GPG is the Gnu Privacy Guard, a PGP replacement. What it does is use something called "keys" to enable you to do things that need verification of someone's identity at either end. It uses a dual-key method, with a "public key" and "private key". The public key you will be able to hand out to everyone you know, post on keyservers, etc. The private key should be closely guarded, and never given out. With these then, you can send someone a file, email, etc. and make sure that only they can read it. Additionally, they can make sure that you in fact are the one who sent it. Having such keys is also a requirement for using Launchpad and other portions of Ubuntu management, such as uploading packages, which ensures that all of the software Ubuntu users gets comes from trusted sources.

How you use them is up to you. I for instance usually 'sign' all of my e-mails (people can verify they came from me), but rarely would 'encrypt' them (make only certain people able to read them). This of course depends on your personal needs, and we will discuss these two actions further. I will begin with the command line way of doing things, but don't worry, we will get to the GUI. I start here because I think it gives a good background, and some personally prefer it. If you're comfortable with cli, that's great. If you'd rather not go there, don't worry about remembering the commands and such too much - pay attention to what I'm saying about the process, as this is largely the same.

The tool that you will use for this is 'gpg', supplied by the gnupg package, which should be installed by default in Ubuntu. It is also possible to use this outside of Ubuntu, even on Windows, but that is beyond the scope of this class. Now, to really get started: The first thing you will need to do of course is to generate a key pair for yourself. This is done on cli with '`gpg --gen-key`'. You will be given a choice about what sort of key to generate, DSA and Elgamal, DSA only, or RSA only. Use the default of DSA and Elgamal, as this is the only configuration that will allow you to encrypt - the others only provide signing capability. It will then ask about the key size. The default is 1024, which is usually fine, since above that other aspects of the system will be a greater security risk than the key length anyway, but you can use 2048 if you'd like. The next question asks how long the key should be valid for. The default is 0 (infinite length), but I highly recommend **not** using the default on this particular question. The reason for this is that after you have uploaded the key to keyservers and that sort of thing, if you ever forget your passphrase to the key you will be unable to revoke it, and a stray key will be floating around just cluttering up the world for no good reason. Sensible things to use would be perhaps one or two years, which you would enter as 1y or 2y respectively. Other lengths of time are described in the man page if you're interested. (to access the man page - run "`man gnupg`" (without the quotes) from a console.

This way, you can simply mark the expiry date on your calendar, and as it approaches if you still have your passphrase, you can modify the expiration date and resubmit it to the keyservers, thus ensuring that you don't leave ghost keys, but also letting you keep the same key for a long time. The one downside is that others will have to refresh their lists, but they should probably be doing that anyway, or else you can certainly ask them to do so if they run into problems with your key.

Now at this point you will be asked about your name, e-mail address, and comment. It is important that for your name you use your legal name as it appears on your drivers license, passport, etc., for reasons that will become obvious later. For the e-mail address just use something you actively check, and you can edit it later in the same manner as the expiry if necessary. Comments can be something like "Package signing key", "Test", "tonyyarusso", "E-mail verification", or whatever's useful to you. It will then show your selections, in the form "`Name (Comment) <e-mail address>`", and you have the option of continuing or modifying what you set. At this point continuing will go through the actual creating of your key. This will involve grabbing random data from the system to make it secure, and doing this from a dormant machine can take a while, so letting your cat walk across the keyboard, moving the mouse, etc. (or just hitting keys with no output like Ctrl, Alt, and Shift) will speed up the process.

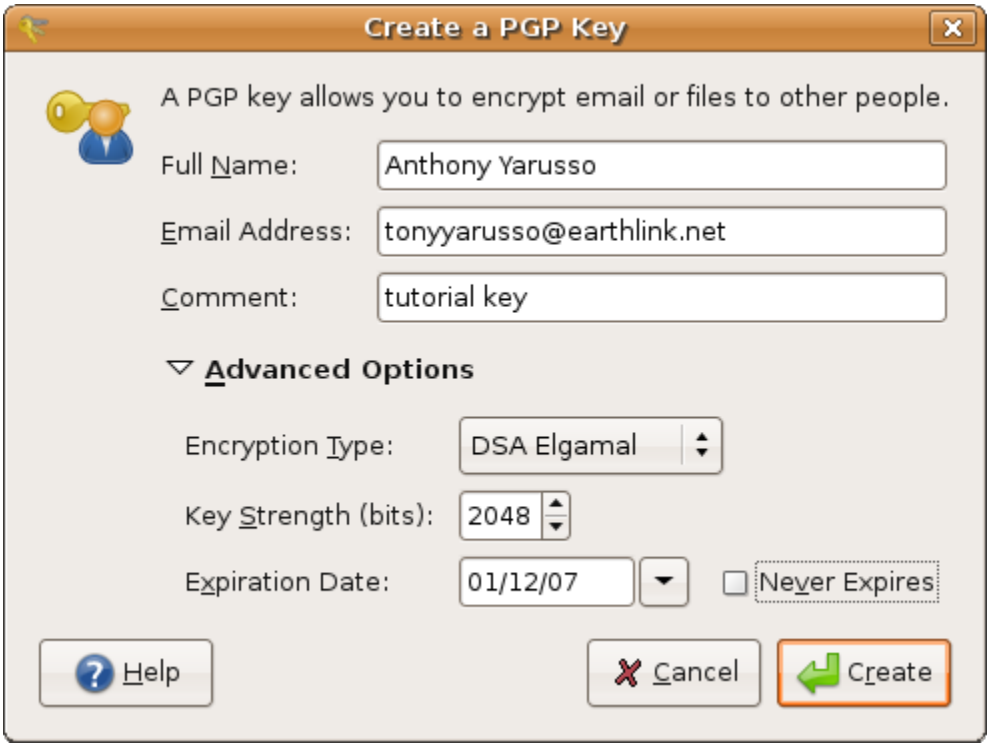
The next step is your passphrase. Note that this is a passphrase, not a password, and so can be a short sentence or something. Use something long enough to be secure, and not a simple dictionary word, as cracking those is trivial. The longer the better, so your memory is the limit. You will need to enter it twice to confirm and make sure you didn't make a typo. You will then get some output listing key info - this means you've successfully created your key!
You may find it useful to add the key ID to bash, so that certain things use it automatically. This is done by adding a line of "`export GPGKEY=D8FC66D2`" (without the quote) to your `~/.bashrc`, where `D8FC66D2` is the key ID you were just given in the previous output. You can also type the same thing just on the command line for the current session, as .bashrc won't load changes until your next login.

Like I mentioned before about losing your passphrase, you can still revoke it if you have a revocation certificate. You can create one with the following command: '`gpg --output revoke.asc --gen-revoke <KEY-ID>`', where `revoke.asc` is the file you're creating. You can print this out and store it in the fire safe in your basement, but keep it carefully protected, as anyone who has access to it can render your key unusable.

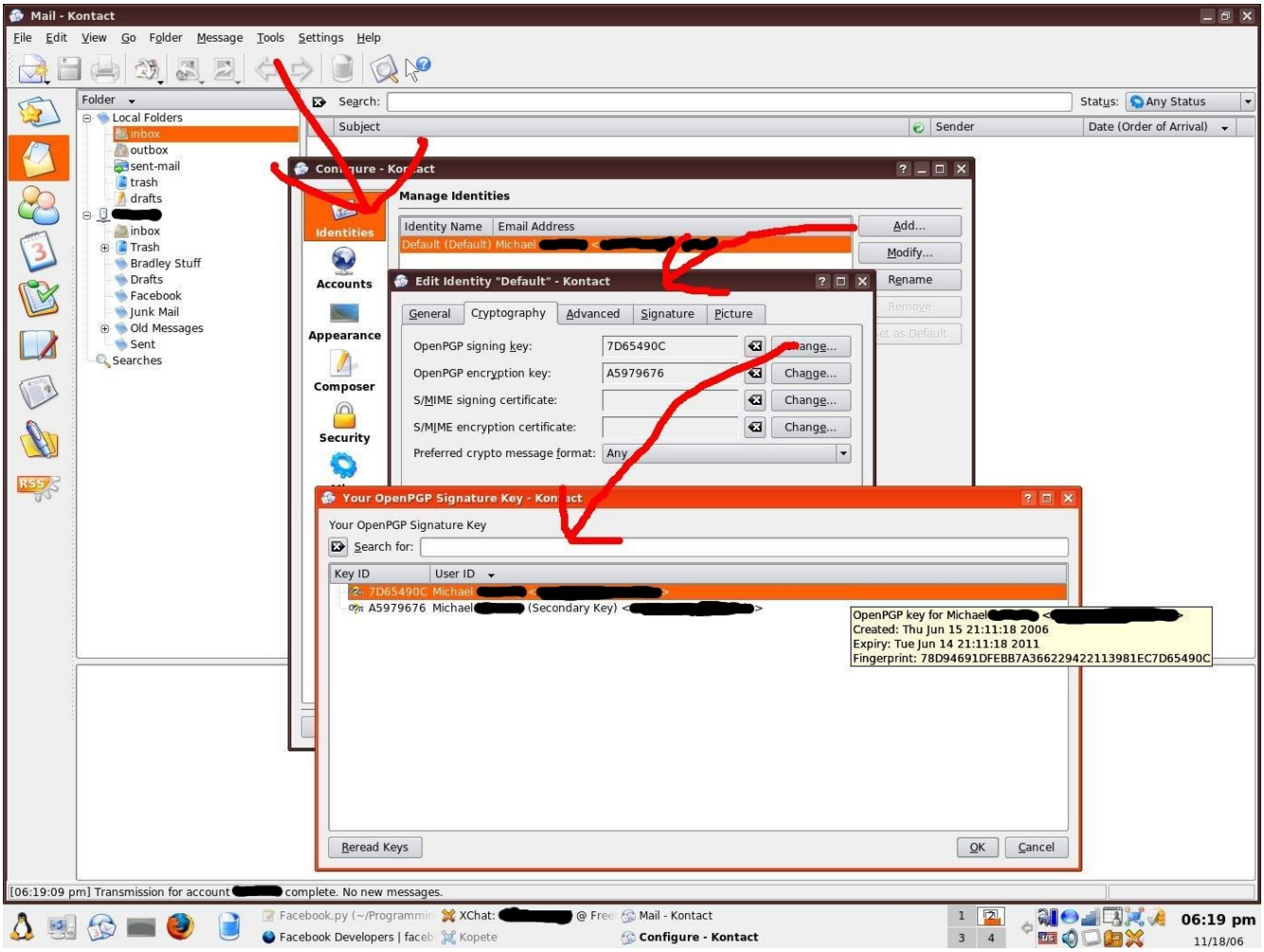
Now it's sharing time! Your key has limited uses if nobody has the public key, so it's time to start handing that out. (Remember, this isn't the private one) Ubuntu maintains their own key server that you can send it to, and keyservers generally sync with each other, so once you've done one, they do the sharing work for you. To send your key to it, use '`gpg --send-keys --keyserver keyserver.ubuntu.com <KEY-ID>`'. If you would rather start with another server, that works too. Now, before we go into things you can do with it, I'll try to cover the GUI methods to an extent.

In Gnome, there is an integrated application called Seahorse (package name is seahorse). This will give you a nice interface to basic key management, as well as working with other parts of the Gnome desktop. It will show up in Ubuntu as "Seahorse" or "Encryption Keys" under Applications > Accessories, depending on your version of Ubuntu. The above process is easily replicated here, with Key >

Create New Key in the menu. Select PGP key as the kind you wish to create. Selecting Continue will show you the standard Full Name, Email Address, and Comment options. You will also want to click "Advanced Options", as that is where you set the Expiration Date, if you choose to do so. Type and length (strength) are there as well. Once you've done all of your options on that one screen, click Create, and see previous note about randomness. You will then have your newly created key available in the Key Manager window to reference at any time, under the My Personal Keys tab. You'll notice there are also tabs for Keys I Trust and Keys I've Collected, which come from things we're getting to shortly. There are also key management frontends available for some e-mail clients, which is also coming up, so if you find you like those, you could potentially skip Seahorse and use them.



In KDE, the graphical interface is in the package 'kgpg'. In KDE, as tonyyarusso said, the graphical interface is called "kgpg" (you can find this under adept). Once kgpg is loaded (K Menu -> utilties -> kgpg) you will see a small padlock icon near your clock. Clicking on this will bring up the "Key Management" window. From here, you can manage all your keys. To generate a key, click on the "Keys" menu, and select "Generate Key Pair", which will open a window asking for your name, your email, an optional comment, the expiration for your key, key size and alogorithm. Use the same values here as you would have generating a key by the command line. Once you click ok, you will be prompted to enter a passphrase, and shown a graph indicating the quality of your passphrase. Once you have entered this (it must be 5 characters or more) then you will be sent to a screen showing "generating key pair". Here, the previous note about randomness should be considered. Once you are done with that, you will recieve a confirmation of your new key being created, with info regarding the key (and your key id). Click ok, and your key will show in the main key management page of kgpg.

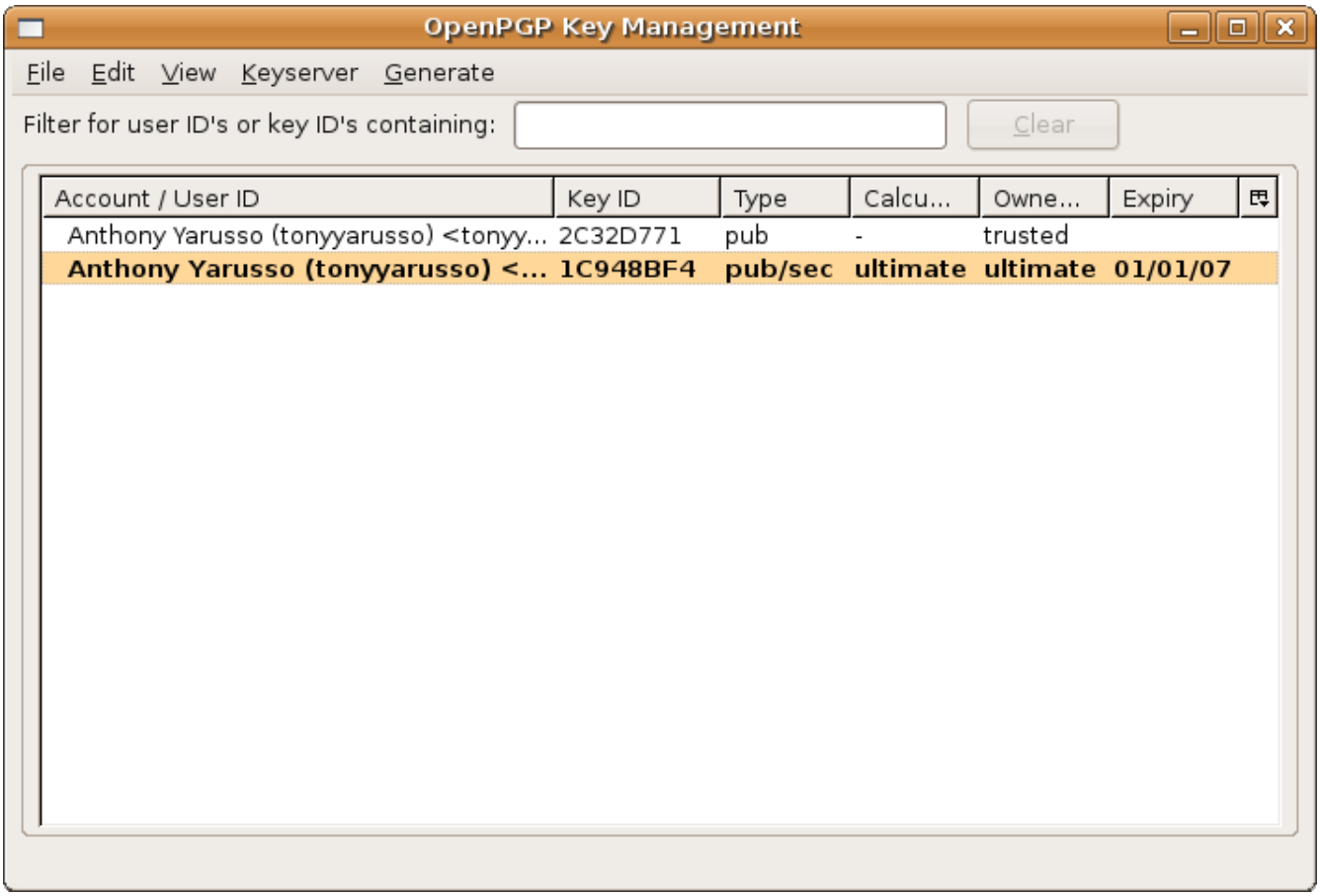


Okay. Seahorse also allows you to do things like syncing your keys to a remote key server, searching such a server for someone else's, and modifying things like the expiry, e-mail address, etc. (even adding a picture of yourself if you want), which are fairly obvious with a few moments of clicking around (equivalent to reading the man page before).

Now we'll get into the first application: e-mail. Evolution, Kontact, and Sylpheed all have encryption and signing capability built in. Mozilla Thunderbird has it available through the enigmail extension, from the 'mozilla-thunderbird-enigmail' package. In Kontact, look under Tools, in the Configure thing, in Identities, and click Modify. Here, look under the Cryptography tab, and you'll see some lovely options. Use the "change" button by OpenPGP encryption and/or signing key and select yours from the list.

In Thunderbird, you will get a new top-level menu called OpenPGP. Under this, "Key Management" will get you a screen like the others. There are also other things you can set under preferences, that are either self-explanatory, or not really necessary to change. Poke around if you like. Once you have your key added, under Account Management each of your TB accounts will have a section called OpenPGP Security. Here you can set whether to sign or encrypt messages by defaul, and the key to

use, etc. Remember that signing is making it possible to verify that you sent the message. Also, when someone sends YOU an encrypted email, they've used YOUR public key. You use your PRIVATE key to decrypt.



Evolution is similar to Kontact. In Evolution, go to the edit -> preferences menu. Here you should see a list of accounts you have set up. If you select the account you wish to use your key with, and click the edit button, you will see a new window with a list of tabs. If you select the "security" tab, the first option here is "PGP/GPG Key ID". Input your GPG Key id here (just the short ID, though it should also accept the fingerprint (without spaces)), and then check the options below as you want to set them, click ok, and your settings are saved, and GPG is set up for that account. If you dont want to sign/encrypt by default, when composing a new mail, the security menu gives access to those features.

Encrypting will actually turn text into gobblydegook, which will only be reversible by the person you encrypt it for. Signed messages are signed for everyone, and the only consequence is that people who don't understand PGP might comment that there is a weird section about your signature in the message. Beyond just using them for your e-mail messages, you can sign and/or encrypt individual files too, for attaching to e-mails, DCCing to someone, transferring on a USB stick, or even just on your hard drive (for things like protecting sensitive files if you lend it to a friend or bring it in for repairs).

Questions: (that don't fit well anywhere else)

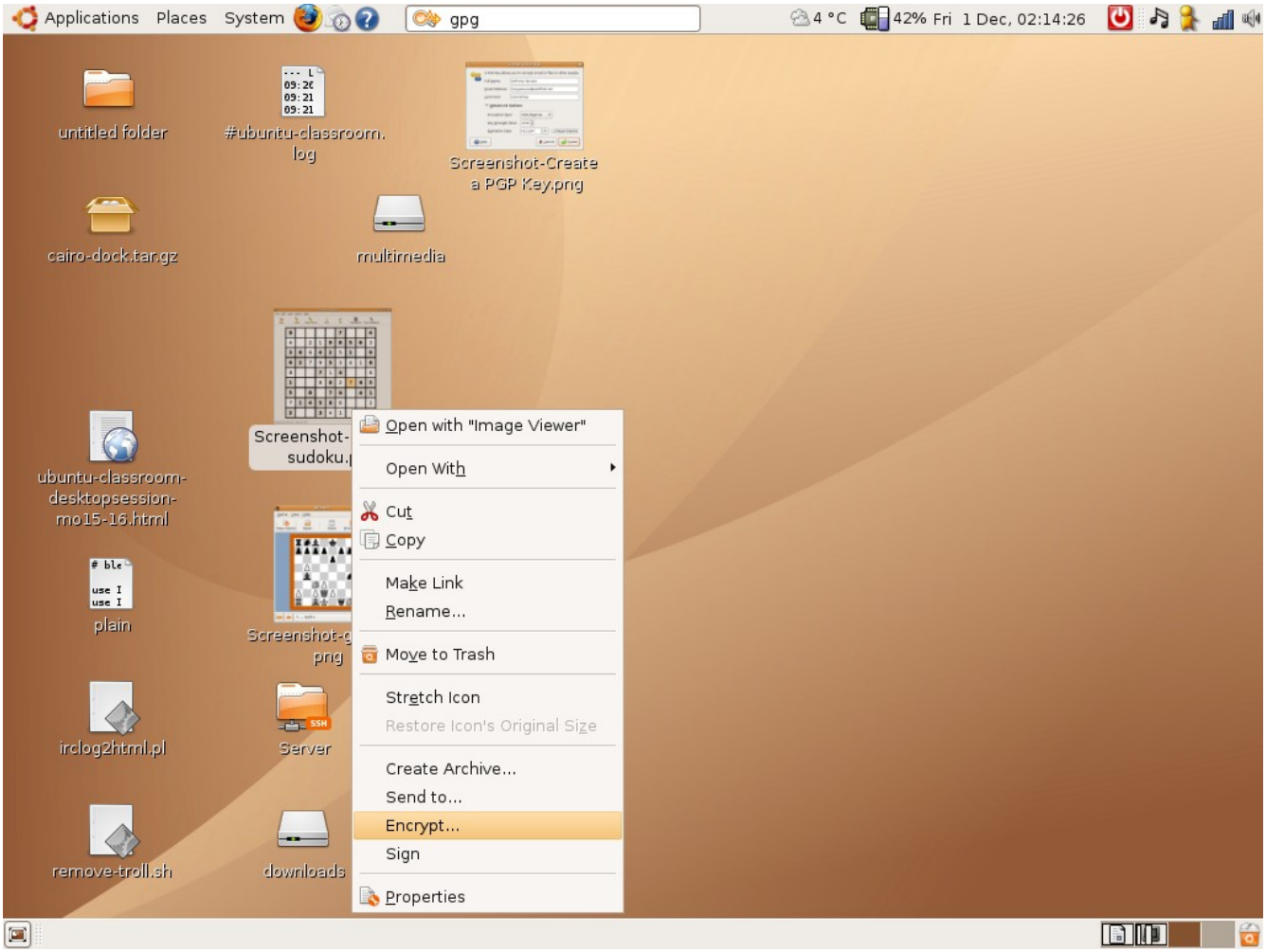
“Can you start your own key server for just your internal network so to have only certain people can host theirkeys and be identified?”
I suppose potentially you could, but I am not sure what purposes that would be necessary for. The only risk to using the usual ones is that your name and e-mail address are published. You would have to set up a server machine of course, and I am not sure what other software it runs on, but in short, if someone can do it, surely you can too. Your best bet is to look into "onak" which is a PGP key server - the package is in ubuntu. You can also only give your key to your friends, if that works.

“What is there for web-based mail like gmail?”
I found this for gmail - http://www.langenhoven.com/code/emailencrypt/gmailencrypt.php. Also, some webmails, including Gmail, also allow POP or IMAP access. For the general case still: If this is the case, you can always use one of the aforementioned clients as well.

Now, signing and encrypting files. To sign a file, we'll start with the command line again. With gpg it takes the syntax of '`gpg --option optionvalue command file`', generally. For this case, '`gpg --sign yourfile`' is the way to add a signature, or `--clearsign` to add a clear text signature. Some options also have short forms, for instance `-s` is equivalent to `--sign`. `--clearsign` is the method requested when replying to the automatic e-mail message from Launchpad to confirm a key that you add to your account there. You can also specify the output filename, with something like '`gpg --output something -s originalfile`'. Note that the options like `--output` come before the command, `-s`, and the file is still last. You can also make a detached signature, and just distribute this with a file separately, with `–detach-sign`. As far as encrypting files goes, '`gpg --output mysafefile --encrypt inputfile`'. You will then be able to decrypt with `–decrypt`. If you have multiple keys, you will want to know how to change which is the default and which one you're using, but I don't want to bore folks with commands, and the full details are on the gpg man page.

Now, graphically in Gnome. As I mentioned, Seahorse integrates quite nicely. If you have it installed, you can be in Nautilus (your file browser), and right-click a file. You will see options to Encrypt and to Sign the file in the context menu, and will simply have to select the key to use. If signing, you just select yours, and if encrypting, select from the list of collected keys those of anyone you are sending the file to. You can search for them by name, e-mail, and the like, which is where the handiness of the keyservers comes in. Note that if you just installed seahorse during this session, you will not see these options until Nautilus has restarted. You can accomplish this with the command '`killall nautilus`', as it will automatically respawn. There is also a plugin for Gedit under Preferences > Plugins that allows you to encrypt a section of text. Also requires gedit to have been restarted since seahorse installation. Combine those with other nifty things, and you can just right-click > encrypt your file, and send it away, say with right-click > send to > gaim

contact, or whatever.



With KDE, again, it's pretty simple to encrypt things. If kgpg is installed, you can right click on a file, and select actions -> encrypt file, and this will give you a list of people to encypt the file to, select the person, and click ok, and you're done. You can also drag files onto the padlock icon in your taskbar, with the same result. For text, you can encrypt/sign anything in your clipboard simply by right clicking on the kgpg icon and selecting "Encrypt Clipboard" or "Sign/Verify Clipboard" repsectively. This will open an editor and prompt you on what to do.

Now, getting your key signed. For some purposes it's good enough to just sign stuff with your key, and mention to your friends what your key ID is or something, but for things like uploading packages for inclusion in Ubuntu, we actually need to verify it. This requires the following steps:

- Find somebody who lives near you. Possibly the most difficult-sounding, but luckily someone has made this simple. A site commonly used for finding people to meet is http://www.biglumber.com/, which will help you locate a number of people in your town, or at least a reasonable drive away, depending on where you live.

- Once you've located a person, you contact them and nicely ask for a meetup some time. Things like UDS, LUG meetings, and that sort of thing are also good opportunities for getting your key signed.
- You will need government-issues photo ID so that the other person can verify your identity. This is how we know that someone who has signed your key has absolutely verified that you are who you say you are, hence having the name on the key match your real name.
- Also, print off the "fingerprint" of your key. This can be done with '`gpg --fingerprint`' and the like. This is what they will use to match to your key.
- When you meet up, simply check their photo ID and have them check yours, and if that's all good, swap fingerprint printoffs.
- Verify that their fingerprint matches what you've got on your printout first. Then, sign that person's key, send the signed copy back to them, and sync your keys list with a keyserver. The servers will add the signature. All done.

You can sign a persons key through the GUI, or with `gpg --sign-key keyid`

Another time you may need to verify yourself is if you ssh into a server that requires your key for authentication. I won't go into that, but details are on https://help.ubuntu.com/community/GPGsigningforSSHHowTo. By having someone else sign your key, and other people signing theirs, and so on, all by meeting in person, you build what's called your 'web of trust'.

Once you look up someone's key, you need to add it to your keyring to use it to encrypt things to them and easily verify signatures from them. From their Launchpad page say, once you've clicked on it you're brought to http://keyserver.ubuntu.com:11371/pks/lookup?search=0x6AAAA569&op=index. From there if you click on the key ID, you should get a huge block of text (which is my key). Copy this starting at `-----begin pgp public key block-----` and ending at `-----END PGP PUBLIC KEY BLOCK-----` and then go into your GUI for GPG and there should be an option to import somewhere. In kgpg, it's under keys->import key -> from clipboard. That's manually; usually, you should just be able to do `gpg --recv-key keyid.` Seahorse has Key > Import in the menu, then you just browse to the file you saved it in. For example, to get Mez's key just run `gpg --recv-key 6aaaa569.` Nalioth has used both kgpg and seahorse, and prefers to gpg from the terminal or enigmail with thunderbird. Of course, if they've published to a keyserver, in Seahorse you can just "Find Remote Keys", search for tonyyarusso, select, and click Import. There are also options in kgpg to import from a keyserver, and gpg on command line also has search functionality.

If you upgrade or reinstall Ubuntu, be sure to back up your ~/.gnupg directory to save your keys.
For referencing this topic later, I suggest you bookmark https://help.ubuntu.com/community/GnuPrivacyGuardHowto
I suggest you also bookmark http://www.gnupg.org/(en)/documentation/faqs.html

tonyyarusso> If you need to contact me later, or want MY public key, see my LP profile: https://launchpad.net/people/tonyyarusso/, and/or associated wiki page.
Mez> and my Launchpad page is https://launchpad.net/people/mez/