, and you might be tempted to choose a passphrase that's short and simple to type, which is a bad idea. Instead, you should set a caching time for your passphrase.



You can do this by entering the desired number of minutes in the field *Remember passphrase for [] minutes of idle time*. In the figure, you will not be asked for the passphrase for 15 minutes.

You will be asked again for the passphrase again when one of these events occurs:

- The specified caching time has expired;
- You restarted the computer.



Do not leave your computer unprotected while the passphrase is stored in the cache!

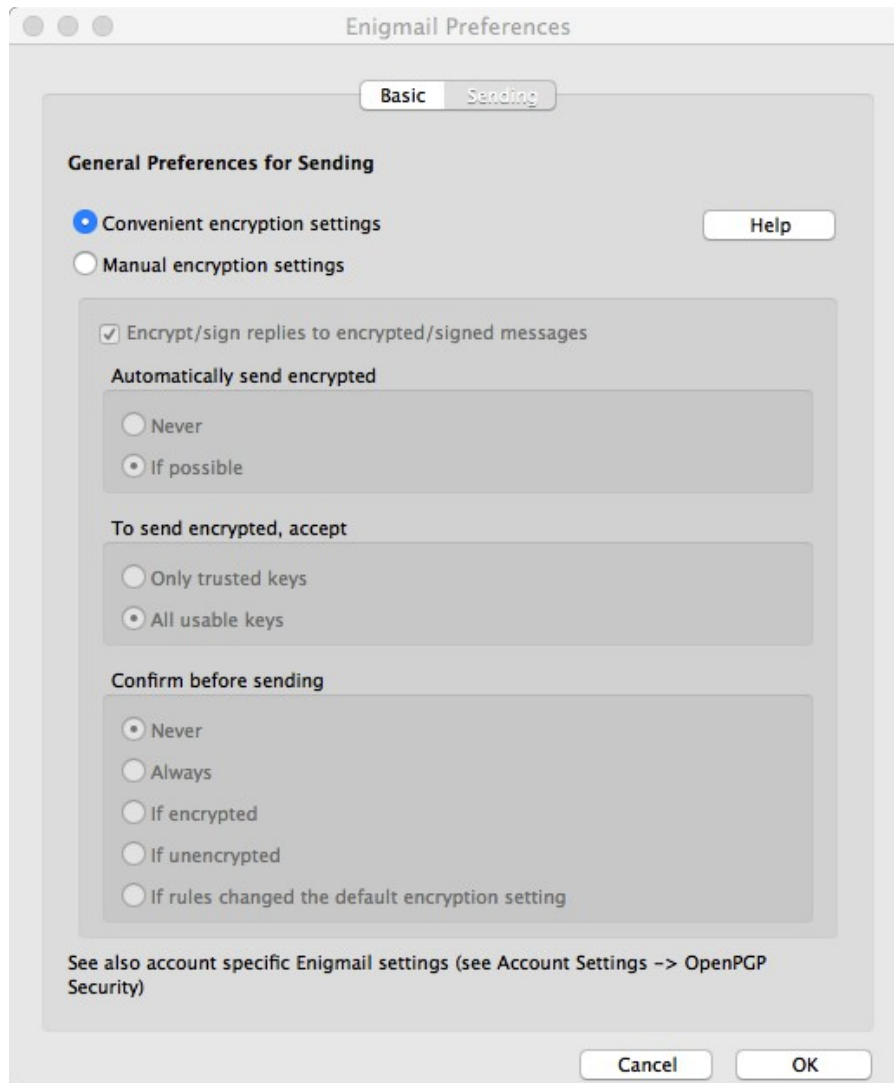
Read more about passphrase handling in paragraph 8.4 at page 100.

Finally, the *Display Expert Settings / Hide Expert Settings* toggle button permits to access the expert preferences. It activates three additional tabs with the

expert settings directly in the same window.

5.1.2. Sending

The “Sending” tab of the preferences shows the options for sending encrypted mails. It is always accessible, even if you haven’t enabled the expert settings.



These settings define how Enigmail will behave when sending secured mail.

By default, “*Convenient encryption settings*” are enabled. This ensures an easy start for beginners. Advanced users may want to change to “*Manual encryption settings*”. The selected buttons in the above screenshot show what makes up “Convenient encryption”.

Encrypt/sign replies to encrypted/signed messages (checked by default):
Automatically switches on encryption/signing when composing a reply to

an encrypted/signed message. This is a smart thing to do, especially if you quote the original message.

Automatically send encrypted

Never:

If checked, never try to automatically send encrypted

If possible (checked by default):

Sends encrypted when you have the public keys of all recipients (everyone in TO, CC and BCC).

To send encrypted accept:

Usually, GnuPG does not allow you to encrypt a message to keys that are not 'valid'. Here you select, if you want to keep this behaviour or not:

Only trusted keys:

keeps GnuPGs behaviour.

All usable keys (checked by default):

will accept every key which is not expired, revoked or disabled

Confirm before sending:

Controls a confirmation dialog before sending any message, so that you can check the signing, encryption, and S/MIME status.

Never (checked by default):

It is advisable to select this option if you send S/MIME signed or encrypted messages from time to time.

Always:

Always prompt for confirmation

If encrypted:

Prompt only when mail is going to be sent encrypted

If unencrypted:

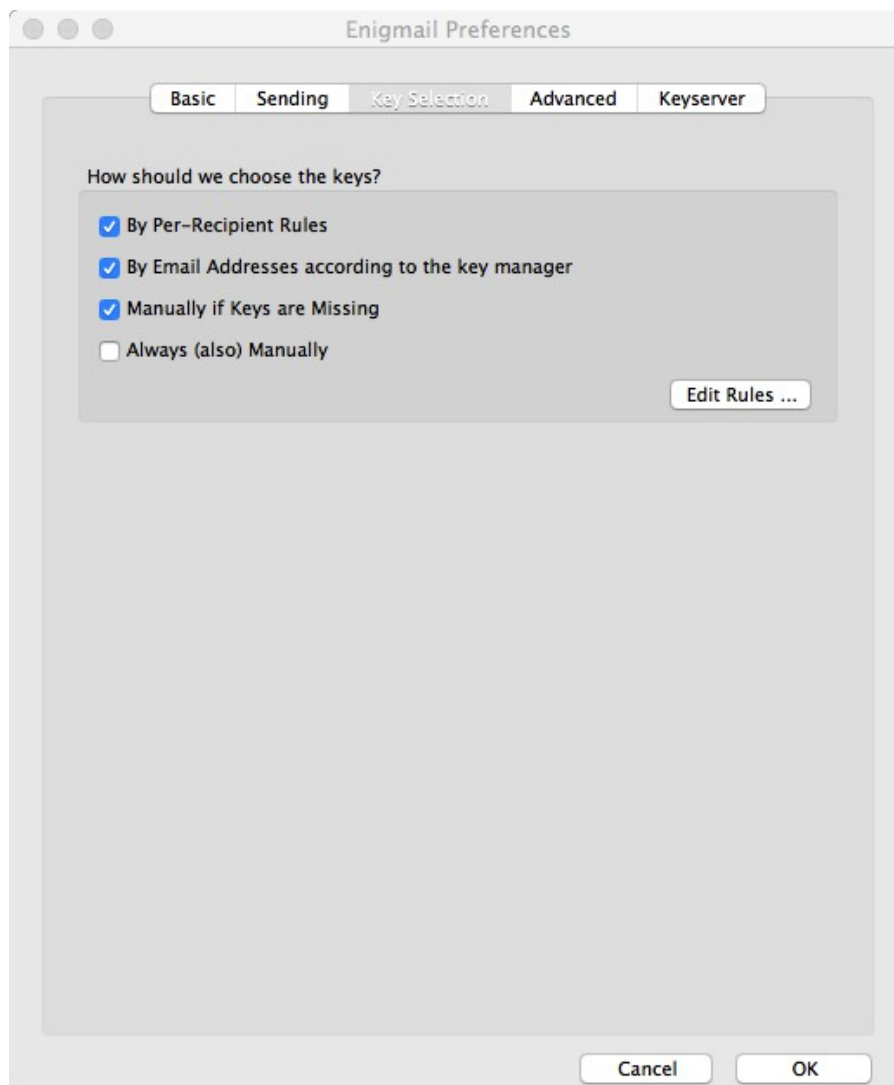
Prompt only when NOT encrypted.

If rules changed the default encryption setting:

Prompts only when Per-Recipient-Rules changed the default encryption setting. This is useful to detect when a rule did switch off encryption.

5.1.3. Key Selection

This is only accessible if you have enabled the expert settings in the “Basic” tab.



These settings define how Enigmail will select, for each recipient, the public keys to encrypt a message with. The following options will enable different ways to get the right key. More than one can be checked. They will be processed in the displayed order. If one of the options retrieves a result, the following options will not be processed any more:

- *By Per-Recipient rules*: chooses the key depending on per-recipient rules. (This is checked by default).
- *By E-Mail-Addresses according to the key manager*: selects the key with a user ID matching the recipients mail address. (This is checked by default).
- *Manually if keys are missing* instructs Enigmail to pop up the Key

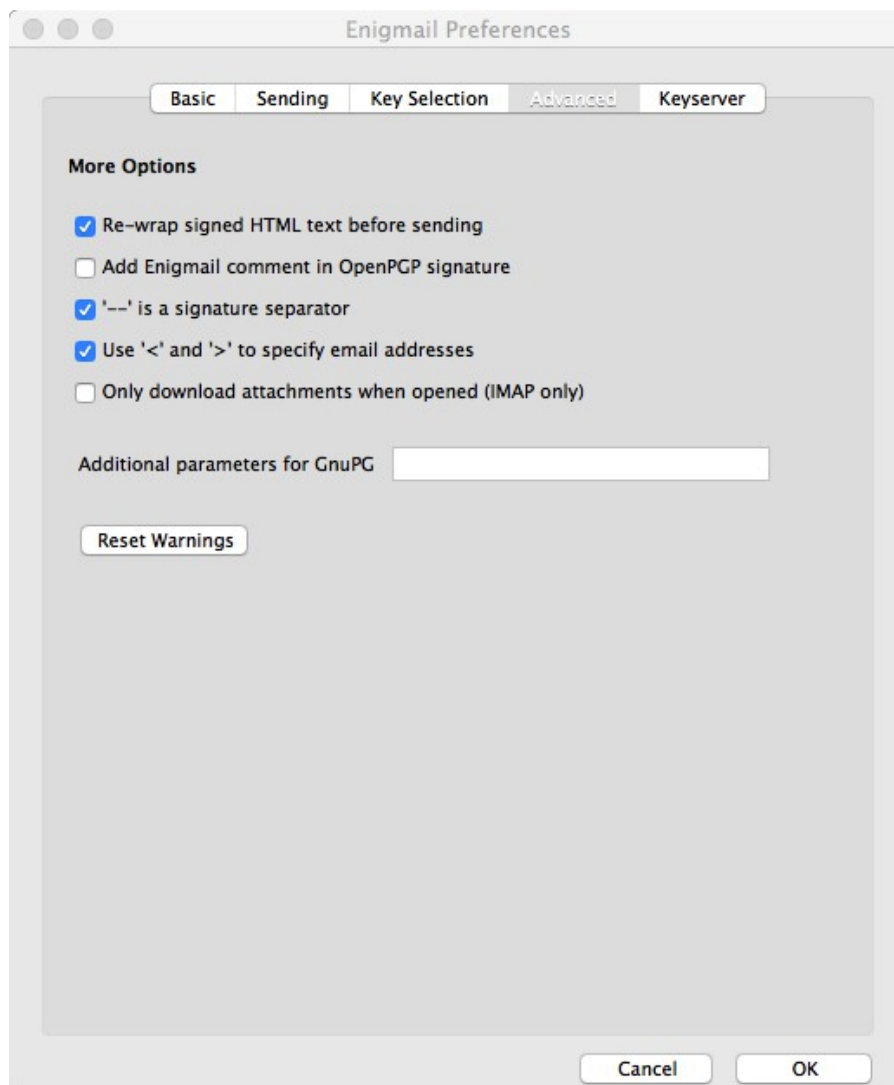
Selection window to let you choose the keys manually (This is checked by default).

- *Always (also) manual*: Will always pop up the Key Selection window to let you choose the keys manually. If one of the above options are selected and have found suitable keys, they will be preselected.

The *Edit Rules...* button opens the Per-Recipient Rules Editor window. See chapter 5.2.2 at page 83 for details about Per-Recipient Rules.

5.1.4. Advanced

This is only accessible if you have enabled the expert settings in the “Basic” tab.



These settings define miscellaneous OpenPGP and Enigmail options.

If you use HTML to compose email messages, messages signed with the inline PGP standard (the default in Enigmail) need to be re-wrapped before they can be sent in order to avoid invalid signatures. We recommend you leave enabled the option *Re-wrap signed HTML text before sending*, unless you have problems caused by re-wrapping.

Add Enigmail comment in OpenPGP signature adds the comment line

Comment: Using GnuPG with Thunderbird -

<http://www.enigmail.net/> to the OpenPGP signature block.

Note that you can add any comment to the OpenPGP signature by calling

GnuPG with the parameter `--comment your_comment` (see below to learn how to specify additional parameters to the GnuPG executable).

When signing, lines starting with a dash (-) are replaced with two dashes separated by a space (- -) according to the OpenPGP standard. This however makes a double-dash line (--) no longer appear as a separator between the message body and a personal signature, usually displayed in grey. By enabling the option *'--' is a signature separator*, Enigmail makes some workaround to correctly handle the signature separator when reading and composing messages.

Use '<' and '>' to specify email addresses. Usually, email addresses are surrounded by angle brackets (< >) to separate the full name part from the email part, e.g. John Random Hacker <jrandomhacker@example.com>.

Deactivating this option removes the brackets from email addresses. This is necessary to ensure compatibility with some provider service, like Hushmail, that does not support brackets in email addresses. Hushmail is a provider for OpenPGP encryption over the web, but keys generated with Hushmail are not fully compatible to OpenPGP.

This option should be normally turned on when encrypting, as Enigmail relies on it to avoid potential confusions and hence security problems, but needs to be turned off for Hushmail keys.

Only download attachments when opened (IMAP only) enables an IMAP feature that makes Thunderbird download only the first 35-40 Kb of a message, downloading attachments only on demand. However, if an encrypted message is larger than this size, it may happen that it is downloaded only in part, its end will be missing, and hence Enigmail will fail to decrypt it.

If you use an IMAP inbox, and notice that Thunderbird sees some of your mails as broken or reports an error when trying to decrypt them, disable this option. Thunderbird will then download the complete message at once. Alternatively, you can click on the broken lock to download the message in full.

The text field *Additional parameters for GnuPG* allows you to have Enigmail call the GnuPG executable with the parameters you prefer.

Finally, the *Reset Warnings* button controls the way Enigmail pops up the interactive dialogs asking you to make a choice. If you ever asked Enigmail to remember your choice for the future (for instance when choosing how Enigmail should sign/encrypt attachments), clicking this button will have Enigmail prompt you the dialog again when needed.

5.1.5. Keyserver



These are the options related to keyservers used to search public keys from.

The text field *Specify your keyserver(s)* allows you to specify a list of OpenPGP keyservers. These keyservers will be proposed to you next time you launch a search for a person's public key on a keyserver.

You may prepend a protocol to the name of a keyserver, e.g.
`hkp://keyserver.example.com` or `ldap://certserver.pgp.com`.

If you want, you may enter a keyserver name in the field *Automatically download keys for signature verification from the following keyserver*. Enigmail will then automatically try to download every public key needed to verify signed messages from the keyserver specified in this field. If you use this option, please specify only one name.

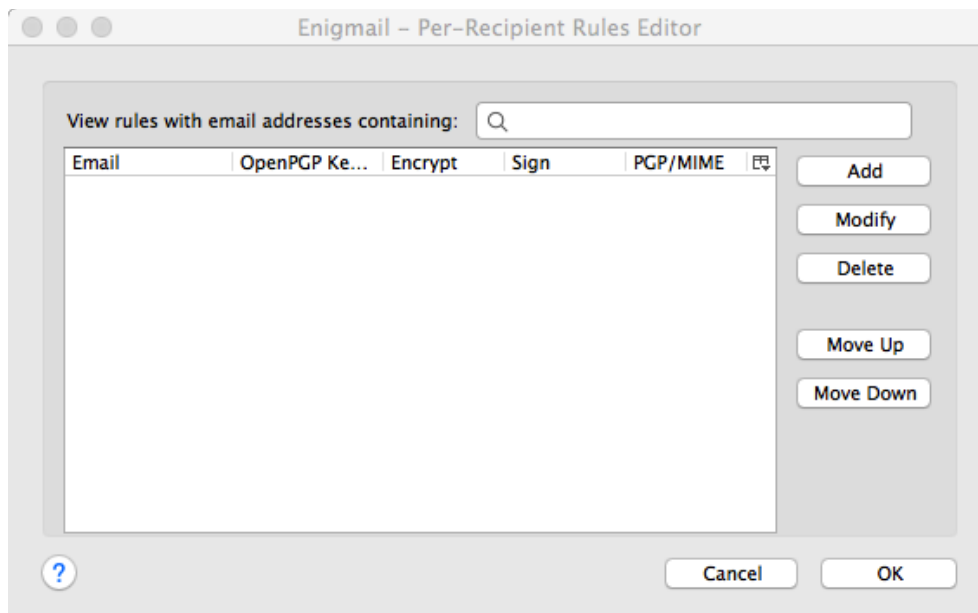
5.2. Per-recipient rules

Enigmail features a powerful per-recipient rule system that, for any recipient, allows you to specify in advance whether to sign, encrypt, or choose between the PGP/MIME standard instead or inline PGP format.

Per-recipient rules also allow you to specify which key to use for an intended recipient of an encrypted message. By default, Enigmail first searches the per-recipient rules and looks up for a rule matching the recipient; if no rule is specified (as it is the case after a fresh install of Enigmail), Enigmail selects the key with a user ID matching the recipient.

5.2.1. Per-Recipient Rules Editor

To edit per-recipient rules, select *Enigmail* → *Edit Per-Recipient Rules*. The picture below shows the Per-Recipient Rules Editor window:

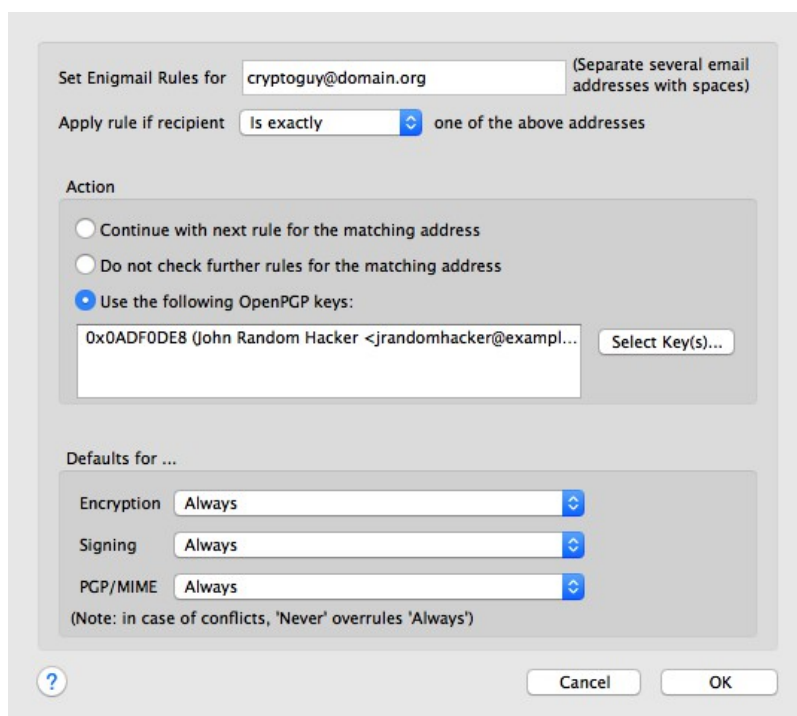


Here, we take the example from chapter 4.3.1, page 59, where we manually encrypted a message to cryptoguy@domain.org. This address is an alias for jrandomhacker@example.com. We have a key for the real address but not for the alias. Thus, Enigmail cannot encrypt automatically.

This can be conveniently solved by Per-Recipient rules. For this purpose, we want to have a rule “When sending a message to [cryptoguy](mailto:cryptoguy@domain.org) always encrypt it with the public key for [jrandomhacker](mailto:jrandomhacker@example.com).”

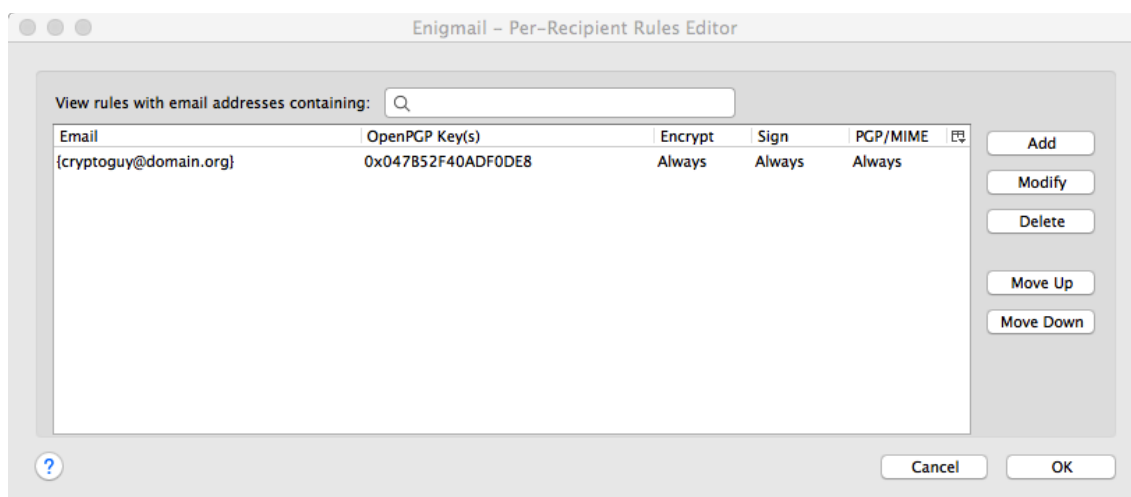
The *Add* button adds a new rule, and *Modify* modifies an existing rule.

We click on “Add” and Enigmail opens a window, where we can enter all parameters for this new rule:



First add the mail address which shall be processed. In this case, we enter cryptoguy@domain.org. Then we select *Use the following OpenPGP keys* and get prompted with the key selection window. We select the key for jrandomhacker@example.com. Then we select *Always* for *Encryption*, *Signing* and *PGP/MIME*.

Click *OK*, and we see our first rule in the list:



If you create more than one rule, they are evaluated in order from top to bottom. You can change the rules order by using the buttons *Move Up* and *Move Down*, while *Delete* will delete a rule.

It's not only possible to set encryption for specific addresses but also to exclude some addresses from encryption or signing. Just select NEVER for encryption or signing fields in the rule.

5.2.2. Recipient Settings

In the *Set OpenPGP Rules for* field you must enter the recipient email address you're writing the rule for. Recipients are the addresses specified in the fields To:, Cc:, and Bcc: of the email message, without distinction. If you want to have a rule for multiple email addresses, enter them all in the field, separated by spaces. Then choose the pattern matching criteria from the drop-down menu (*Is exactly, Contains, Starts with, Ends with*).

In the *Action* zone you specify the rule behaviour. If there is a match with the specified recipient email address(es):

- *Continue with next rule for the matching address*
Allows you to define a rule without having to specify a key ID in the *Use the following OpenPGP keys:* field. This way, the email address is used to check for a key when sending the message.
Further rules for the same address will be processed.
- *Do not check further rules for the matching address*
Stops processing any other rules for the matching address if this rule is matched.
Rule processing restarts with the next recipient.
- *Use the following OpenPGP keys*
Allows you to specify which recipient keys will be used for encryption. Use the *Select Key(s)...* button to choose the keys. This is the recommended method as it is fail safe.
Further rules will be processed.

In the *Defaults for...* zone you decide whether to activate signing, encryption and PGP/MIME if the rule is matched. Each function can be independently set to three options:

- *Never* specifies that the function will be off.
- *Yes, if selected in Message Composition* allows you to set the option at the time of message composition.
- *Always* specifies that the function will be on.

When sending a message to multiple recipients, in case of conflicts between rules, *Never* overrules *Always*. For instance, if you created two rules for the following two recipients:

alice@example.com
Signing: *Always*
Encryption: *Always*
PGP/MIME: *Yes, if selected in Message Composition*

bob@domain.org
Signing: *Always*
Encryption: *Never*
PGP/MIME: *Never*

and you try to send a signed and encrypted message to `alice@example.com` and `bob@domain.org`, the message will be signed only. Also, should you have turned on PGP/MIME when composing the message, this setting would have been ignored and the message won't be encrypted with PGP/MIME.

5.2.3. Notes

You can jump directly to the Recipient Settings, and create a rule for a particular email address, by right-clicking on that address from the Message window or the Address Book and selecting *Create Enigmail-Rule from Address...* from the pop-up menu.

The rules are processed sequentially in the order displayed in the rules editor. If a rule contains an OpenPGP key, the rule is applied, but the address that triggered the match will not be rechecked in any following rules.

In order to minimize the number of rules you have to create, you should set carefully your OpenPGP global and account settings.

How Enigmail chooses the recipient keys for an encrypted message is controlled by the settings in *Enigmail → Preferences → Sending* and *Enigmail → Preferences → Key Selection*. By default, Enigmail first checks per-recipient rules, then the user IDs in your public keyring.

You can enable the option *Enigmail → Preferences → Sending → Confirm before Sending: Always* in order to check the status for signing, encryption, and PGP/MIME before a message is sent.

Per-recipient rules are stored in a file called `pgprules.xml` located in your profile directory. If you backup/restore your profile manually, you should make sure to include this file. The format of this file is documented in section 9.3 at page 116.

6. TROUBLESHOOTING

This chapter contains several tips to troubleshoot any problem you may encounter when installing or using Enigmail.

Debugging

Enigmail will always keep a debug log. You can view this debug log via the menu command *Enigmail* → *Debugging Options* → *View Log*. This opens a window, where you can save the contents in a file on your computer in case you need to send it to the developers.

You can check the commands sent to GnuPG in the Enigmail Console, which is available via the menu command *Enigmail* → *Debugging Options* → *View Console*.

Enigmail fails to install on Firefox

Enigmail is an extension for Thunderbird. It is not supposed to, and hence cannot, be installed in Firefox.

If you use Firefox to download Enigmail, you need to right-click on the download link, select *Save as...*, and save the XPI file on your computer.

Then open Thunderbird, go to *Tools* → *Add-ons* → *Extensions*, click the “gear-wheel” button select *Install Add-On from File* and choose the Enigmail- XPI file. Restart Thunderbird afterwards.

I can't tell whether Enigmail works or not

If the installation was successful, restart Thunderbird. The menubar now should have an *Enigmail* entry. Selecting *Enigmail* → *About Enigmail* will display the Enigmail version number and GnuPG executable details.

I installed a new extension and Enigmail stopped working

Some extension cause conflicts with Enigmail, preventing it to successfully sign/encrypt outgoing mail or verify/decrypt incoming mail.

Enigmail is unable to access the keyserver.

Keyservers use the Horowitz Keyserver Protocol (HKP) to exchange keys through TCP port 11371. If you are behind a firewall, you must ensure that this port is open for outgoing connections. Alternatively, many keyservers allow access to clients also on HTTP (TCP port 80), which is normally open.

If you are using HTTP proxy behind a firewall, you must add the following line to your `gnupg.conf` file:

```
keyserver-options http-proxy=your_proxy_host_name
```

Enigmail sees some emails as broken.

This problem occurs often when using an IMAP mailserver and is due to Thunderbird not downloading the message as a whole. To fix this problem, go to *Enigmail* → *Preferences* → *Advanced* and disable the option *Only download attachments when opened (IMAP only)*. See also the explanation of this option in Section 5.1.4 at page 78.

Thunderbird shows an alert about an “Unresponsive script”

Sometimes, Enigmail (or technically better: the underlying GnuPG) take a long time for the cryptographic operations and Thunderbird complains about an “Unresponsive Script”. If this happens while sending an encrypted mail, you should NEVER click on “Cancel”, as this would send the mail unencrypted! In this case always select “Continue”. If several attempts do not help, then quit and restart Thunderbird.

I cannot read encrypted messages sent to me! I get an error “Secret key needed to decrypt message”.

Unless you accidentally deleted your key pair (for which there is no remedy, unless you have a backup – see below), the message you received was not encrypted with your public key. The sender most likely encrypted it with his public key only instead of yours.

Make sure the sender has your public key, and tell him to encrypt the message with it.

I have lost my passphrase / my key pair / my private key

A note: Your private key is bundled with your public key in your key pair, hence losing your private key and losing your key pair means exactly the same.

There is no way to recover your passphrase: your only hope is to try to remember what it was. If you don't succeed, you lose the use of your private key, and hence your whole key pair is now useless.

There is no way to recover your private key, either. It cannot be obtained from your public key or from any message that was signed/encrypted by that private

key. You can only recover it if you made a backup in the past.

Hence, losing the passphrase or the key is definitive. If you generated a revocation certificate (and you should have), use it to revoke the key pair. You must also generate a new key pair, send the new public key to your contacts and warn them not to use the old public key any more.

Messages that were sent to you encrypted with the old key cannot be decrypted any more. Messages that were signed by you with the old key can still be verified by the recipients by using the old (revoked) key.

To avoid this disaster, it is recommended that you backup in advance your key pair: from Key Management, select *File* → *Export Keys to File*, make sure you included the secret key, then store the file in a safe place. Make sure you chose a passphrase you can remember, too.

After I reinstalled Enigmail, all keys have disappeared from the Key Management window

The keys are still there, but are displayed only the keys that match the search criteria entered in the *Search for* field. If you want to see all keys, tick the checkbox *Display all keys by default*.

I get an error whenever I try to post to a newsgroup



You are trying to post an encrypted message to a newsgroup. This doesn't make any sense as a newsgroup, like a mailing list, is a public space and not an entity that could own a key pair. (Just ask yourself: who is supposed to own the private key? What would be the trust associated with this entity?) Send your message unencrypted. If you just want to obfuscate information, like spoilers, you can as well use ROT13.

I have some other problem I can't solve

In this case, if you know how to use GnuPG, you can try to achieve the same result through the GnuPG command line. For instance, if you cannot remove public key 0xABCDEF01 from Key Management, open a shell prompt and issue the following command: `gpg --delete-keys 0xABCDEF01`

If the above doesn't work or you don't feel yourself enough experienced to use GnuPG, ask the friendly Enigmail/GnuPG community for support. References

for obtaining support are listed in chapter 12 at page 121.

7. FAQ

This chapter contains the Frequently Asked Question about Enigmail and around.

Can Enigmail be used for webmail? When will this feature be added?

Enigmail is developed for Thunderbird. There is no intention from our team to extend Enigmail to support web based mail, or web applications in general.

The mailvelope project (<https://www.mailvelope.com/>) is an extension for Mozilla Firefox or Google Chrome allowing OpenPGP secured messages in webmail.

Also the Whiteout Browser App (<https://whiteout.io/#product>) can be used to process OpenPGP secured mails using your browser. This is not really a webmail, but a mail client emulation in a browser window.

How do I uninstall Enigmail?

Go to *Tools* → *Add-ons* → *Extensions*, click on the *Remove* button, then restart Thunderbird.

How can I get the HTML view back?

Go to *View* → *Message body as* → *Original HTML*

Which key type/size should I choose for my key pair? Which is best?

There is no such thing as “the best key.” All choices have consequences and trade-offs. You might feel that a 4096-bit RSA key is safer, but the person you're sending email to might be trying to read it on an old PDA which takes over a minute to decrypt each message. You might decide to use SHA-1 digest because it's widely supported in OpenPGP implementations, but SHA-1 has some mathematical flaw and does not offer long-term security. Finding precisely the optimal set of consequences and trade-offs is a very subtle thing, and the perfect set for you will probably not be the same for anyone else.

The IETF OpenPGP Working Group has spent over a decade looking at which choices offer an excellent balance of speed, safety, and compatibility for the vast majority of users. Their opinions have evolved over time to take into account the technology and threats of the day. The people of the GnuPG project

are active participants in the Working Group, and as such GnuPG implements the Working Group's recommendations.

Therefore, the best advice we can give is to stick to Enigmail's defaults. They are not perfect, because no two people have the same definition of perfection. However, the defaults are excellent for the overwhelming majority of users.

How can I test if I'm using Enigmail correctly?

First, you can try to send yourself some message signed/encrypted, and check if you are able to verify/decrypt them correctly. Then you can send messages to Adele, “the Friendly OpenPGP email robot”, at `adele-en@gnupg.de`. Adele is an automated program that is able to receive and understand OpenPGP messages and reply accordingly. Remember to send her your public key so she can encrypt test messages to you. Some examples of communications with Adele are shown in Chapter 4.5 at page 64.

How do I encrypt automatically my email messages?

If you use the default configuration, Enigmail will automatically encrypt all messages whenever possible – that is, if you have the public keys for all recipients.

Additionally, you may set single (or all) identities to ALWAYS encrypt (and opt-out while sending) if you don't have a recipient's key and it is acceptable for you to send THAT message unencrypted.

Is it possible to permanently decrypt email messages?

Yes, this is possible in Enigmail, please refer to chapter 4.6 at page 68.

What shall I do if I'm getting a “bad signature” from a message?

This is nothing spectacular: It happens now and then, that a signed message was changed during transport, producing a “bad signature”. This is mostly caused by shortcomings in one of the participated software implementations. This does NOT mean that the sending person is not trustworthy. Please refer to chapter 4.2.2, and the paragraph at page 57.

Why does Enigmail tell me “Untrusted good signature”, I already have the key of the sender?

This means that the signature cryptographically verifies, but the sender's key is not fully valid in your public keyring. This is the default for freshly imported keys. Please see section 3.4.5 at page 41, on how to get full validity.

Is it possible to use S/MIME and OpenPGP encryption concurrently?

No, you cannot mix S/MIME and OpenPGP in the same message as the two standards and their implementation in Mozilla, interfere with each other. If you want to use S/MIME you should not enable the Enigmail option *Encrypt messages by default* in your account settings (nor the corresponding one from S/MIME).

How do I specify the hash algorithm?

Enigmail relies by default on GnuPG for selecting the hash (digest) algorithm. From GnuPG, the hash algorithm can be specified in the file `gpg.conf` using the parameter `digest-algo hash_algorithm`.

If you want to select the hash algorithm from within Enigmail, you can do so by modifying the preference `extensions.enigmail.mimeHashAlgorithm` and assigning to it one of the following values:

- 0 – automatic selection, let GnuPG choose (default, recommended)
- 1 – SHA1
- 2 – RIPEMD160
- 3 – SHA256
- 4 – SHA384
- 5 – SHA512

To know more about modifying the preference values manually, also read Section 9.2 at page 105.

I have lost my key pair and can't import the revocation certificate

You must first import your public key, either from a key server or from a mail correspondent. After this you can import the revocation certificate. Please refer to section 3.7 at page 46.

How do I enable the debug log in Enigmail?

Select *Enigmail* → *Debugging Options* → *View Log*. You can view the log and save it to a file. See also Section 6 at page 85.

How do I report a bug?

Visit <https://www.enigmail.net/support/bugs.php>. Please check first the list of already known bugs so that a bug doesn't get submitted twice. If you spotted a new bug, you can file a bug report. If you're in doubt, please first ask on the mailing list or in the user forum. See section 12 at page 121 how to reach them.

How many people use Enigmail?

You can view the Enigmail download and usage statistics at <https://addons.mozilla.org/en-US/thunderbird/addon/enigmail/statistics/>.

It would be great if Enigmail could do this-and-this! Could you please implement it?

You can submit feature requests in the Enigmail Forum, Feature Requests thread. To know how to reach the Forum, see Chapter 12 at page 121.

But please first consider that Enigmail follows the OpenPGP standard. It is not its purpose to innovate or invent new protocols. If the feature you propose is not included in or not compliant to the standard, the feature is not going to be included in Enigmail, no matter how many users ask for it.

The Enigmail source code is freely available, though. If you really need such a feature, you can download the code and modify it to suit your needs. Please consider first that breaking standards is generally not a wise idea, and will result in incompatible products.

How can I encrypt the Subject?

It is not possible to encrypt or sign the Subject of a mail message, nor any other mail header. See Section 4.7 at page 71 to read why.

Why can't I select some keys for encryption in the Key Selection window?

Keys that are revoked or expired cannot be used to encrypt. Download a valid public key from a keyserver, or contact your recipient and have him mail you his new, valid public key. Do not forget to ensure the integrity of this key by a secure channel!

8. NOTES, TIPS & TRICKS

8.1. How to choose a good passphrase






The passphrase is the last line of defence to your private key, should your key pair fall in enemy hands. This might happen more easily than you think, by means of someone stealing your laptop, a malware uploading your private documents from your infected computer to a rogue server, or simply by your momentary thoughtlessness when you distribute your whole key pair instead of your public key.







With your secret key and your passphrase, anyone can impersonate you by signing messages on your behalf, and decrypt messages that were intended for your eyes only.

Luckily, the passphrase provides a quite good protection, since it encrypts the private key with a strong cipher. It is important that you choose a strong passphrase that could not be easily cracked by password guessing or brute-force programs. In this section we illustrate some criterion to do so.






GnuPG/Enigmail also allow you to not set a passphrase on your key pair. **This is absolutely not recommended**, and should be done only in exceptional circumstances, for instance when non-interactive processing is needed.

Do not use the following as your passphrase:

-  Your name, address, age, date or place of birth, car license plate, the name of your spouse, children, parents, pets, or any other information related to you;
-  Words in any language/dialect, past or present, real or imaginary, e.g. French, Cockney, Latin, Elven, and Klingon;
-  Names of real or fictitious people or places;
-  Names of movies, songs, music bands, groups, and such;
-  Obvious sequences of letters and/or numbers e.g. abc123, qwerty, YYYYYYYY

-  Numerical constants e.g. 2.71828 18284 59 (it's the mathematical constant e)
-  Any of the above written in all uppercase, all lowercase, or with alternated case e.g. CaLiFoRnIa
-  Any of the above prefixed or suffixed by a single character e.g. +California, California3
-  Any of the above with obvious replacements e.g. C4l1f0rn14
-  Anything that's less than 8 characters long (Enigmail will not even let you choose a passphrase that's shorter than that)
-  A password that you already use (e.g. on web sites or for your email account)

Instead, use these criteria to create a passphrase:

-  Use always a mix of at least 3 of the following characters in your passphrase: uppercase letters, lowercase letters, numbers, symbols like # * ! ? + - (& /
-  Insert two characters or more inside a word or name e.g. Ch7op8in, Debus!Z*sy
-  Join two words or names by two or more characters e.g. Bach#+Strauss
-  Nest one word or name inside another e.g. BeLudwigethoven
-  Condensate a proverb, a quote, a verse from a poem, a phrase from a movie, or any sentence you could have fixed in your mind:

Iw20yat/SPttbtp/thbgiaaos/btagtras.

This might seem impossible to remember but is in fact quite easy, once you think about the lyrics of *Sgt. Pepper's Lonely Hearts Club Band* by Lennon/McCartney:

“It was twenty years ago today
Sgt. Pepper taught the band to play
They've been going in and out of style
But they're guaranteed to raise a smile.”

Each letter of the passphrase is the first letter of each word. In the first line, the number is written in figures instead of being spelt out. In the second line, the name of the protagonist of the song is in uppercase letters. Each verse is separated by a slash, and a final dot is added. You can make up the rules as you prefer.

Another possible passphrase would be

HwmyrsmtBeyuclhm?

This one comes from Bob Dylan's *Blowin' In The Wind*, and is derived from the first and last letter of each word, considering only the first four of each verse:

“How many roads must a man walk down
Before you call him a man?”

These are the strongest passphrases, as they look like random sequences of letters.

You can use an existing quote, and make up the rules to transform it in a passphrase; should you ever forget the quote, a quick look on a book will solve the problem. You may also invent your own quote, although in this case forgetting it would be fatal.

8.2. Protection of the local computer

You should be aware of the truth that your encrypted mails are as safe as allowed by the computer you use Enigmail on. This point can never be stressed enough.

If your computer is infected with a key logger and a malware that grants an intruder remote access on your files, all the cryptographic robustness of OpenPGP and the strongest passphrase won't protect your messages from being snooped or falsified. In a similar way, if you leave your computer unattended with your passphrase cached on, prepare yourself for nasty surprises. In fact, even using cryptography, your communications cannot be secure if your computer isn't. Even worse, cryptography could lure you into a false sense of security, making you more prone to share sensitive information via email.

The ciphers OpenPGP uses are the strongest known, and OpenPGP encryption is virtually unbreakable if done in the right way. However, there are a lot of other things that can go wrong.

The well-established fact that OpenPGP is the strongest link in the chain of security simply means that an attacker wanting to read your encrypted messages won't try to brute-force the encryption (which would take millions of years), but will focus on other weaknesses instead.

He might break into your computer and steal your secret key. Then, infect your computer with a spyware to record your passphrase, or directly record your

secret messages as you're typing them. For the purpose of recording, he might as well use a hardware key logger installed between keyboard and computer. Or simply a hidden camera pointed towards your screen. Or even a TEMPEST device, hoping that you still use a CRT screen.

Once he gets his hands on the contents of your computer, either physically or from a remote location over the network, he may search for any plaintext remnants in nonvolatile storage devices or RAM.

From where did you get your copy of GnuPG and Enigmail? You should only trust software downloaded from the official web sites. Copies obtained from other sources might have been tampered with, and as such contain viruses, backdoors or trojans.

Finally, an attacker might persuade, force, or delude you (e.g. by impersonation) to surrender your passphrase, your secret key, or your messages. And all these attacks can be carried over your correspondents, too. The possibilities are endless.

8.2.1. Basic protection

You must follow these golden rules in order to keep your computer reasonably safe:

- Don't install, run, or open software of dubious origin (e.g. warez found on peer-to-peer networks, or programs hosted on untrusted web sites). This includes suspicious email attachments and macros on word processing programs.
- Use an antivirus/antimalware software, updated daily. Make frequent scans of your computer and external hard drives.
- Install OS vendor patches. Keep all your software up-to-date, and keep yourself informed of the latest vulnerabilities.
- Use a screen lock when you are not physically in front of your computer and lock it immediately when strangers are around around.
- Use strong passwords. Don't write them down in easy-to-find places.
- If you use a Wi-Fi connection, enable WPA2 on your access point.

8.2.2. Increased protection

If your communications involve critically sensitive information, you should not leave your computer physically accessible at all – even when turned off. If stolen, the thief would have access to all your files, including your secret key. The private key will still be protected by the passphrase but, by performing analysis and forensics on the filesystem, the thief will have access to a lot of plaintext data (temporary files, memory swap files, and such) that could include information you thought was encrypted. Windows leaves a lot of data around, and other OS's aren't much better with respect to this.

You might consider using whole-disk encryption at this point. Section 8.3.2 mentions some disk encryption software for additional protection of your key pair; most of this software can also be used to encrypt the whole OS.

It is also worth noting that a technically skilled intruder having physical access to a turned-off computer could infect it, leaving no traces, by replacing the bootloader with an infected one (*evil maid attack*).

8.3. Keeping your key pair in a safer place

To increase the security of your secret key you may decide to store your key pair in a different location than the default directory chosen by GnuPG, which for Windows is `C:\Documents and Settings\your_username\Application Data\GnuPG` in the local computer.

The easier solution is to keep the GnuPG files in an external USB drive, or an encrypted volume in the local hard disk. A more complex solution involves the use of a smart card.

8.3.1. External USB drive

First, mount the external drive and move there all GnuPG files (your keyring, the random seed file, and configuration files) that were contained in the default directory. Thunderbird must not be running while you move the files.

Then, you must tell GnuPG where the new location is, by passing the additional parameter `--homedir new_location` to the GnuPG executable. This is done directly inside the OpenPGP Preferences, via the menu command *Enigmail* → *Preferences* → *Advanced*, in the field *Additional parameters for GnuPG*. How to do this was explained in Section 5.1.4.

Once you have done this, you can use Enigmail in the usual way. Remember to have your external drive mounted before running Enigmail or GnuPG.

8.3.2. Encrypted volume

You may store the GnuPG files on, instead of an external drive, an encrypted virtual volume in the local hard disk (or even, for extra protection, an encrypted virtual volume on an external drive itself).

There are some on-the-fly encryption programs available, however, a lot has changed during the last months, so that we cannot give a long standing recommendation.

The encrypted virtual volume will behave just like an external drive. Once you have installed the encryption program of your choice, created the encrypted virtual volume, and mounted it, do the necessary setup by following the same steps explained in Section 8.3.1.

You also may want to use whole-disc-encryption, offered by most modern operating systems, like Bitlocker on Window or FileVault on Mac OS X. See https://en.wikipedia.org/wiki/Comparison_of_disk_encryption_software for an overview.

8.3.3. OpenPGP card

Enigmail supports the OpenPGP card, a smart card compatible with ISO standards 7816-4 and 7816-8; see <http://g10code.com/p-card.html>. The figures below show front and back of an OpenPGP card:



OpenPGP cards are distributed by Kernel Concepts, <http://www.kernelconcepts.de>. It is also possible to obtain a OpenPGP card by becoming a Fellow of the Free Software Foundation Europe; please read http://wiki.fsfe.org/FellowshipSmartCard?action=show&redirect=Crypto_Card for more information.

OpenPGP v2.0 cards feature three independent RSA keys, for signing, encryption, and authentication, of up to 4096 bits each. Some older gnupg versions might support shorter key lengths. The card is used to store the actual secret key. A secret key stub remains within the secret keyring so that gnupg knows about the key on the card and can prompt you to insert the card if it is needed and perform key operations.

The purpose of using a smart card is that the secrets it contains cannot be copied from the card. Therefore, as long as the card stays physically in your possession, you know that your secret key is safe.

There are two methods to initialize a card. Following the first method, the key is generated on-card, i.e. the card calculates the key using its built-in random generator; in this way the secret key never leaves the card. Otherwise, a standard RSA key can be generated in a safe environment, e.g. a clean Linux workstation not connected to any network and booted from a CD-ROM. The secret key is then moved to the card.

This key can later be stored to another OpenPGP card if the original card gets lost or broken. However, the new card will have new signing and authentication keys.

For advanced users: the method that guarantees the maximum availability of the keys, at the expense of secrecy, is to create a compatible key. This is done by creating via the GnuPG command line (use the `--expert` flag) keys with distinct functionalities (1024-4096 bit, RSA only). These keys allow you to backup a fully functional key, for which no card is needed, which is helpful in case you revoke your card key but still want your mail archive to be readable.

You can also create a full clone of that key on another card if availability is vital. As long as you protect your original backup key appropriately, this allows you to leave your card in a system managed by someone else without the fear that your secret key could be stolen unnoticed. In fact, since the secret key cannot be copied from the card, the only way to pick up the key is to physically steal the card – which you'll notice.

From the menu item *Enigmail* → *Manage SmartCard...* you can access all smart card operations:

- manage the user data (name, sex, language, login ID, URL of the public key) stored on the OpenPGP card;
- generate a new key on-card;
- change your PIN (123456 by default) and Admin-PIN (12345678 by default).



Generating a new key on-card will overwrite the pre-existing key.

Remember to change your PIN and Admin-PIN before generating a new key. The PIN is not restricted to digits only but can be any combination of characters; choose strong PINs since they are the only protection to the secret key if the card is lost or stolen. However, bear in mind that non-numeric PINs cannot be entered on PIN-pad readers.

It is strongly recommended that you test to recover your secret keys (both your card and the key on your local computer) from a backup key and a blank card. If you have only one card available, you may still simulate the recover (v2.0

cards only) by resetting the card via the command

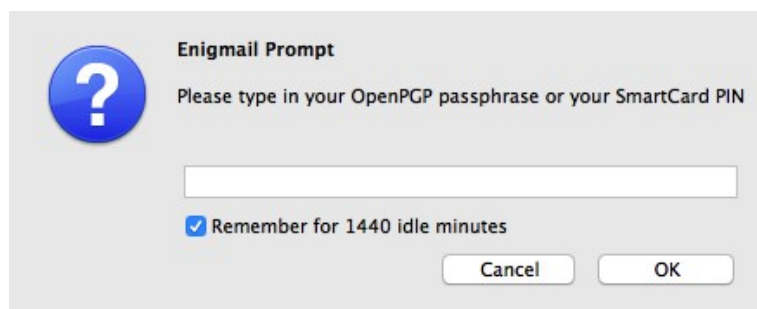
```
gpg-connect-agent < resetfile
```

where *resetfile* is an ASCII text file composed of the following lines:

```
/hex
scd serialno
scd apdu 00 20 00 81 08 40 40 40 40 40 40 40 40
scd apdu 00 20 00 81 08 40 40 40 40 40 40 40 40
scd apdu 00 20 00 81 08 40 40 40 40 40 40 40 40
scd apdu 00 20 00 81 08 40 40 40 40 40 40 40 40
scd apdu 00 20 00 83 08 40 40 40 40 40 40 40 40
scd apdu 00 20 00 83 08 40 40 40 40 40 40 40 40
scd apdu 00 20 00 83 08 40 40 40 40 40 40 40 40
scd apdu 00 20 00 83 08 40 40 40 40 40 40 40 40
scd apdu 00 e6 00 00
scd apdu 00 44 00 00
/echo card has been reset to factory defaults
```

8.4. Passphrase Handling

There are basically two ways how your passphrase can be handled. The first is the “old” way, where Enigmail is keeping control and caches the passphrase itself. Enigmail automatically selects this method if it detects a GnuPG version 1.4.x. This method is sufficient if you only have one private key and therefore only one passphrase. However, it's getting cumbersome if you own several private keys. You never know for which key the passphrase you are asked for. Here's an example of the Enigmail internal passphrase prompt:



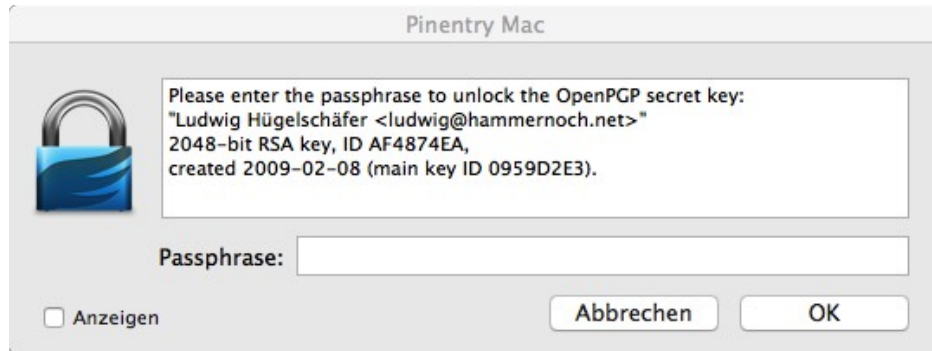
Nowadays, there's a much better alternative: GPG-agent.

Enigmail will (and must) use GPG-agent, if it detects a GnuPG version greater than 2.0. GPG-agent is an independent passphrase handling software and part of the GnuPG 2.x package. It runs outside of Thunderbird/Enigmail and offers a lot of advantages:

- Caching of passphrases for different keys

- Protection of memory being swapped to disc
- Common system for all applications requesting a passphrase

GPG-agent needs a software named pinentry providing the graphical dialog. There are different flavours pinentry; their appearance differs slightly with the operating system and on Linux with your window manager. The next image shows pinentry-mac (Mac OS X) asking for my passphrase:



The GnuPG versions installed by the Setup wizard on Windows and Mac OS X are 2.0.x. Both use the GPG-agent.

On Linux, the installed versions by default differ, but virtually every distribution offers a GnuPG 2 package.

Note: Enigmail 1.8 will be the last version that supports GnuPG 1.4.x. The next major release (1.9) will require GnuPG 2.x, so people are encouraged to upgrade to or additionally install GnuPG 2.x within the next months.

Read more about this here:

https://www.enigmail.net/support/transition_to_gnupg2.php

8.5. Key verification procedure

When you verify keys from other persons, you should check that the key really belongs to the person which is named in the user Id. Therefore you should compare all information you find electronically on the key with those you get from the person directly.

The following procedure is regarded as good practice:

1. Meet the person face-to-face
2. Receive their fingerprint from them
3. Receive their email address(es) from them
4. See at least one form of government-issued identification
5. Get the key from a keyserver or directly from the person
6. Verify that the email address(es) on their user ID(s) match the email address(es) they gave you
7. Verify that the fingerprint on their key matches the fingerprint they gave you

Exchanging fingerprints can be made in form of paper or speech. You can exchange paper sheets or sit together, one reading the fingerprint, the other one comparing it on the display of his computer.

If you exchange papers, you should note on it that you've seen the government-issued identification, especially if you collect more than one sheets at one occasion.

There are organised events for exchanging keys: people come together for a so called key signing party, usually on congresses about computer security or open source software. Read more about them here:

https://en.wikipedia.org/wiki/Key_signing_party.

9. ADVANCED OPERATIONS

9.1. Manually installing GnuPG

This paragraph is only needed, if you did not let the setup-wizard install gnuPG for you on your computer or you must do it for other reasons.

9.1.1. Installing GnuPG on Microsoft Windows

Most people prefer to use Gpg4win (<http://www.gpg4win.org>), a sibling project to GnuPG.

There are several types of downloads:

- The normal (full) installer, providing a GUI for key management, encryption, and decryption. The Gpg4win installer also contains additional optional components such as a plug-in for MS Outlook, and a mail client (Claws Mail) already integrated with the GnuPG plug-in.
- The “light” installer, without Keymanager and documentation.
- The “vanilla” installer, containing only the GnuPG components.

You don't need to modify anything in the configuration of GnuPG; the default settings will work fine.

Learning how to use the command-line GnuPG is not required; once installed, all operations will be achieved via Enigmail.

9.1.2. Installing GnuPG on Macintosh OS X

You have three basic ways to install GnuPG on OS X. Most users will choose the first option.

The first and most popular option is to use the MacGPG package by gpgtools.org. They provide pre-built binaries of GnuPG 2.0.22 and later for users running OS X 10.6 (Snow Leopard) or later. Download the GPG-Suite from <https://gpgtools.org/gpgsuite.html> and install it. If you don't want to use the other components (Certificate Manager, Apple-Mail plugin and Services application), you can deselect them during the installation process. Only the MacGPG component is required for Enigmail.

The next two options are only for those familiar with building software from source. They require the Xcode development software from Apple.

One of these sources is the Homebrew project (<http://brew.sh/index.html>) which has grown popular during the last years. Open up Terminal.app and type

```
sudo brew install gnupg2
sudo brew install pinentry-mac
```

The other source is the MacPorts Project (<http://www.macports.org>), that keeps a current version of GnuPG in their source tree. If you're using MacPorts, open up Terminal.app and type

```
sudo port install gnupg2
sudo port install pinentry-mac
```

9.1.3. Installing GnuPG on Linux / UNIX

The best thing is to get a pre-compiled version of GnuPG for your UNIX system using the package manager of your platform. Compiling GnuPG from source is not recommended for beginners.

Most Linux distributions today include GnuPG by default. To find out if this is the case, open a command prompt and type

```
gpg --version
```

If it tells you that you've got GnuPG 1.4.9 or some later version, then you don't need to do anything.

You are encouraged to additionally install gpg2 or gnupg2 package using your packet manager software, e.g apt, yum, yast. On many distributions the package is called "gnupg2" or "gpg2".

The various flavours of BSD UNIX have a common way of installing software. Please see the instructions for MacPorts above.

9.2. Manually editing the preferences

Manual editing of preferences are intended for advanced users only!

Enigmail preferences are stored together with Mozilla global preferences inside the `user.js` and `prefs.js` files in your profile directory. Mozilla first reads the preferences in `user.js`, if this file exists, and copies them into `prefs.js`. You can create the file `user.js` in order to keep your custom preferences that you do not wish to be ever changed; this file, unlike `prefs.js`, won't be modified by Mozilla.

The preferences are accessible in Thunderbird under *Tools* → *Preferences* → *Advanced* → *General* → *Config Editor*

From there, you can change the preferences values, which will then be written in `prefs.js` under the following format:

```
user_pref("preference_name", value);
```

You can review the default preferences settings in the `enigmail.js` file.

The following three paragraphs provide a reference to the various Enigmail preferences. First is printed the preference name and its default value; then follows its meaning.

The first paragraph will list all general preferences which are settable via the normal user interface (UI). In the second paragraph you can review the per-identity/account settings default values.

The third paragraph will list all preferences not

All will be listed in alphabetical order.

9.2.1. General preferences (UI settable)

extensions.enigmail.advancedUser ***false***

Shows the advanced settings.
Enigmail → *Preferences* → *Display Expert Settings / Hide Expert Settings*

extensions.enigmail.agentPath ***""***

The path to (and including) the GnuPG executable. If it is already in the PATH, this setting can be left blank. **Note:** This is not the path to GPG-agent!

Enigmail → Preferences → Basic; GnuPG executable path

extensions.enigmail.agentAdditionalParam **""**

Additional parameters passed to the GnuPG executable. Default value is empty.

Enigmail → Preferences → Advanced → Additional parameters for GnuPG

extensions.enigmail.acceptedKeys **1**

Selection which keys to accept for sending. Default is “1”.

Enigmail → Preferences → Sending → Manual encryption settings; To send encrypted, accept

- 0: accept valid/authenticated keys
- 1: accept all keys (except disabled, expired)

extensions.enigmail.assignKeysByRules **true**

Use Per-Recipient Rules to select keys.

Enigmail → Preferences → Key Selection → By Per-Recipient Rules

extensions.enigmail.assignKeysByEmailAddr **true**

Use Email address to select keys

Enigmail → Preferences → Key Selection → By Email Addresses according to the keymanager

extensions.enigmail.assignKeysManuallyIfMissing **true**

Select keys manually if missing.

Enigmail → Preferences → Key Selection → Manually if keys are missing

extensions.enigmail.assignKeysManuallyAlways **false**

Always select keys manually. May be used in combination with other selections.

Enigmail → Preferences → Key Selection → Always (also) Manually

`extensions.enigmail.autoDecrypt` **`true`**

Automatically decrypt/verify received messages.

Enigmail → Automatically Decrypt/Verify Messages

`extensions.enigmail.autoKeyRetrieve` **`""`**

Keyserver to automatically retrieve keys from if missing (e.g. during signature verification)

Enigmail → Preferences → Keyserver → Automatically download keys for signature verification from the following keyserver

`extensions.enigmail.autoSendEncrypted` **`1`**

Selection for automatically send encrypted if all keys are valid

- 0: Never
- 1: If possible (all keys are found and accepted, Default)

Enigmail → Preferences → Manual encryption settings → Automatically send encrypted.

`extensions.enigmail.confirmBeforeSending` **`0`**

Pops up a confirmation dialog before sending a message.

Enigmail → Preferences → Sending → Manual encryption settings → Confirm before sending

- **0: never (Default)**
- 1: always
- 2: if sent encrypted
- 3: if sent unencrypted
- 4: if Per-Recipient rules changed the default encryption setting

`extensions.enigmail.doubleDashSeparator` **`true`**

Handles the double dash (--) as a signature separator.

Enigmail → Preferences → Advanced → '--' is a signature separator

extensions.enigmail.encryptionModel **0**

Controls how Enigmail will select the settings for messages to be encrypted.
Available values are:

0 – convenient encryption settings (default)

1 – manual encryption settings

Enigmail → Preferences → Sending → Convenient encryption settings/Manual encryption settings

extensions.enigmail.hushMailSupport **false**

Enables support for Hushmail.

Enigmail → Preferences → Advanced → Use '<' and '>' to specify email addresses

extensions.enigmail.keepSettingsForReply **true**

Enables encryption of replies to encrypted messages.

Enigmail → Preferences → Sending → Encrypt replies to encrypted message

extensions.enigmail.keyManShowAllKeys **true**

If set to true (default), all keys are displayed. If set to false, the Key Management window shows only those keys that match the search terms entered in the field “Search for”.

Enigmail → Key Management → Display All Keys by Default

extensions.enigmail.keyserver ***"pool.sks-keyservers.net,
keys.gnupg.net,
pgp.mit.edu"***

The list of keyserver(s) to use.

Enigmail → Preferences → Keyserver → Specify your keyserver(s)

extensions.enigmail.maxIdleMinutes **5**

Caches the passphrase for the specified time. Default is 5 minutes.

Enigmail → Preferences → Remember passphrase for [] minutes of idle time

extensions.enigmail.noPassphrase ***false***

Tells Enigmail to never ask for a passphrase.

Enigmail → Preferences → Basic → Never ask for any passphrase

Note: This option will **not** be shown if you use GnuPG version 2.x!

extensions.enigmail.useDefaultComment ***true***

If set to false, adds an Enigmail comment in OpenPGP signature.

Enigmail → Preferences → Advanced → Add Enigmail comment in OpenPGP signature

extensions.enigmail.useGpgAgent ***false***

Use GPG-agent to handle passphrases.

Enigmail → Preferences → Advanced → Use gpg-agent for passphrases

Note: if GnuPG v2.0 or newer is used, then GPG-agent is mandatory and the option is **not** shown!

extensions.enigmail.wrapHtmlBeforeSend ***true***

Re-wrap HTML text in signed messages before sending. Default is on.

Enigmail → Preferences → Advanced → Re-wrap signed HTML text before sending

9.2.2. Per Account/Per Identity UI Settings

These settings can be accessed by *Tools > Account settings > (your account) > OpenPGP security*

All following UI explanations start at this point.

Important: You can have multiple identities per account and you should take care to configure all of those you want to use with Enigmail, especially if you added them after running the Enigmail Setup-Wizard.

You can access the settings for additional identities by
Tools > Account settings > (your account) > Manage identities > (your identity)
> Edit > Tab: OpenPGP Security

mail.identity.default.autoEncryptDrafts ***true***

Checkbox: Encrypt draft messages on saving

On by default

mail.identity.default.attachPgpKey" ***false***

Advanced > Checkbox: Attach my public key to messages

Off by default

mail.identity.default.defaultSigningPolicy ***0***

Checkbox: Sign messages by default

0 = Off by default

nonzero = On

mail.identity.default.defaultEncryptionPolicy ***0***

Checkbox: Encrypt messages by default

0 = Off by default

nonzero = On

mail.identity.default.enablePgp ***false***

Checkbox: Encrypt messages by default

Off by Default

mail.identity.default.openPgpHeaderMode ***0***

Advanced > Checkboxes: Send OpenPGP Key ID/Send URL for key retrieval

0 = Off by default

0x1 = Send Key ID from `pgpkeyId` preference below

0x10 = Send URL for key retrieval from `openPgpUrlName` preference below

0x11 = Send both

mail.identity.default.openPgpUrlName **""**

*Advanced > Fill box right of "Send URL for key retrieval"
Empty string by default*

mail.identity.default.pgpkeyId **""**

*Fill box below "Use specific OpenPGP key ID" or click on "Select key..."
Empty string by default*

mail.identity.default.pgpKeyMode **0**

Radio buttons "Use specific OpenPGP key ID" or "Use email address of this identity to identify OpenPGP key"

0 = Get Key ID from Email address (Default)

nonzero = Use Key ID from `pgpkeyId` preference (Recommended)

mail.identity.default.pgpMimeMode **false**

*Checkbox: "Use PGP/MIME by default"
false = Inline PGP (Default)
true = PGP/MIME*

mail.identity.default.pgpSignPlain **false**

*Checkbox: sign non-encrypted messages
Off by default.*

mail.identity.default.pgpSignEncrypted **false**

*Checkbox: sign encrypted messages
Off by default.*

9.2.3. JS preferences

Warning: Editing the following preferences in the `prefs.js` or `user.js` files yourself can result in misbehaviour and data loss when introducing a syntax error! Doing this is therefore **not** recommended for new or average users! If you want to do it

anyway, close Thunderbird beforehand. And, as always: **Make backups!**

extensions.enigmail.addHeaders ***false***

Adds custom X-Enigmail mail headers to all outgoing messages. These headers are not currently used for any function, but may be used by Enigmail in the future.

Currently the added header is:

X-Enigmail-Version: 1.8 (or your current version)

extensions.enigmail.composeHtmlAlertCount ***3***

Sets the number of times a warning message is shown when composing in HTML and attempting to send a signed message using inline PGP.

extensions.enigmail.configuredVersion ***""***

The last configured Enigmail version. This setting **should not** be changed.

extensions.enigmail.displayPartiallySigned ***true***

Handles the display of the Enigmail status bar if only part of the message is signed (this may happen for instance when a person replies quoting a signed message). An example of a partly signed message is shown in the figure at page 66.

If set to false, PGP headers that appear within the message body will be ignored and displayed literally. This does not apply for the PGP headers at the beginning and the end of the message body, which are present there when the message is normally signed in full; in this case, the Enigmail status bar will appear.

If set to true (default), Enigmail even removes the quote prefix character (>) from signed text embedded in the rest of the body if the message itself as a whole is not signed. This only works if the embedded quote has not been modified at all.

extensions.enigmail.displaySecondaryUid ***true***

Forces Enigmail to search your keyring for secondary IDs in order to match the sender address of a received message, and displays it instead of the default key ID.

extensions.enigmail.displaySignWarn ***true***

Warns when you change the signing status by clicking on the sign icon in the bottom right corner of the message composition window.

extensions.enigmail.encryptAttachments **1**

This setting stores the value of the last encryption method used to send a message with attachment.

extensions.enigmail.encryptAttachmentsSkipDlg **0**

Controls whether to skip the attachment dialog, where the user is asked how attachments should be encrypted/signed.

extensions.enigmail.encryptToNews **false**

If set to false (default), won't allow user to send an encrypted message to a newsgroup.

extensions.enigmail.encryptToSelf **true**

Automatically encrypts outgoing messages with your public key, too.

extensions.enigmail.handleDoubleClick **true**

Normally, to decrypt and open an encrypted attachment you right-click on it and select *Decrypt and Open* from the pop-up menu. This option enables automatic decryption and opening if you double-click on the attachment.

extensions.enigmail.initAlert **true**

Displays a warning if Enigmail fails to initialize.

extensions.enigmail.inlineAttachAsciiArmor **false**

Encrypts attachments in ASCII-armored format when sending inline PGP encrypted messages.

extensions.enigmail.inlineAttachExt **".pgp"**

Sets the extension to be appended when creating attachments to inline PGP encrypted messages. Default is `.pgp` (i.e. a file `filename.ext` will be renamed to `filename.ext.pgp` when encrypted).

extensions.enigmail.inlineSigAttachExt ***".sig"***

Sets the extension to be appended when creating attachments to inline PGP signed messages. Default is `.sig` (i.e. a file `filename.ext` will be accompanied by an additional file named `filename.ext.sig` containing its signature).

extensions.enigmail.logDirectory ***""***

Specifies the log directory. Default is empty (unset).

extensions.enigmail.mimeHashAlgorithm ***0***

GnuPG hash algorithm to use. Available values are:

- 0 – automatic selection, let GnuPG choose (default)
- 1 – SHA1
- 2 – RIPEMD160
- 3 – SHA256
- 4 – SHA384
- 5 – SHA512
- 6 – SHA224

extensions.enigmail.mimePreferPgp ***1***

Prefer S/MIME or PGP/MIME.

Due to technical reasons it is not possible to use both PGP/MIME and S/MIME in one message. This option decides which standard is chosen if both are activated (after Per-Recipient rules were processed). Available values are:

- 0: PGP/MIME
- **1: ask (default)**
- 2: S/MIME

extensions.enigmail.parseAllHeaders ***true***

Tells Enigmail to parse all MIME headers in the message. This value is for

testing purposes only, and **must** be left enabled!

extensions.enigmail.quotedPrintableWarn **0**

Issues a warning when Enigmail detects that a message going to be sent contains 8-bit characters and will use Quoted Printable encoding. Default is off. This setting remembers the selected state.

extensions.enigmail.respectHttpProxy **true**

If set, use the same HTTP proxy settings that were defined in Thunderbird to retrieve keys from key servers.

extensions.enigmail.supportMultiPass **false**

Support for multiple passphrase on the same key pair. Reserved for future use; do not change this setting.

extensions.enigmail.useGpgKeysTool **true**

Enables Enigmail to use the `gpgkeys_hkp`, `gpgkeys_ldap`, and `gpgkeys_http` tools to retrieve keys from key servers without using `gpg` itself.

extensions.enigmail.warnClearPassphrase **true**

Has Enigmail pop up a confirmation message informing you that the passphrase has been cleared.

extensions.enigmail.warnGpgAgentAndIdleTime **true**

Warn if GPG-agent should be used, but cannot be found and remember passphrase for X minutes is active (and remember selected state).

Background: If Enigmail cannot connect to GPG-agent although it must be used (using GnuPG 2.0 and later). This is the case when your system uses a specialized tool for passphrase handling such as `gnome-keyring` or `seahorse-agent`. Unfortunately Enigmail cannot control the passphrase timeout for those tools.

Therefore the passphrase timeout settings in Enigmail are disregarded.

extensions.enigmail.warnOnRulesConflict **0**

Issues a warning when sending a message to multiple addresses with conflicting per-recipient rules. Default is 0 (off). This setting remembers the selected state.

extensions.enigmail.warnOnSendingNewsgroups ***true***

Issues a warning when trying to send an encrypted message to a newsgroup.

extensions.enigmail.warnRefreshAll ***true***

Enable a warning about lengthy download when all keys are to be refreshed. This setting remembers the selected state.

extensions.enigmail.warnDownloadContactKeys ***true***

When downloading keys for all contacts, this can be a lengthy process, depending on network speed and number of contacts. This setting remembers the selected state.

9.3. XML format of per-recipient rules

This Section details the format of `pgprules.xml`, which is an XML file generated by Enigmail and containing the per-recipient rules. This is intended as a technical reference for developers only; normal users should never edit the XML file manually.

The `pgprules.xml` file has the following structure:

```
<pgpRuleList>
<pgpRule email='{alice@example.com}' keyId='0x1234ABCD' sign='1'
encrypt='1' pgpMime='1'/>
<pgpRule email='{bob@ {user@domain}' keyId='0xCDEF6789' sign='2'
encrypt='1' pgpMime='0'/>
<pgpRule email='{mailinglist@domain.org}' keyId='0x11111111
0x22222222 0x33333333' sign='2' encrypt='2' pgpMime='0'/>
...
</pgpRuleList>
```

Each `<pgpRule .../>` line is a per-recipient rule stating how Enigmail should enable or disable encryption, signing and PGP/MIME and which key to use. The file is processed sequentially; if a rule contains a key ID attribute with some value, the rule is applied, but the address that matched will not be rechecked in any following rule. The attributes are defined as follows.

`email` defines the recipient address(es) to match. Multiple email addresses are

separated by spaces. The matching is done on substrings, with curly brackets ({}) defining substring boundaries:

- {user@domain} matches exactly and only user@domain
- user@domain} matches anything that ends with user@domain
- {user@domain matches anything that starts with user@domain
- user@domain matches anything containing user@domain

keyId is the list of key IDs to use for the recipient. The key ID is specified in the 8-byte format (e.g. 0x1234ABCD) or in the 16-byte format (e.g. 0x1234567890ABCDEF). Multiple keys are separated by spaces. If a dot (.) is the only value in the field, Enigmail does not use a specific key ID and finds the correct key using the email address. Any further rule for this recipient will be ignored.

sign specifies message signing, encrypt specifies message encryption, and pgpMime specifies PGP/MIME use. All these attributes must have one of the following values:

- 0 – Disables the action even if it was enabled in the Message Composition window. This is equivalent to the *Never* option in the GUI.
- 1 – Uses the setting specified in the Message Composition window. This is equivalent to the *Yes, if selected in Message Composition* option in the GUI.
- 2 – Enables the action even if it was not enabled in the Message Composition window. This is equivalent to the *Always* option in the GUI.

When a message is sent to multiple recipients, and multiple rules are applied, the value 0 overrides the value 2: if one of the rules disables the action, the action will not be applied for the message, regardless of any other rule with value 2.

9.4. Available languages

Enigmail is translated in many languages. The following locales are already included in Enigmail 1.8:

ar	Arabic	ko-KR	Korean
bg-BG	Bulgarian	lt-LT	Lithuanian
ca	Catalan	nb-NO	Norwegian
cs	Czech	nl	Dutch
de	German	pl-PL	Polish
el	Greek	pt-BR	Portuguese (Brazil)
en-US	English (USA)	pt-PT	Portuguese (Portugal)
es-ES	Spanish	ru-RU	Russian
fi-FI	Finnish	sk-SK	Slovak

fr-FR	French	sl-SI	Slovenian
gl-ES	Galician	sv-SE	Swedish
hu-HU	Hungarian	tr	Turkish
it-IT	Italian	vi	Vietnamese
ja-JP	Japanese	zh-CN	Chinese

A note for testers: Enigmail nightly builds may be not fully localized. Untranslated items appear in english.

Advanced users of Enigmail, fluent in english and another language can contribute to translations. Please follow the instructions given here:

<https://www.enigmail.net/download/langpack.php>

10. ACKNOWLEDGEMENTS

This Handbook is the second edition, describing Enigmail version 1.8. It is based on the first edition, but has been overhauled in large parts, reordered and adapted to new features and new user interface, and fixing some errors.

The first edition, written by Daniele Raffo in 2009 and describing Enigmail 1.0, originated from the Quick Start Guide written by Robert J. Hansen, and incorporates technical references written by Patrick Brunschwig (the current developer of Enigmail) and Olav Seyfarth.

While writing the handbook I followed the comments, corrections, criticisms, and encouragements from the whole Enigmail team; the handbook also contains many contributions originally posted by the team on the Enigmail forum, newsgroup, and mailing lists. Thanks to them all!

Thanks to Werner Koch for reviewing the part about the OpenPGP smart card, and for giving permission to use his pictures at page 98.

Michael of frontrowcomputer.com, Christian Marg, and Dave Schaefer also helped with very useful tips and comments.

This document was composed with LibreOffice.org 4 by The Document Foundation.

The icons used throughout this document are part of the Crystal Clear set by Everaldo Coelho, released under GNU LGPL.

The Enigmail logo was designed by Olav Seyfarth. It uses the Good Times font by Ray Larabie, released as freeware.

Thunderbird and SeaMonkey are trademarks of the Mozilla Foundation.

11. THE ENIGMAIL TEAM

Patrick Brunschwig	Project lead, Code maintainer, User Support, Web Site
Nico Josuttis	Developer
Ludwig Hügelschäfer	Developer, Mac Support
John P. Clizbe	Quality Assurance and User Support
Olav Seyfarth	Website and User Support
John W. Moore	Testing and User Support
Robert J. Hansen	Usability and User Support
Daniele Raffo	Forum Management and Documentation

Former team members:

Ramalingam Saravanan	Original author of the Enigmail Extension
Barry Porter	Web Site, User Support
Shane M. Coughlan	Web Site, Testing

12. SUPPORT

This handbook, once read in full, should answer all questions you might have about Enigmail and give you a thorough understanding of it. You can always find the most up-to-date information (and Enigmail executable) directly on the Enigmail official web site:

<https://www.enigmail.net/>

Should you experience problems or want to ask further questions, you can obtain support (provided by volunteers) from several places: there is a Forum, a mailing list, and a newsgroup dedicated to Enigmail. For information about how to access them, visit

<https://www.enigmail.net/support/index.php>

If your question is not strictly related to Enigmail but rather to broad OpenPGP topics (e.g. signing, encryption, key management, trust...), you may also address yourself to the GnuPG resources. GnuPG documentation and support is available at

<http://www.gnupg.org/documentation/index.en.html>

13. HISTORY

Public key cryptography was firstly discovered by James Ellis, Clifford Cocks and Malcolm Williamson of the British Government Communication Headquarters in 1975, but the discovery was filed as classified information and never divulged. The following year researchers Whitfield Diffie, Martin Hellman and Ralph Merkle independently made the same discovery and published it on a paper. One year later Ronald Rivest, Adi Shamir and Leonard Adleman provided the first practical implementation of a public key cryptography algorithm by developing the RSA cipher.

Then in 1991 Phil Zimmermann, a free speech activist and anti-nuclear pacifist, developed Pretty Good Privacy (PGP), the first software available to the general public that utilized RSA for email encryption and signing. Zimmermann, after having asked a friend to post the program on the worldwide Usenet, found himself prosecuted by the government and was even charged by the FBI for illegal weapon export. The charges were eventually dropped, and Zimmermann later founded PGP Inc., which now is part of Symantec Corporation.

In 1997 PGP Inc. submitted a standardization proposal to the Internet Engineering Task Force. The standard was called OpenPGP and defined in 1998 in the IETF document RFC 2440. The latest version of the OpenPGP standard is described in RFC 4880, published in 2007.

PGP is now a commercial product for communication security and privacy in corporate, business and home environment, and is available at <http://www.symantec.com/encryption/>.

Nowadays there are many OpenPGP-compliant products: the most widespread is probably GnuPG (GNU Privacy Guard, or GPG for short) which is developed since 1999 by Werner Koch. The GnuPG Project is hosted at <http://www.gnupg.org>.

GnuPG is free, open-source and available for several platforms. It is a command-line only tool, which means that it does not have a graphical interface.

Enigmail, first released in 2001 by Ramalingam Saravanan and maintained by Patrick Brunschwig since 2003, is an extension for Mozilla based mail clients. Enigmail interfaces seamlessly with GnuPG and provides a GUI to make easy for everyone to securely encrypt, decrypt, sign, and verify the signature on email messages. The homepage of the Enigmail Project is <https://www.enigmail.net/>.

Enigmail, GnuPG and Thunderbird are all free and open-source software. They can be downloaded, copied and used for free. As open-source projects, their source code is available for everyone who desires to examine or customize it.

OpenPGP compliant software in all its variants, is the most famous and widely used public-key-encryption software in the world. Since its creation it has allowed people in totalitarian countries to enjoy privacy, enforce free speech, fight censorship, and protect human rights. It makes use of the strongest ciphers known in the scientific literature, and if utilized properly it is virtually unbreakable.