Open Source Tools for email Security with Mail.app in Mac OS X

NebraskaCERT, Omaha
August 11, 2005

Mat Caughron, CISSP
PHP Consulting

Public Service Announcement

Please silence all pagers

and cell phones now.

Thanks!

Disclaimers

Only You Know Completely What Your Risks Are.

Cryptography is a Powerful Tool.

All Tools Have Limitations.

Remember: Security Transcends Technology.

Why get detailed with GPG?

Robust crypto is easier to use now than ever before and even the human interface components are becoming open source.

OpenPGP standard co-authored by PGP people even after merging with NetworkAss.
http://www.faqs.org/rfcs/rfc2440.html

If you're not using robust hashes for software release, you are failing to follow best practices.

More reasons to cover GPG:

- Who here uses email? Who trusts email? What are the limitations of this medium?
- non-repudiation has both good and bad effects.
- How can you have a good security conference without a key signing? Perhaps next year we can organize one?
- Imagine if all cryptography frameworks were commercial or closed source? A publicly auditable PKI system is a good learning tool (if nothing else.)
- Finally, raising the level of paranoia will only get you so far. GPG is a tool for letting you do something about it.

email Security from Several Years Ago...

- VPN's were not available and application-level encryption was not popular.
- Most passwords were sent in the clear.
- When encryption was used, it was difficult to find public keys for people at large.
- As email has evolved into one of the primary means of corporate communication this situation is less tolerable.

Some Improvements on the Protocol Level

- IMAPS - secure IMAP protocol over port for thin-client email access via port 993. A wide variety of authentication methods is available.
- SMTPS - secure SMTP uses OpenSSL or equivalent for sending mail, via port 443.
- Authentication methods can vary for the above.

Commercial Encryption Tools

- PGP uses IDEA which is patent encumbered.
- It is possible to compile IDEA into GPG.
- Apple includes x.509 email certificate PKCS support in Mail.app and it works very well.
- Risk aggregation of Thawte free email certificates and similar notarization notarization is controversial. Do you trust the root certificate authorities? To what extent?

Email Encryption with Mail.app (According to Apple)

1. The x.509 framework encryption is NICELY integrated into Mail in Panther and Tiger.

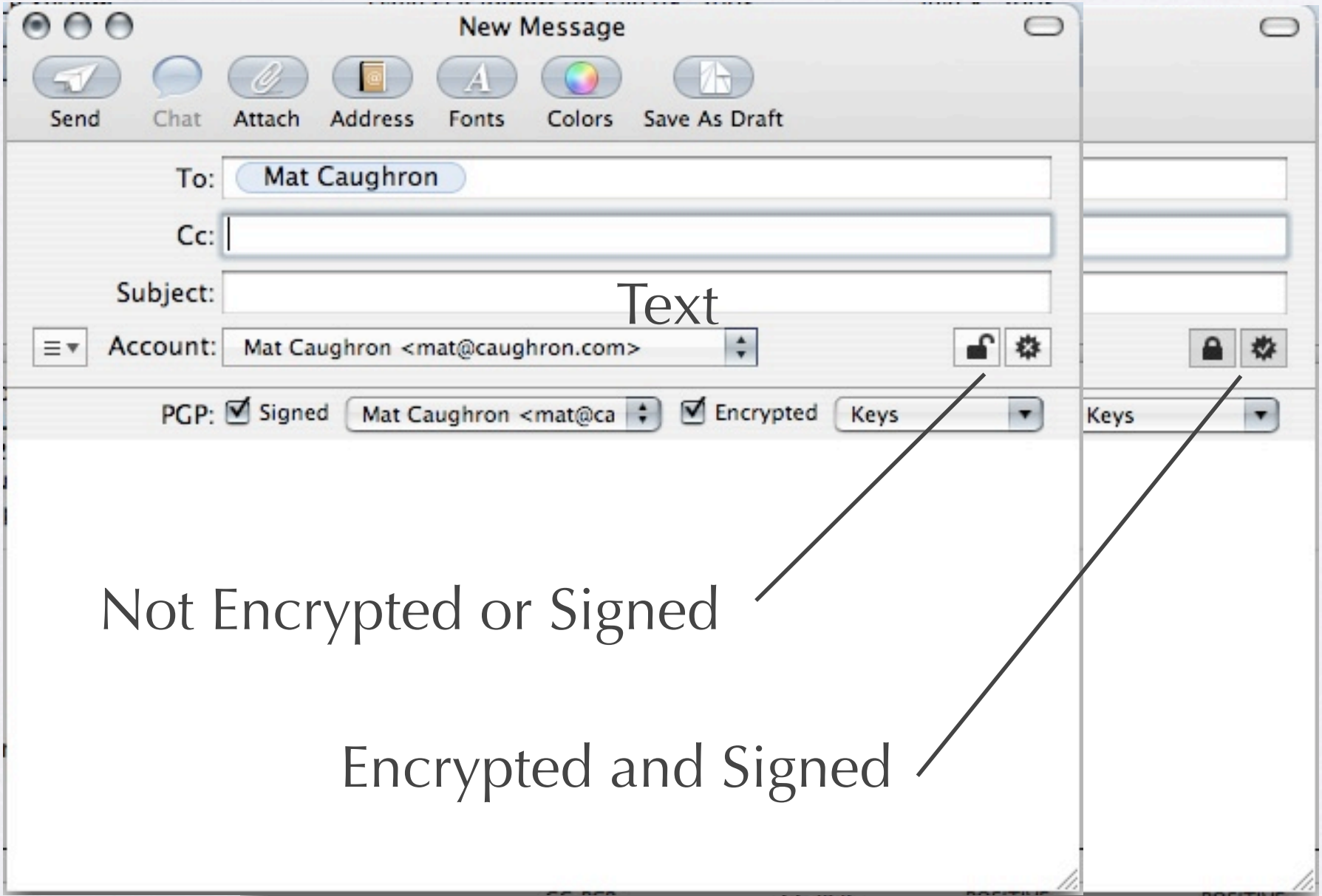
2. Private key setup is accomplished with the Keychain Access application.

3. Public key exchange occurs automatically with the reception of a signed message, and this is integrated with Keychain.

4. Cryptographic strength is limited by keysize, highest I've seen is 2048 bit.

5. Let's take a look...

x.509 Certificates in Action



GPG versus X.509

- Key management with GPG is manual.
- Key management with x.509 is automatic.
- GPG is Open Source.
- Apple's implementation of X.509 is not.

Today

- Email Encryption Overview
  - Symmetric or Private Key crypto
  - Public Key crypto
  - ABC's of GPG
  -
- The focus today is Mail.app
  - But Thunderbird from Mozilla is nice too!
  -
- Demonstrations of GPG Integration with Mail.app

Symmetric Encryption with a Single Key

Classic methods for encryption only use one key for encryption. The sender encrypts the message with this key. To be able to decrypt this the receiver needs to have this very same key.

Problem: This key must have been given to the receiver in a way, that others won't have had the opportunity to obtain this key. If somebody else does have the key, this method of encryption is useless.

Public Key Cryptography

The use of so-called Public Keys solves the secure key exchange problem.

Two keys are involved: one key is a Public Key that can be spread through all sorts of media and may be obtained by anyone. The other key is the Private Key. This key is secret and cannot be spread.

The private key is only available to the owner. When the system is well implemented the secret key cannot be derived from the public key. Now the sender will crypt the message with the public key belonging to the receiver. Then decryption will be done with the secret key of the receiver.

Lockbox Parable and the Key Management Challenge

- PKI is the equivalent of sending a package to someone with a lockbox in it, to which only you have the key.
- They can communicate with you privately but for you to communicate with them privately, they would need to send you a lockbox to which they have the key.
- The challenge of key management involves holding keys for everyone with whom you communicate.

About GPGMail

- written by Stephane Corthesy in Switzerland
- www.sente.ch/software/GPGMail
- current, stable versions available for 10.4 Tiger as well as 10.3 Panther
- Adds a nice configuration pane to Mail.app's preferences pane.

GPG versus PGP

- GNU Privacy Guard is Free Software, and is licensed under the GPL.
- PGP stands for Pretty Good Privacy and is a commercial application.
- These are largely interoperable thanks to the S/MIME standard.
- Google is funding the development of a clean-room BSD-licensed version of GPG this summer in collaboration with NetBSD developers.

BSD licensed privacy guard (pgp) www.netbsd.org/Foundation/press/soc.html
Student Internship for Manuel Freire To be mentored by Alistair G. Crooks, Curt Sampson

How to Install GPGMail

You need to have GnuPG installed first. This can be done in a variety of ways: Fink, Darwinports, MacGPG, compile from source yourself, etc.

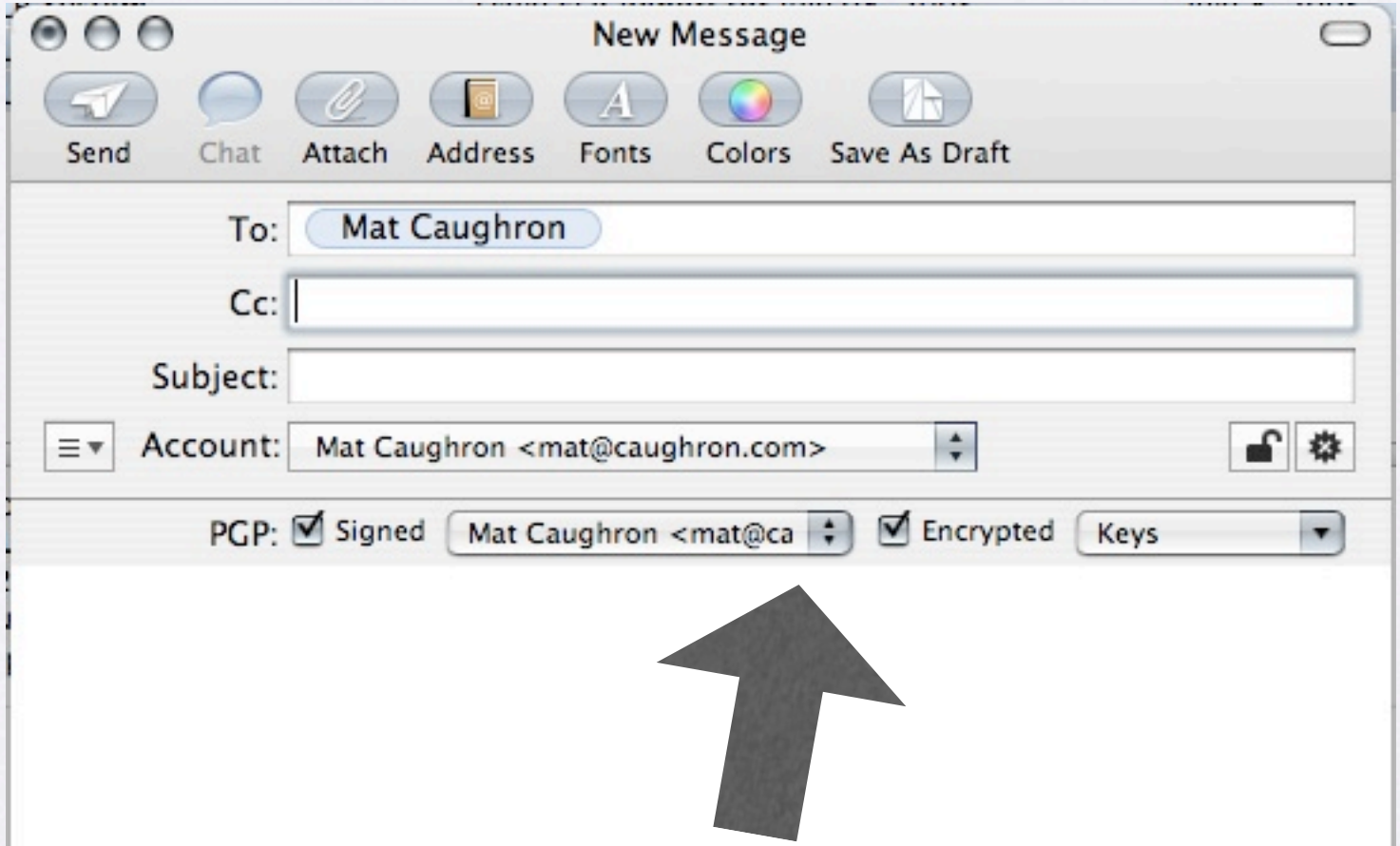
1. Create, setup, or import your private GPG key

2. Copy **GPGMail.mailbundle** into **/Library/Mail/Bundles**

3. Here's the trick, in a Terminal window type:

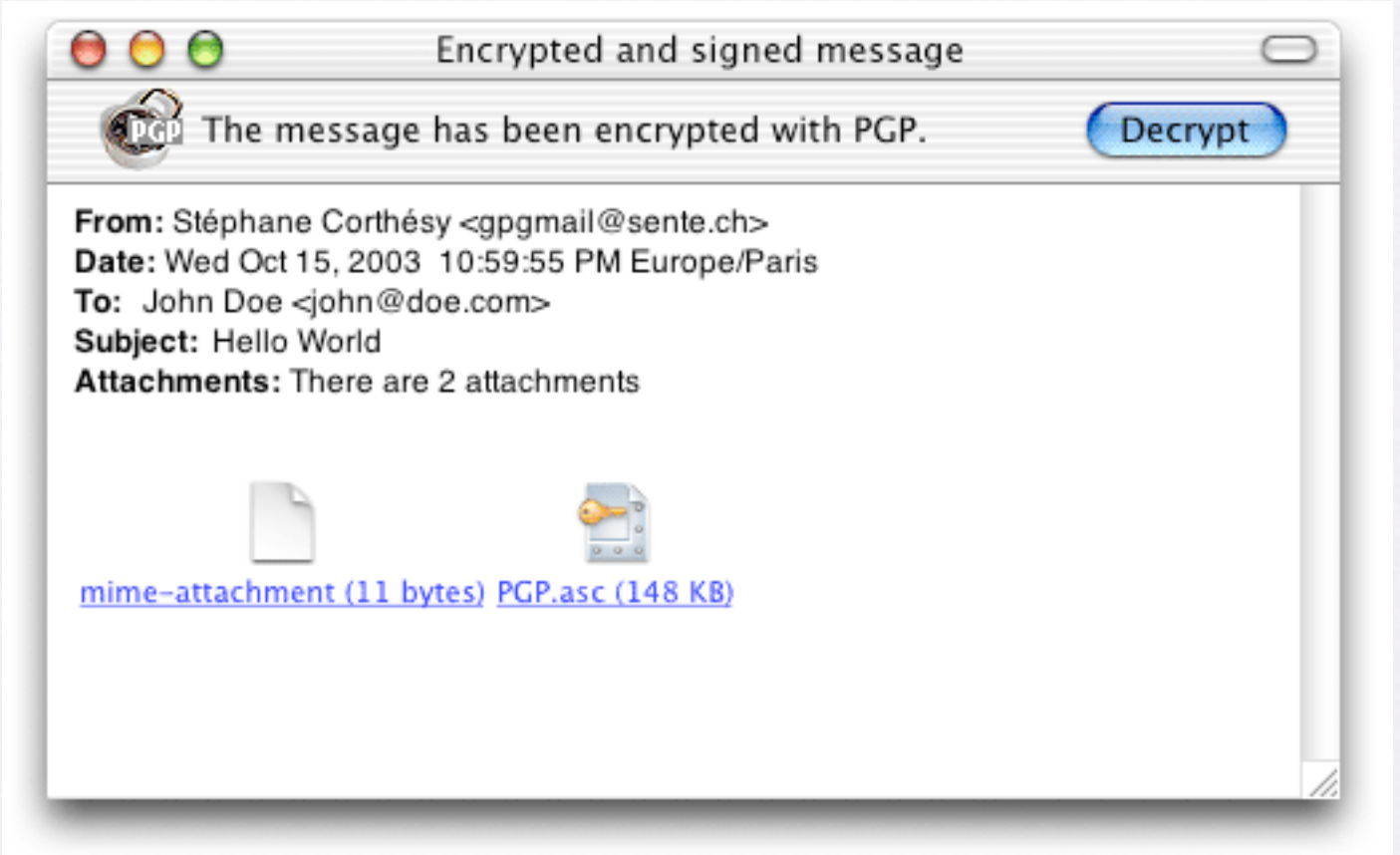
```
defaults write com.apple.mail EnableBundles YES
defaults write com.apple.mail BundleCompatibilityVersion 1
```

GNU Privacy Guard Plug-In for Mail.app



The Selection of Keys for
Both Encryption and Signing is Allowed

Incoming Message...



What GPGMail Does

Decrypts messages from others encrypted with your public key
Authenticates signed messages from others using your copy of their public key
Encrypts new messages using other's public keys
Lets you choose from among your encryption keys
Provides passphrase encryption

Can automatically sign new messages
Also can be used to sign keys
And dynamically refreshes keys that change

Some GPGMail Limitations

GPGMail does not support PGP key distribution (see RFC 3156)

GPGMail encrypts/signs the whole message, and can decrypt/verify only the whole message. You can not choose which part you want to encrypt.

The encryption operation cannot be interrupted.

You cannot redirect a PGP-MIME signed message without loosing the signature.

Encrypted messages are stored encrypted and are not indexed by Mail.app. Perhaps this is a feature, not a bug? This depends on your risk model.

gpg --gen-key

- What kind of key? DSA-EG, DSA, RSA
- Keysize from 256 - 4096 bits in length
- Expiration timeframe
- Identity strings: email, real name, comment
- Passphrase for the private key.

You can change your passphrase at any time with **gpg --edit-key**

/dev/random
and the Yarrow plant

od /dev/random

In the good old days...

```
FreeBSD, pre Darwin

vi /etc/rc.conf
```

insert this:

```
rand_irqs="1 5 10 14"
```

then run:

```
rndc-confgen
```

see http://people.freebsd.org/~dougb/randomness.html

iChing

Yarrow-160:
Notes on the Design and Analysis of the Yarrow Cryptographic Pseudorandom Number Generator

J. Kelsey, B. Schneier, and N. Ferguson

Sixth Annual Workshop on Selected Areas in Cryptography, Springer Verlag, August 1999.

ABSTRACT: We describe the design of Yarrow, a family of cryptographic pseudo-random number generators (PRNG). We describe the concept of a PRNG as a separate cryptographic primitive, and the design principles used to develop Yarrow. We then discuss the ways that PRNGs can fail in practice, which motivates our discussion of the components of Yarrow and how they make Yarrow secure. Next, we define a specific instance of a PRNG in the Yarrow family that makes use of available technology today. We conclude with a brief listing of open questions and intended improvements in future releases.

http://www.schneier.com/paper-yarrow.ps.gz

The Rest of the Story...

What Size Key?

Recommended Key Length	Year
1280 bits	1995
1280 bits	2000
1536 bits	2005
1536 bits	2010
2048 bits	2015

B.Schneier, "Applied Cryptography, Second Edition", Wiley, 1996.

Key Escrow / Backup

This slide may be the most important slide of this entire presentation.

- **gpg -K** to list private keys.
- **gpg -a --export 5A067115 | lpr**

Cut the sheet of paper in half and mail to two friends. Or put a copy in a safe deposit box. Whatever you do, please DO back your key up or you will aggregate the risk over time of losing ALL of your encrypted data.

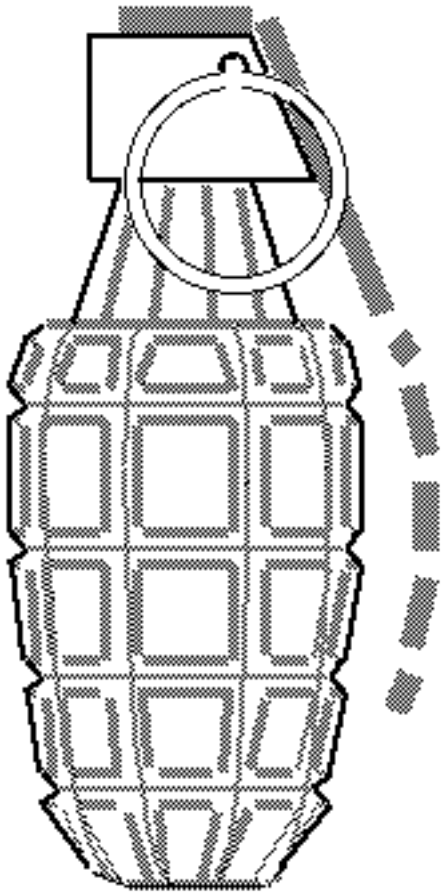
Risk Aggregation

How is a key-pair like a hand-grenade?

- Two parts
- No aiming
- Hard to use safely

How are they not alike?

- With a grenade, you throw the dangerous part away. . .



Common Sense Approach to Risk Management of GPG

- Be aware that the longer you use a key, the more a compromise will hurt.
- Is risk aggregation inevitable? If you don't plan around it, yes!
- One year public key expiration is reasonable.
- Be prepared to have your key change (revocation how-to's are readily available).

gpg --edit-key email@address.com

Command> showpref

pub 1024D/5A067115 created: 2005-07-12 expires: never
usage: CS trust: ultimate validity: ultimate

[ultimate] (1). Tech Support <support@my-domain-name.com>

Cipher: AES256, AES192, AES, CAST5, 3DES
Digest: SHA1, RIPEMD160
Compression: ZLIB, ZIP, Uncompressed
Features: MDC, Keyserver no-modify

Which Algorithm to Use?

- Read more about algorithms in PGP DH vs. RSA FAQ at
- www.hertreg.ac.uk/ss/pgp-faq.html
- See Bruce Schnier's useful overview in the text, *Applied Cryptography*.
- Summary: DSA/ ElGamal is not patented and is widely used. DSA/EG is a widely recommended encryption algorithm. Patent expiration on RSA may level the playing field somewhat. Want details?

DSA-ElGamal versus RSA according to Sam Simpson

In summary, DH PGP keys can be considered stronger than PGP RSA keys for the following reasons (I have removed references from this section for clarity - refer to previous sections of the document for justification of these claims):

1. DH & DSS offer slightly more security per bit than RSA. This is likely to remain the case. Conversely, RSA offers more security per clock cycle than DH/DSS.

2. The hash used by RSA keys is badly flawed. Remember Schneier's comment "I am wary of using MD5". DH keys use SHA-1 (or RIPEMD). Sure, it's possible (with new versions of PGP) to select the hash function used with RSA keys, but these signatures are incompatible with standard v2.x versions of PGP and also continue to suffer from key ID attacks (described next). RSA keys always use MD5 to create key fingerprints [Bar99b].

3. RSA keys can be created with an arbitrary Key ID or fingerprint - which can thus be used to spoof keyservers. This is just not possible with DH keys.

4. DH keys offer a choice of multiple ciphers. If one cipher is broken then not all DH keys will be useless. If IDEA is broken then all RSA keys are useless.

5. DH keys consist of two components: signature keys and encryption keys. Breaking (or being forced to divulge) one of the keys doesn't break the other. Breaking an RSA key allows an adversary to perform both decryption and signing.

6. DH keys can have multiple decryption subkeys. In this way a master (signature) key can be kept for many years while multiple decryption subkeys can be used to minimise the damage due to a "major break" in one of the encryption keys. Again, this feature is not available with RSA PGP keys.

7. Theoretically, DH keys appear stronger than RSA keys. One notes that some instances of the DHP are provably equivalent to the underlying hard mathematical problem (DLP). No such equivalence has been shown for RSA, in fact some instances of the RSAP are provably not equivalent to the underlying hard problem (IFP).

PGP Keyservers

- Useful for growing contact lists.
- Just like Hotel California: once you're in some of them (MIT?), you can never leave.
- Visualization of trust networks is nifty.
- Google for "GPG keyserver" for more.
- GPG Keychain Access provides interface.

Affiliated Pieces

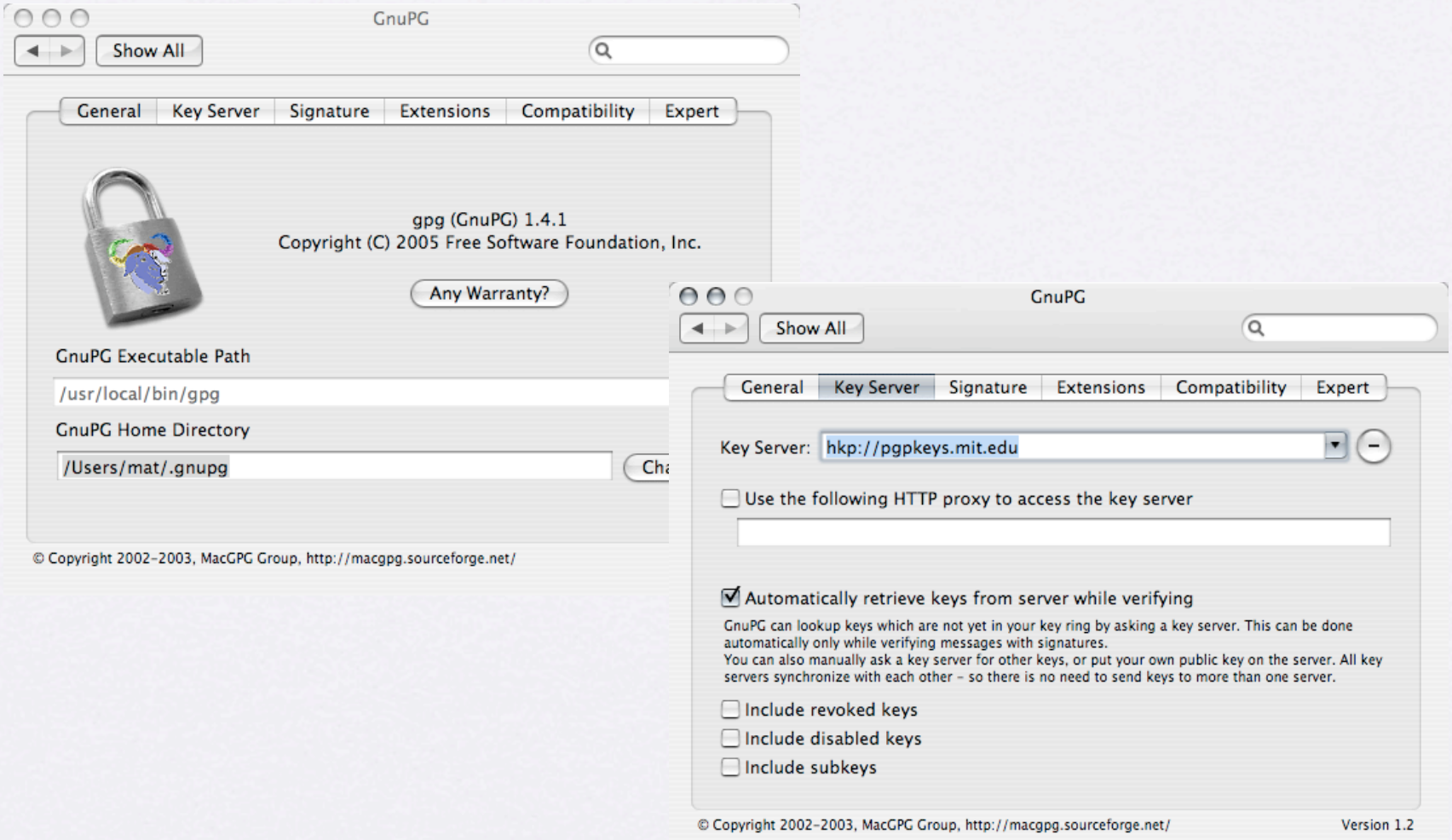
In addition to MacGPG (the main GPG software package) and GPGMail (the plugin-in for Mail.app) there are other pieces that you might want to know about...

1. GPGPreferences
2. GPG Keychain Access
3. GPGFileTool
4. GPGDropThing
5. ABKey

GnuPG Preferences Pane

- Provides interface for editing ~/gpg.conf
- Is included with GPG Keychain Access
- Contains list of popular key servers.
- Interface for TIGER and IDEA extensions.

GnuPG Prefs


GnuPG
Show All
General
Key Server
Signature
Extensions
Compatibility
Expert
gpg (GnuPG) 1.4.1
Copyright (C) 2005 Free Software Foundation, Inc.
Any Warranty?
GnuPG Executable Path
/usr/local/bin/gpg
GnuPG Home Directory
/Users/mat/.gnupg
© Copyright 2002-2003, MacGPG Group, http://macgpg.sourceforge.net/
GnuPG
Show All
General
Key Server
Signature
Extensions
Compatibility
Expert
Key Server:
hkp://pgpkeys.mit.edu
Use the following HTTP proxy to access the key server
Automatically retrieve keys from server while verifying
GnuPG can lookup keys which are not yet in your key ring by asking a key server. This can be done automatically only while verifying messages with signatures.
You can also manually ask a key server for other keys, or put your own public key on the server. All key servers synchronize with each other - so there is no need to send keys to more than one server.
Include revoked keys
Include disabled keys
Include subkeys
© Copyright 2002-2003, MacGPG Group, http://macgpg.sourceforge.net/
Version 1.2

GPGFileTool

- Has nice documentation and FAQ
- Signs
- Encrypts
- Clearsigns for files
- Decrypts
- Verifies signatures
- Useful for batch operation on files.

ABKey

In a nutshell? Address Book integration of GPG key management.

Although Apple's Address Book provides central management of your contacts' details, it was previously necessary to use different applications to manage their GPG keys. The ABKey plugin enhances Address Book by allowing you to see which of your contacts have current, revoked and expired keys as well as providing more detailed information on request.

Furthermore, the plugin works together with newer versions of GPGMail to provide per-contact control over signing and encryption preferences. You can state for a contact, for instance, to always encrypt or never sign or encrypt.

www.far-blue.co.uk/projects/keymanager.html

Darwinports Tools

- Complete set of secure email daemons for MacOSX server (courier, uw-imap, dovecot, etc.)
- See mail.darwinports.com
- Search for “email” or “gpg” to see a long list of gpg-affiliated programs.
- Some Darwinports developers are signing their portfiles and software with GPG keys.

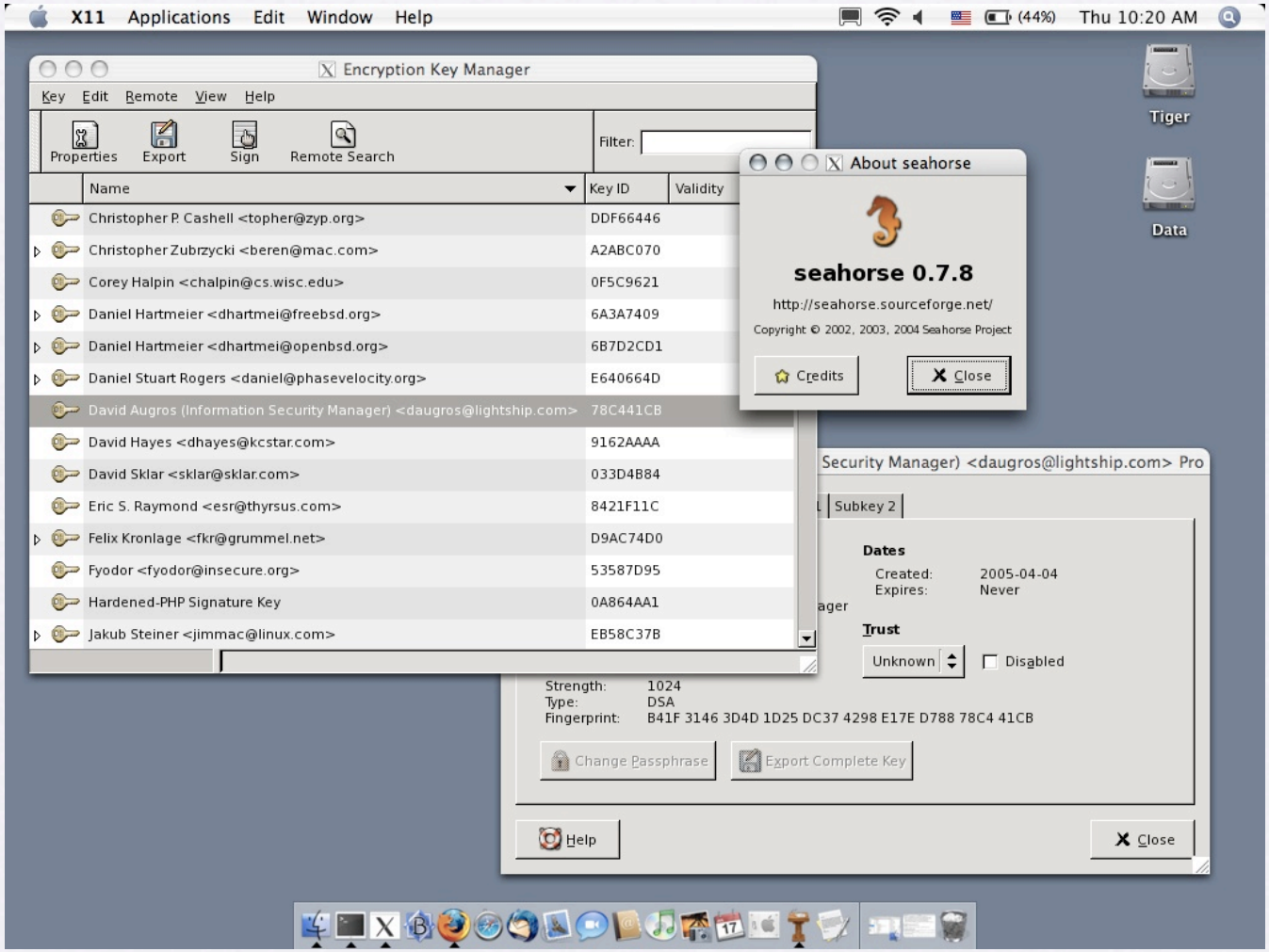
GNOME Project's Seahorse

GUI tool for Linux/BSD users.

Now Mac users can say "We have that too"

Seahorse is a GNOME application for managing PGP keys. It also integrates with nautilus, gedit and other places for encryption, decryption and other operations. Seahorse is based on GPG and GPGME and works under XDarwin or Apple's X11 environment. Best installed with Fink or Darwinports.

Seahorse in Action



Keysigning Concepts

Fingerprint versus keyid

Example

email: mat@phpconsulting.com

key: 9E3CD47E

fingerprint: 35FC 7ECC 1F67 4525 FE96
A0C3 FB3B B58A 9E3C D47E

Four Steps to Keysigning

1. Get a copy of the other key.

   Public keyservers make this fairly easy to do. However, this might be a direct copy via email or a USB keyfob or CDROM.

2. Verify the key, for example, by comparing the fingerprint. Do you trust the emailed copy of the fingerprint? Good to have this on your business card.

3. Sign the key with your own. This will likely require entering the password for your private key.

4. Give the signed key back, or upload to the keyserver.

Four Steps to Keysigning

1. `gpg --keyserver <keyserver> --recv-keys <Key_ID>`
2. `gpg --fingerprint <Key_ID>`

Verification occurs prior to assertion.

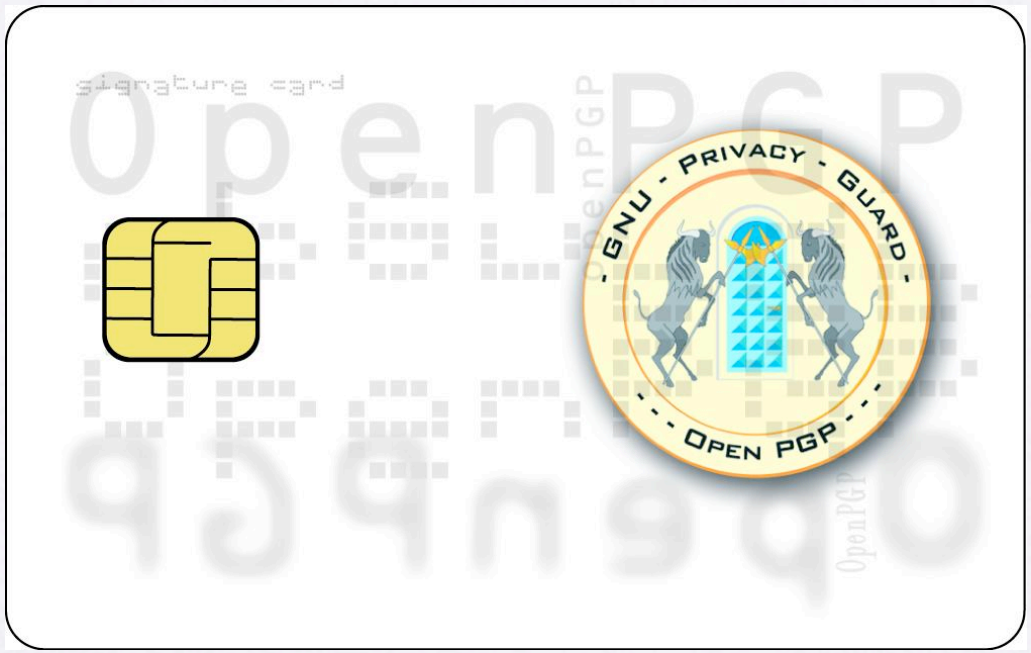
3. `gpg --default-key <Your_preferred_key> --sign-key <Key_ID>`
4. `gpg --keyserver <keyserver> --send-key <Key_ID>`

Further Directions, GPG Smart Cards

GPG has support for GPG smart cards.

addcardkey add a key to a smartcard
keytocard move a key to a smartcard
bkuptocard copies key to a smartcard

AmericanExpress has USB smartcard readers for low cost, check eBay. A German company makes cards with your GPG key on them: see www.g10code.de. Card support seems to be largely experimental.



OpenPGP Smart Card

PPC CARD·SYSTEMS powered by g10code

www.ppc-card.de

www.g10code.de

- Software available under the GNU GPL
- Compatible with OpenPGP standard
- Compatible with ISO 7816-4 and -8
- RSA with 1024 bit
- Key generation on card (<20 seconds)
- Hardware Random Number Generator (FIPS 140-1)
- Key import functionality

Software download: www.gnupg.org

eMail Certificate Notarization and Key Signing

- Take your pick:
- Thawte notarization requires two forms of ID. This provides a free email x.509 personal certificate for the Thawte web of trust.
- If you'd like, I will be happy to sign your GPG key.

Whitfield Diffie quotation

"*In writing PGP, Phil Zimmermann did something for cryptography that no technical paper could do: he gave people who were concerned with privacy but were not cryptographers (and not necessarily even programmers) a tool they could use to protect their communications*".

GPG on Mac OSX brings these tools further into the popular domain. Mac users who are used to point and click programs can now set up and use robust cryptography. Consider contributing to the Free Software Foundation to support this software.

Available on Request
< mat@phpconsulting.com >

Get your very own email cryptography kit.

I have prepared a disk image for you to take with you if you are interested. The software also fits on a CDR or 64MB+ USB key.

The contents of this presentation and all associated open source software are contained on it, including: MacGPG, GPGMail, ABKey, GPG source code, iTrustGPG, the GNU Privacy Guide, and other helpful documentation.

Thanks for Coming.

Remember to escrow your keys!

I hope you enjoyed listening to this presentation as much as I've enjoyed giving it.

See you again next year?

Mat Caughron, CISSP
mat@phpconsulting.com
(402) 968-1332