pretending to be you; not good. Anything from this point is damage limitation, but you can prepare for and prevent this potential calamity ahead of time by creating a **revocation certificate**. A revocation certificate is a signature on your public key which basically says "I'm not valid, you can't use me", and when updated, prevents anyone else from using it either. You create this certificate as soon as (or soon after) you generate the original key-pair, and then keep it safe.

You may have noticed the option of the 'Expiration date' when you created your Key Pair - this is quite important.

Make sure you have a backup of your Public/Private Key Pair somewhere safe. But you already take full backups regularly anyway, don't you? :-)

Somewhere mention about saving drafts locally, not on the server?

GNU Privacy Guard: A CompSoc guide to daily use of email encryption

Mac Edition

Martin Dehnel
James Fielder
(Durham University)

May 2012

What is GPG?

GPG, or GNU Privacy Guard is, in a nutshell, *a free and easy way to send and receive emails completely securely*. The way most emails are currently sent is completely insecure, and is directly analogous to sending all of your mail on a postcard, available for any postman or eavesdropper along the way to read: we don't think this is good enough, and want to encourage more people to think and care about their privacy online.

GPG allows you to send **completely secure** emails to anyone else with an email address and a key: no special email provider is needed.

How does it work?

To make sure that only the intended recipient can read the email you've sent them, you need to encrypt the message. This means that to anyone without the right password (or 'key') the message will look like random gibberish.
As you may or may not have ever met the person you want to email, you probably won't have a way of agreeing a password without knowing for certain that no-one can intercept it (meaning they could read your messages too). Instead, as an analogy, you create an *electronic padlock and key*, or **Public / Private Key Pair**. The Public Key is the padlock, and the Private Key is the key you use to unlock it. You mustn't send anyone a copy of the private key over the internet as someone could copy it, but instead you can send out an unlocked padlock (the public key) to anyone who wants to send you an email; this way they can 'lock' (encrypt) the message with the padlock, but no-one, not even the sender can 'unlock' (decrypt) it – only you, the person with the private key can do that.
So that anyone can send you an email securely, you distribute your 'open padlock' (Public Key) freely using a Key Server, such as http://pgp.mit.edu. Anyone can type in your name or email address and find your public key this way, but don't worry, there's no way anyone can work backwards from the Public Key to work out anything about your Private Key.

Setup and usage guide, Mac edition

(If you are using Windows or Linux please go to CompSoc's GPG website, http://gpg.compsoc.dur.ac.uk. Also go to the website for full screenshots of this walkthrough.)
Please don't be put off by the length of this guide – most of the steps are completely obvious when in GPGTools.

1. Download and install the GPGTools software from **http://www.gpgtools.org/**
2. Open '**GPG Keychain Access**' from the 'Applications' folder - this is like Mac OS X's Keychain, but for Public / Private Keys. It will store your Public and Private Key Pair (in **bold**), and allow you to send emails to the people for whom you have public keys.
3. On opening GPG Keychain Access for the first time, you'll be asked to generate a Public/Private Key Pair. Enter your name and email address, and then choose 'Advanced Options', specifying a Key Length of **4096** (this is the strongest type of key). Leave everything else.
4. Click 'Generate Key' and then when it starts working, bash away randomly at the keyboard for a minute or so: generating random data on the computer helps ensure that your key is as secure as possible.
5. Think of and enter a secure passphrase – this isn't your Private Key, this is another layer of protection to keep your Private Key secret, but it is still **very** important. Make sure the passphrase is at least 10 characters long, and don't forget it! Enter it again for confirmation.
6. You'll now see in the Keychain window a key with your name and email address in **bold**. You're all set to go!
7. Now you just need to look up keys for anyone you might want to email. Add 'pgp.mit.edu' to the Keyserver field in 'Preferences' (Cmd + ,), and then choose 'Search for Key' (Cmd + F) to find someone else's public key. Try searching for 'gpg@compsoc.dur.ac.uk' - that's us. If this doesn't work, go to http://pgp.mit.edu and search for it manually; you can then save this locally and import it into GPG Keychain Manager for use.
8. Assuming you have your emails running in Mac Mail, open it up (from scratch) and you should now be able to send an encrypted email to us. In the new email window, you should now see a locked padlock and a star with a tick in it below the subject line (right hand side) if you've entered the email address of someone whose key you have. Give it a try - send us an email! See our website for other clients.
9. You'll have to enter your passphrase every time you send or receive an encrypted email - when you receive one, this is to unlock your private key to decrypt the message. When you send one, this is to unlock your private key to sign the email, to prove that it was you who sent it.
10. Finally, if you can't find your own email address in the keyserver, you may have to upload it manually – Durham's firewall doesn't always play nicely with keyservers. Try 'Send to Keyserver' in the 'Key' menu (Cmd+Shift+K), or if that doesn't work, 'Export' your key (**don't** choose "Allow secret key export"), and then copy and paste the contents of the text file into pgp.mit.edu's "Submit a key" box.

Trust

So what happens if you receive a signed email from someone claiming to be ~~a rich african prince~~ important: how can you really know whether they are who they say they are? Obviously the only way to *really* know who they are is to have met them, and seen their passport or other form of photo-ID. But if you've never met them, you have to take their identity on trust, unless you can find another way of validating it. What's the easiest way to do that? Find someone else who has met them, someone who will vouch for them. This is the basis behind the concept of **signatures** for Public Keys; anyone who you've met in real life (or who trusts that they know your identity) can sign your keys to say, e.g. "This is really Martin Dehnel's Key Pair". This way, if you've never met me, but have met and trust James, and you see that James has signed my public key, you can be reasonably sure that my public key was really generated by me, and not by someone pretending to be me. When someone else signs your public key, you should re-upload it to the key-server, so that other people can see these signatures.

Revocation

So what happens if someone gets hold of your private key (and passphrase)? Obviously, they would be able to read all of your über-private encrypted emails, and send emails