GPG: The Final Frontier

These are the voyages of the GPG Encryptiprise…
David Liddle and Nick Starke

Disclaimer

Please move all private keys to a portable USB thumb drive and deliver to David Liddle.

Who we are

David Liddle -

- Currently studying to complete BMIS: Networking Degree
- Looking for entry level work within the IT field

Nick Starke -

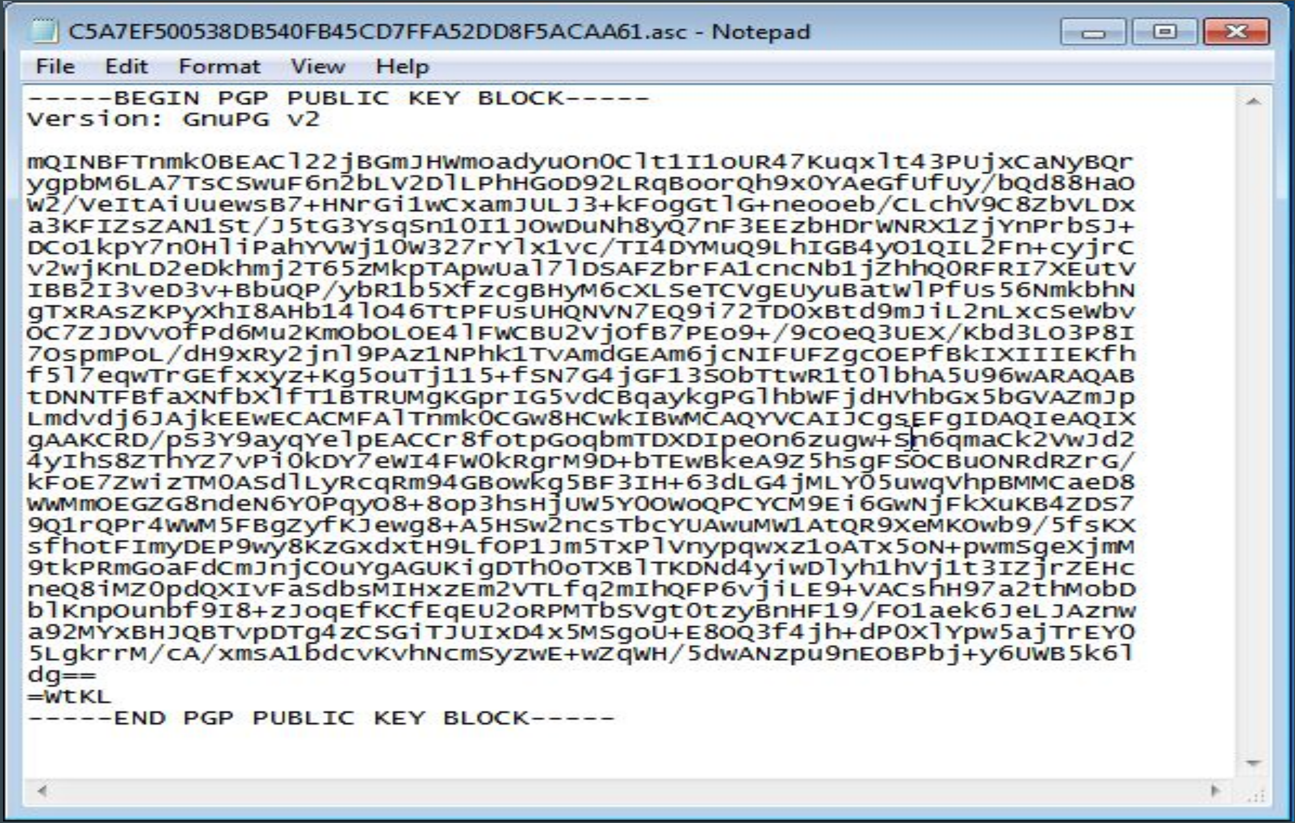
- A guy who does the things with the stuff, IE security researcher from Des Moines.

What is GPG

GPG (Gnu Privacy Guard/OpenPGP) is a continuation of Phil Zimmerman's PGP which he wrote in 1991. PGP (Pretty Good Privacy) uses a compound method of compression, symmetric and asymmetric key encryption, and encrypted signatures verifying the author's and recipient's identity and message encryption; it is mainly applied to plain text email for transmission over the internet. PGP is now available from Symantec and OpenPGP.

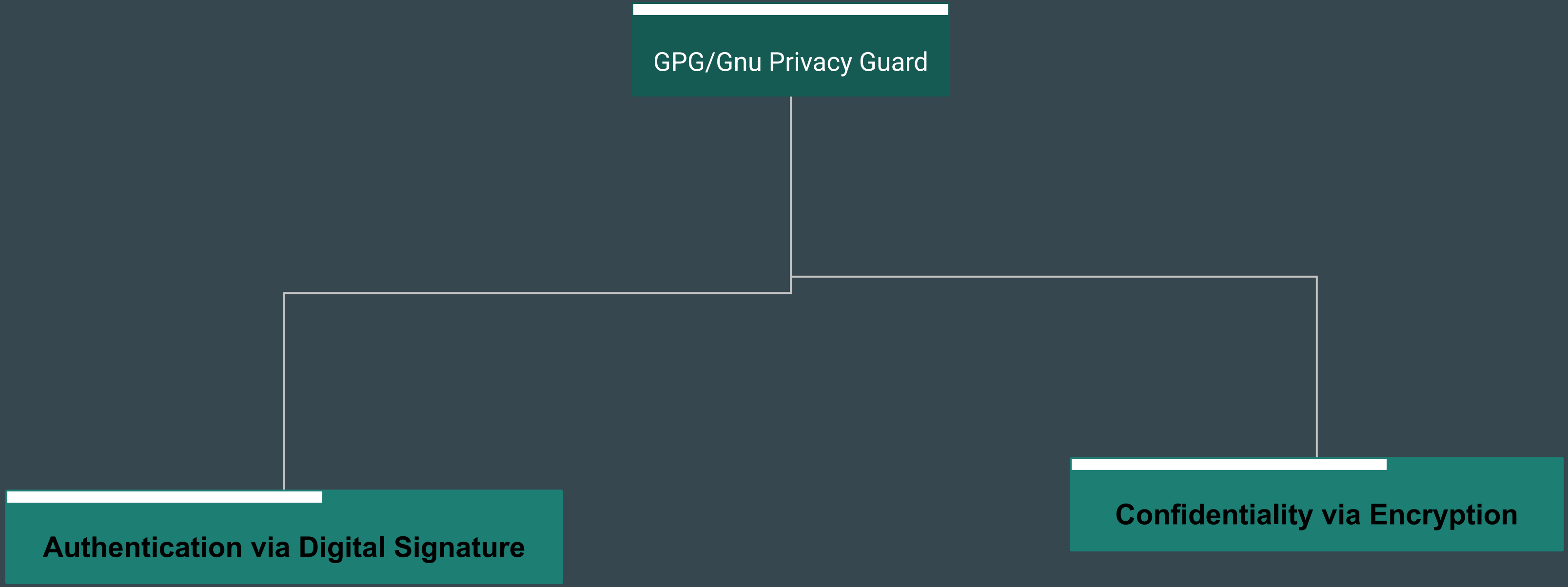
Historically, PGP has been quite controversial, initially inciting the negative attention of the Government (who cited it as illegal to distribute outside the US...and not happy they couldn't initially crack it), and the authors of the RSA algorithm and MIT (who cited unlawful use of their algorithm to encrypt without a license).

HOW IT WORKS!!!


C5A7EF500538DB540FB45CD7FFA52DD8F5ACAA61.asc - Notepad

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2

mQINBFTnmk0BEACl22jBGmJHWmoadyuOn0Clt1I1oUR47Kuqxlt43PUjxCaNyBQr
ygpbM6LA7TsCSwuF6n2bLV2DlLPhHGoD92LRqBoorQh9x0YAeGfUfUy/bQd88HaO
W2/VeItAiUuewsB7+HNrGi1wCxamJULJ3+kFogGtlG+neooeb/CLchV9C8ZbVLDx
a3KFIZsZAN1St/J5tG3YsqSn10I1JOwDuNh8yQ7nF3EEzbHDrWNRX1ZjYnPrbSJ+
DCo1kpY7n0HliPahYVWj10W327rYlx1vc/TI4DYMuQ9LhIGB4yO1QIL2Fn+cyjrC
v2wjKnLD2eDkhmj2T65zMkpTApwUal7lDSAFZbrFA1cncNb1jZhhQ0RFRI7XEutV
IBB2I3veD3v+BbuQP/ybR1b5XfzcgBHyM6cXLSeTCVgEUyuBatWlPfUs56NmkbhN
gTxRASZKPyXhI8AHb14lO46TtPFUSUHQNVN7EQ9i72TD0xBtd9mJiL2nLxcSeWbv
OC7ZJDVvOfPd6Mu2KmObOLOE4lFWCBU2VjOfB7PEo9+/9cOeQ3UEX/Kbd3LO3P8I
7OspmPoL/dH9xRy2jnl9PAz1NPhk1TvAmdGEAm6jcNIFUFZgcOEPfBkIXIIIEKfh
f5l7eqwTrGEfxxyz+Kg5ouTj115+fSN7G4jGF13SObTtwR1t0lbhA5U96wARAQAB
tDNNTFBfaXNfbXlfT1BTRUMgKGprIG5vdCBqaykgPGlhbWFjdHVhbGx5bGVAZmJp
Lmdvdj6JAjkEEwECACMFAlTnmk0CGw8HCwkIBwMCAQYVCAIJCgsEFgIDAQIeAQIX
gAAKCRD/pS3Y9ayqYelpEACCr8fotpGoqbmTDXDIpeOn6zugw+Sn6qmaCk2VwJd2
4yIhS8ZThYZ7vPi0kDY7eWI4FW0kRgrM9D+bTEwBkeA9Z5hsgFSOCBuONRdRZrG/
kFoE7ZwizTM0ASdlLyRcqRm94GBowkg5BF3IH+63dLG4jMLY05uwqVhpBMMCaeD8
WWMmOEGZG8ndeN6Y0PqyO8+8op3hsHjUW5YOOWoQPCYCM9Ei6GwNjFkXuKB4ZDS7
9Q1rQPr4WWM5FBgZyfKJewg8+A5HSw2ncsTbcYUAwuMW1AtQR9XeMKOwb9/5fsKX
sfhotFImyDEP9wy8KzGxdxtH9LfOP1Jm5TxPlVnypqwxz1oATx5oN+pwmSgeXjmM
9tkPRmGoaFdCmJnjCOuYgAGUKigDTh0oTXBlTKDNd4yiwDlyh1hVj1t3IZjrZEHc
neQ8iMZ0pdQXIvFaSdbsMIHxzEm2VTLfq2mIhQFP6vjiLE9+VACshH97a2thMobD
blKnpOunbf9I8+zJoqEfKCfEqEU2oRPMTbSVgt0tzyBnHF19/FO1aek6JeLJAznw
a92MYxBHJQBTvpDTg4zCSGiTJUIxD4x5MSgoU+E8OQ3f4jh+dP0XlYpw5ajTrEY0
5LgkrrM/cA/xmsA1bdcvKvhNcmSyzwE+wZqWH/5dwANzpu9nEOBPbj+y6UWB5k6l
dg==
=WtKL
-----END PGP PUBLIC KEY BLOCK-----
```

Two Main PGP/GPG Functions


GPG/Gnu Privacy Guard
Authentication via Digital Signature
Confidentiality via Encryption

Confidentiality via Encryption

- 1. The sender creates a message.
- 2. The sending OpenPGP generates a random number to be used as a session key for this message only.
- 3. The session key is encrypted using each recipient's public key. These "encrypted session keys" start the message.
- 4. The sending OpenPGP encrypts the message using the session key,
  which forms the remainder of the message. Note that the message
  is also usually compressed.
- 5. The receiving OpenPGP decrypts the session key using the
  recipient's private key.
- 6. The receiving OpenPGP decrypts the message using the session key.
  If the message was compressed, it will be decompressed.

Authentication via Digital Signature Steps

- 1. The sender creates a message.
- 2. The sending software generates a hash code of the message.
- 3. The sending software generates a signature from the hash code
- using the sender's private key.
- 4. The binary signature is attached to the message.
- 5. The receiving software keeps a copy of the message signature.
- 6. The receiving software generates a new hash code for the received message and verifies it using the message's signature. If the verification is successful, the message is accepted as authentic.

What are the use-cases for GPG?

- When you don’t want anyone in the middle to know what you’re sending. Examples:
    - GMAIL

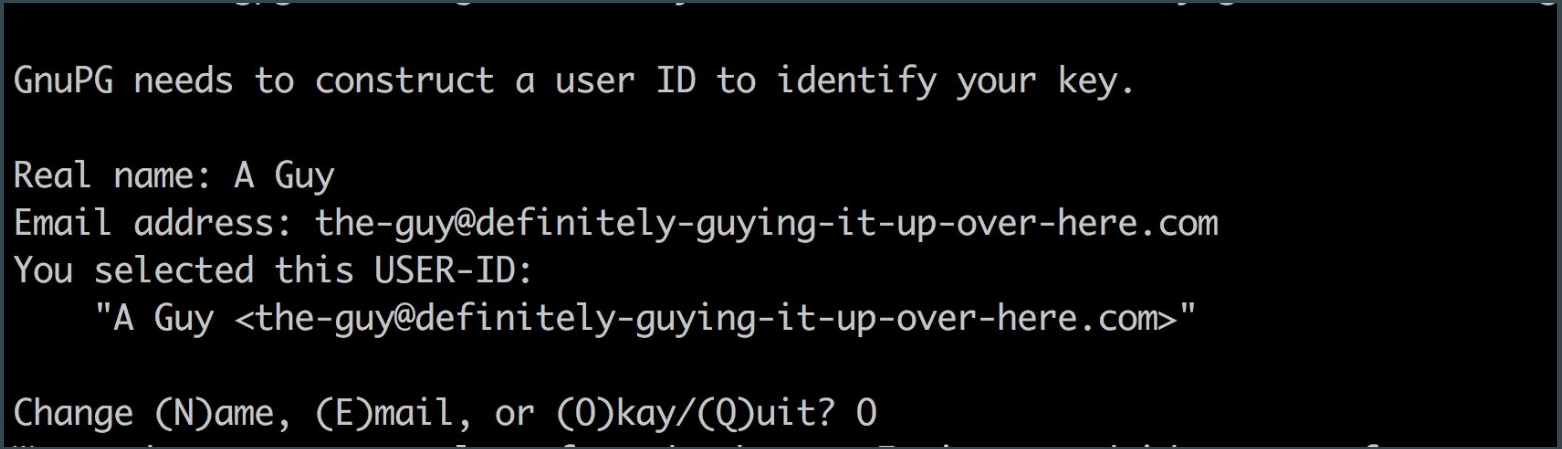
How do I use GPG?

This is how I learned GPG:

http://irtfweb.ifa.hawaii.edu/~lockhart/gpg/

```
$ gpg --gen-key
```

This will create your public/private key pair. You will be asked a bunch of questions, you should answer all of them.

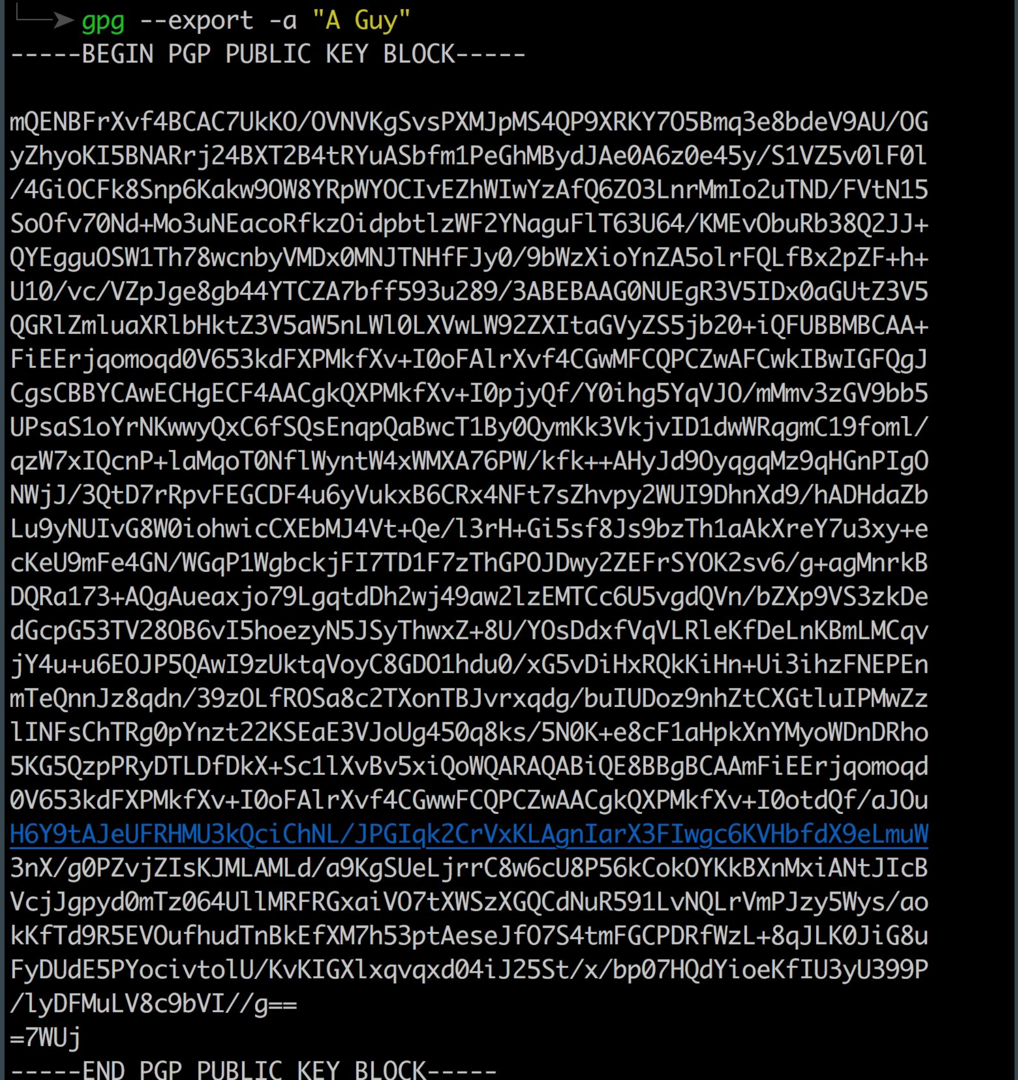


```
GnuPG needs to construct a user ID to identify your key.

Real name: A Guy
Email address: the-guy@definitely-guying-it-up-over-here.com
You selected this USER-ID:
    "A Guy <the-guy@definitely-guying-it-up-over-here.com>"

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

GPG Export Public Key

```
$ gpg --export -a “A Guy” > public.key
```

Now you can send “public.key” to your friends!

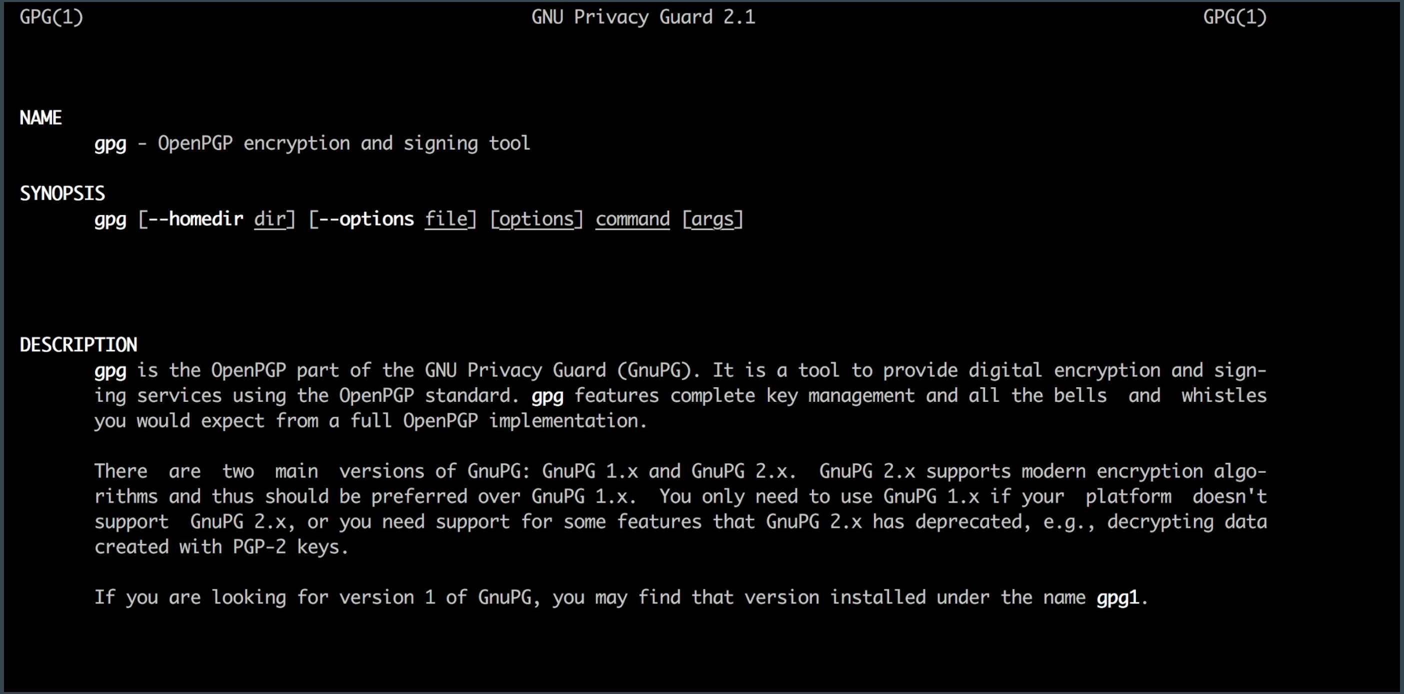


```
└──➤ gpg --export -a "A Guy"
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQENBFrXvf4BCAC7UkKO/OVNVKgSvsPXMJpMS4QP9XRKY7O5Bmq3e8bdeV9AU/OG
yZhyoKI5BNARrj24BXT2B4tRYuASbfm1PeGhMBydJAe0A6z0e45y/S1VZ5v0lF0l
/4GiOCFk8Snp6Kakw9OW8YRpWYOCIvEZhWIwYzAfQ6ZO3LnrMmIo2uTND/FVtN15
SoOfv70Nd+Mo3uNEacoRfkzOidpbtlzWF2YNaguFlT63U64/KMEvObuRb38Q2JJ+
QYEgguOSW1Th78wcnbyVMDx0MNJTNHfFJy0/9bWzXioYnZA5olrFQLfBx2pZF+h+
U10/vc/VZpJge8gb44YTCZA7bff593u289/3ABEBAAG0NUEgR3V5IDx0aGUtZ3V5
QGRlZmluaXRlbHktZ3V5aW5nLWl0LXVwLW92ZXItaGVyZS5jb20+iQFUBBMBCAA+
FiEErjqomoqd0V653kdFXPMkfXv+I0oFAlrXvf4CGwMFCQPCZwAFCwkIBwIGFQgJ
CgsCBBYCAwECHgECF4AACgkQXPMkfXv+I0pjyQf/Y0ihg5YqVJO/mMmv3zGV9bb5
UPsaS1oYrNKwwyQxC6fSQsEnqpQaBwcT1By0QymKk3VkjvID1dwWRqgmC19foml/
qzW7xIQcnP+laMqoT0NflWyntW4xWMXA76PW/kfk++AHyJd9OyqgqMz9qHGnPIgO
NWjJ/3QtD7rRpvFEGCDF4u6yVukxB6CRx4NFt7sZhvpy2WUI9DhnXd9/hADHdaZb
Lu9yNUIvG8W0iohwicCXEbMJ4Vt+Qe/l3rH+Gi5sf8Js9bzTh1aAkXreY7u3xy+e
cKeU9mFe4GN/WGqP1WgbckjFI7TD1F7zThGPOJDwy2ZEFrSYOK2sv6/g+agMnrkB
DQRa173+AQgAueaxjo79LgqtdDh2wj49aw2lzEMTCc6U5vgdQVn/bZXp9VS3zkDe
dGcpG53TV28OB6vI5hoezyN5JSyThwxZ+8U/YOsDdxfVqVLRleKfDeLnKBmLMCqv
jY4u+u6EOJP5QAwI9zUktqVoyC8GDO1hdu0/xG5vDiHxRQkKiHn+Ui3ihzFNEPEn
mTeQnnJz8qdn/39zOLfROSa8c2TXonTBJvrxqdg/buIUDoz9nhZtCXGtluIPMwZz
lINFsChTRg0pYnzt22KSEaE3VJoUg450q8ks/5N0K+e8cF1aHpkXnYMyoWDnDRho
5KG5QzpPRyDTLDfDkX+Sc1lXvBv5xiQoWQARAQABiQE8BBgBCAAmFiEErjqomoqd
0V653kdFXPMkfXv+I0oFAlrXvf4CGwwFCQPCZwAACgkQXPMkfXv+I0otdQf/aJOu
H6Y9tAJeUFRHMU3kQciChNL/JPGIqk2CrVxKLAgnIarX3FIwgc6KVHbfdX9eLmuW
3nX/g0PZvjZIsKJMLAMLd/a9KgSUeLjrrC8w6cU8P56kCokOYKkBXnMxiANtJIcB
VcjJgpyd0mTz064UllMRFRGxaiVO7tXWSzXGQCdNuR591LvNQLrVmPJzy5Wys/ao
kKfTd9R5EVOufhudTnBkEfXM7h53ptAeseJfO7S4tmFGCPDRfWzL+8qJLK0JiG8u
FyDUdE5PYocivtolU/KvKIGXlxqvqxd04iJ25St/x/bp07HQdYioeKfIU3yU399P
/lyDFMuLV8c9bVI//g==
=7WUj
-----END PGP PUBLIC KEY BLOCK-----
```

Encrypting an email with someone else’s public key

```
gpg -e -u “A Guy” -r “Recipient Name” the_file_with_the_secrets.txt
```

For the rest of the commands:



```
GPG(1)                        GNU Privacy Guard 2.1                        GPG(1)



NAME
       gpg - OpenPGP encryption and signing tool

SYNOPSIS
       gpg [--homedir dir] [--options file] [options] command [args]




DESCRIPTION
       gpg is the OpenPGP part of the GNU Privacy Guard (GnuPG). It is a tool to provide digital encryption and sign-
       ing services using the OpenPGP standard. gpg features complete key management and all the bells  and  whistles
       you would expect from a full OpenPGP implementation.

       There  are  two  main  versions of GnuPG: GnuPG 1.x and GnuPG 2.x.  GnuPG 2.x supports modern encryption algo-
       rithms and thus should be preferred over GnuPG 1.x.  You only need to use GnuPG 1.x if your  platform  doesn't
       support  GnuPG 2.x, or you need support for some features that GnuPG 2.x has deprecated, e.g., decrypting data
       created with PGP-2 keys.

       If you are looking for version 1 of GnuPG, you may find that version installed under the name gpg1.
```

GPG Cryptography

- Compression
- Public and Private Key encryption
- Radix-64/ASCII Armor

References

4880 OpenPGP Message Format. D. Shaw, H. Finney, J. Callas, L. Donnerhacke, R. Thayer. November 2007. (Also RFC5581, RFC6637, RFC2015, RFC3156) (Status: Internet Standards Track Protocol) (DOI: 10.17487/RFC4880)

Zimmerman, Phil R. (1999). *Why I Wrote PGP.* Retrieved from https://www.philzimmermann.com/EN/essays/WhyIWrotePGP.html

Back, Adam. *PGP TImeline.* Retrived from http://www.cypherspace.org/adam/timeline/

Questions?

Thanks for attending our presentation!

Contact Details:

David Liddle:

Nick Starke: https://twitter.com/nstarke | https://gist.github.com/nstarke