paranoid.EMAIL 

Using GnuGP to secure your communications

Joshua I. James
Joshua@cybercrimetech.com
GPG: 0x9D635AE7606B15C4

Have you ever lied?

- **If someone lied to you, is it your fault for being wrong?**
- **If someone lies to a judge, is it the judges fault for being wrong?**
- **Humans make mistakes**
  - even Police / Prosecutors
- **Ultimate question: how can we trust?**

Overview

- **In this presentation we will quickly talk about:**
  - Signing and verifying data
  - Web of Trust
  - Encrypting email

Verifying (Data) Digital Evidence

- **If hashes match, data is exactly the same[1]**
  - Hashing (md5, sha1)
  - **Manual** documentation

```
joshua@Icarus ~/Documents/temp $ echo "Hash test" > hash.test
joshua@Icarus ~/Documents/temp $ md5sum hash.test
081e194ea05f91d75cf0ba0ace3328f8  hash.test
```

```
joshua@Icarus ~/Documents/temp $ cat hash.test
Hash test1
joshua@Icarus ~/Documents/temp $ md5sum hash.test
eb06719afd58a3cbac9cc9e79c0e30d7  hash.test
```

[1] See hash collision vulnerabilities: https://en.wikipedia.org/wiki/MD5#Collision_vulnerabilities

Verifying (Data) Digital Evidence

- **What is the problem with this method?**
- **What is missing?**
  - *Who?*
  - *When?*
  - ~~What~~

```
joshua@Icarus ~/Documents/temp $ cat hash.test
Hash test1
joshua@Icarus ~/Documents/temp $ md5sum hash.test
eb06719afd58a3cbac9cc9e79c0e30d7  hash.test
```

Keys and Signing

- **Instead of anonymously hashing the data, we can *sign* the data**
  - Signing the data produces a signature file
  - Similar to hashing, but includes information like *who* and *when*

```
joshua@Icarus ~/Documents/temp $ gpg --sign hash.test

You need a passphrase to unlock the secret key for
user: "Joshua James <joshua@cybercrimetech.com>"
2048-bit RSA key, ID 52D5E535, created 2013-01-09 (main key ID 606B15C4)
```

Signing data with GnuPG on Linux [https://www.gnupg.org/]

Keys and Signing

- **Now we can verify the data:**
  - Signature was made 2015/02/13
  - Signatory's Name + Email

```
joshua@Icarus ~/Documents/temp $ gpg --verify hash.test.sig
gpg: Signature made 2015년 02월 13일 (금)  using RSA key ID 52D5E535
gpg: Good signature from "Joshua James <joshua@cybercrimetech.com>"
gpg:                 aka "Joshua James <joshua.james@ucd.ie>"
gpg:                 aka "Joshua James <joshua.i.james@gmail.com>"
gpg:                 aka "Joshua I. James <joshua@paranoid.email>"
gpg:                 aka "Joshua I. James <joshua@2048.email>"
```

Keys and Signing

- **If the data changes, the verification will fail**

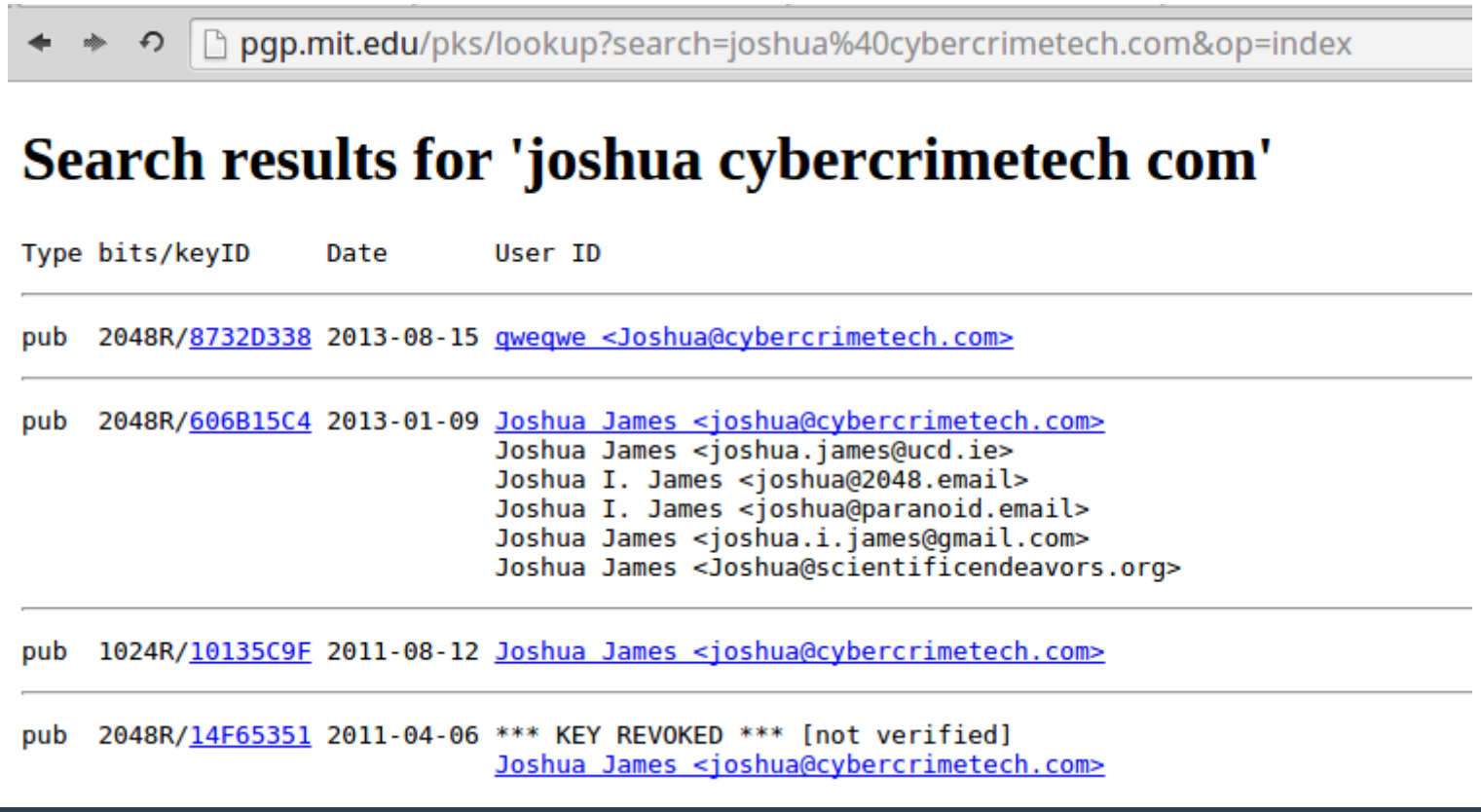
```
joshua@Icarus ~/Documents/temp $ gpg --verify hash.test.sig
gpg: Signature made 2015년 02월 13일 (금)  using RSA key ID 52D5E535
gpg: BAD signature from "Joshua James <joshua@cybercrimetech.com>"
```

Web of Trust (WoT)

- **So we know a signature is valid, how can we trust it?**
  - When creating your own signatures, you can choose your name and email address
    - Attacker could use **any** name and email address

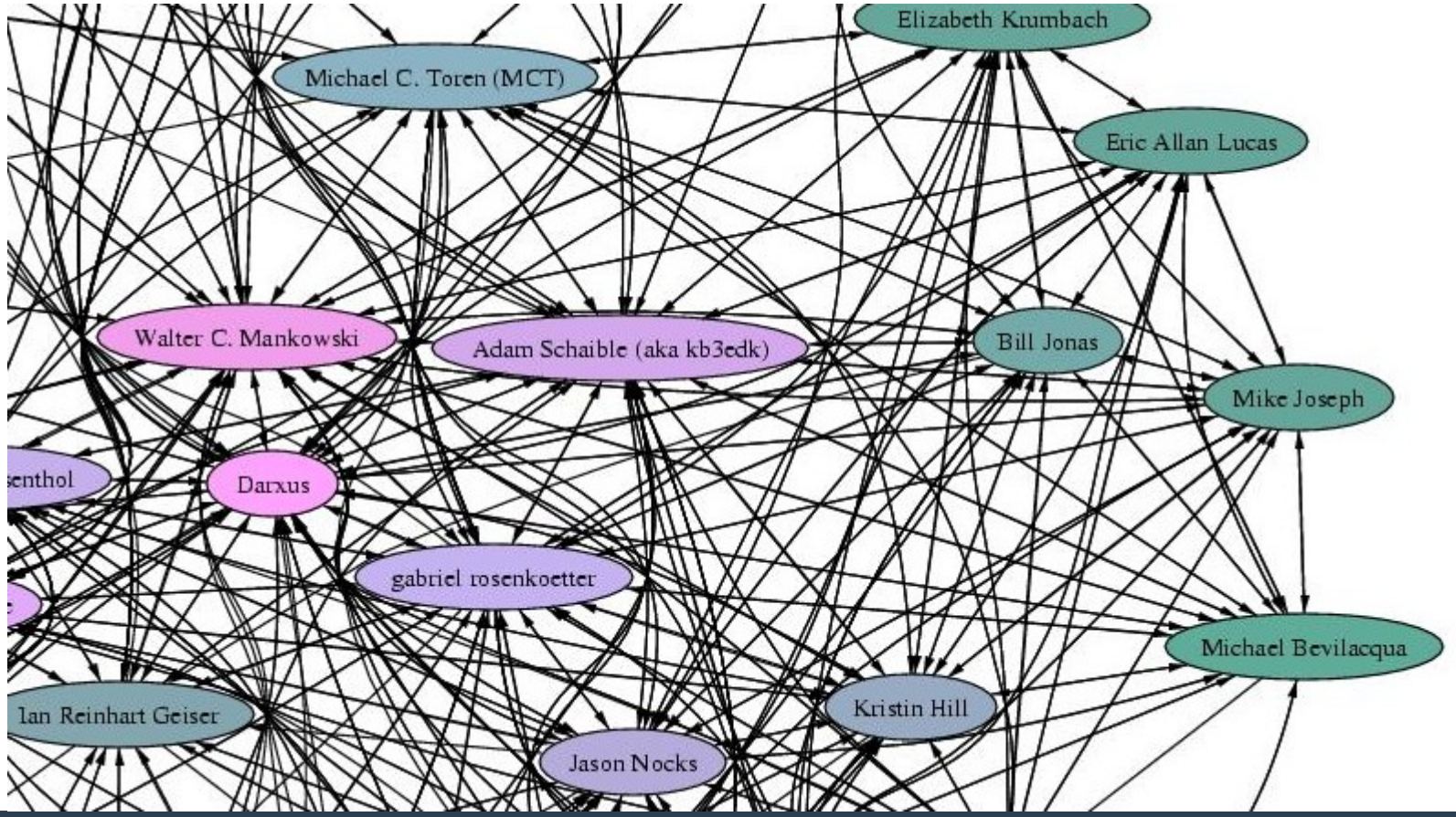
Web of Trust (WoT)

- **I created one of these keys**
  - Which one can you trust?



Web of Trust (WoT)

- **The Web of Trust is like a *verified* social network**



From http://www.phillylinux.org

Search results for '0x5d69c71e9ca55a17'

Other users who have signed (verified) this private key

```
pub  2048R/9CA55A17 2012-07-10

uid Adi Kamdar <adi@eff.org>
sig  sig3  9CA55A17 2012-07-10 __________ 2016-07-10 [selfsig]
sig  sig   9C7DD150 2012-07-10 __________ __________ Seth David Schoen <schoen@loyalty.org>
sig  sig3  99999697 2012-07-23 __________ __________ Micah Lee <micah.lee@theintercept.com>
sig  sig3  7D9A1527 2013-06-18 2017-06-18 __________ Kurt Opsahl (KurtEFF) <kurt@eff.org>
sig  sig   A3FDE45E 2013-09-18 __________ __________ Danny O'Brien <danny@spesh.com>
sig  sig   D3366755 2014-02-05 __________ __________ William Budington <bill@eff.org>
sig  sig2  7684381F 2014-02-05 __________ __________ Kelly Esguerra <kelly@eff.org>
sig  sig   8CA3A9E8 2014-02-05 __________ __________ Joseph Bonneau (Stanford CS email) <jbonneau@cs.stanford.edu>
sig  sig3  8BF8DD0B 2014-02-05 __________ __________ leez <leez@eff.org>
sig  sig2  F0AFE2CA 2014-02-05 __________ __________ Cooper Quintin <cooperq@garlic.is>
sig  sig3  CC5FFED2 2014-02-05 __________ __________ Andrew Crocker <andrew@eff.org>
sig  sig3  7B06DFFA 2014-02-05 __________ __________ Aaron Jue <aaron@eff.org>
sig  sig3  D82C109E 2014-02-05 __________ __________ Nicole Puller <nicole@eff.org>
sig  sig3  F4CEF0A5 2014-02-05 __________ __________ mark burdett <mfb@eff.org>
sig  sig3  9A367709 2014-02-05 __________ __________ Parker Higgins (EFF activism team) <parker@eff.org>
sig  sig   FAC78CF7 2014-02-05 __________ __________ Yan Zhu <zyan@mit.edu>
sig  sig2  16681C9C 2014-02-05 2018-02-05 __________ Richard Esguerra <richard@eff.org>
sig  sig3  8AD19194 2014-02-05 2018-02-05 __________ Hugh D'Andrade <hugh@eff.org>
sig  sig2  0BBAB306 2014-02-06 2018-02-06 __________ Nick Doty <npdoty@w3.org>
sig  sig3  80AF07D3 2014-02-06 __________ __________ Flamsmark <flamsmark@gmail.com>
     Policy URL: http://flamsmark.com/id.txt
sig  sig3  A947E771 2014-05-10 __________ __________ John Fresco <john.fresco@utah.edu>
sig  sig3  FE41ED26 2014-05-12 __________ __________ Lonnie Olson <lonnie@lonnieolson.com>
sig  sig3  6A539F02 2014-05-12 __________ __________ David Owen <dsowen@fugue88.ws>
sig  sig   3B76C63F 2014-05-12 __________ __________ Christopher Kelly <phoenix.ckelly@gmail.com>
sig  sig   A036D05A 2014-05-12 __________ __________ D. E. Evans <sinuhe@gnu.org>
sig  sig3  4D68D707 2014-05-13 __________ __________ Mitch Morby <mmorby1@gmail.com>
sig  sig   DA49B416 2014-05-13 __________ __________ Allen Mons <albmons@gmail.com>
sig  sig3  F7612611 2014-05-14 __________ __________ Gavin Howard <lfyre.lang@gmail.com>
sig  sig3  D82A65F8 2014-05-14 __________ __________ Trenton Anderson <trenton.anderson@iqinbox.com>
sig  sig3  0060DF7B 2014-05-15 __________ __________ Frostyfrog <frosty@frostyfrog.net>
sig  sig   22A3D89D 2014-05-20 __________ __________ Jake Peterson (acogdev) <cqjake@gmail.com>
sig  sig   774FC55A 2014-06-12 __________ __________ Jeremy Gillula <jeremy@eff.org>
sig  sig   2909338C 2014-08-10 __________ __________ Michael Englehorn <michael@englehorn.com>
sig  sig   ABBDD750 2014-08-11 __________ __________ Jamie Lynn O'Marr <greymaiden@gmail.com>
sig  sig   4BAF5D09 2014-08-12 __________ __________ Eric Swanson <eswanson@alloscomp.com>
sig  sig   ED873D23 2014-08-14 __________ __________ Alan Eliasen <eliasen@mindspring.com>
sig  sig2  A2198A7C 2014-08-16 __________ __________ Bruce Tindall <bruce.tindall@gmail.com>
sig  sig   D4217DB1 2014-08-18 __________ __________ Mark Smith <mark@halibut.com>
sig  sig3  A5C3B18D 2014-09-03 2018-09-03 __________ Justin Culbertson <jculberts@gmail.com>
sig  sig2  BC71BC44 2014-09-17 __________ __________ Duncan Townsend <duncant@mit.edu>
```

Web of Trust (WoT)

- **By creating a Web of Trust, people verify you are the owner of your public key (usually require a photo ID)**
- **When you and your key are (correctly) verified by a large number of people, the level of trust in that public key also increases**
- **By signing someone else's key you are putting your reputation at risk**
  - If you are a bad verifier, your key may be trusted less

Secure Communication (email)

- **Now we have:**
  - Public key (that other people trust – WoT)
  - Private key (that we keep secret)
- **How can we send a secure email?**

  1) I download your public key
  2) I use your **public** key to encrypt the email
  3) I send the email per usual
  4) You receive the email
  5) You use your **private** key to decrypt the email

Secure Communication (email)

- **Why would we want to encrypt email if we are not criminals?**
  - Many groups are interested in the contents of our emails
    - Hackers
    - Governments
    - Advertisers
    - Competitors
    - Ex girlfriends / boyfriends (New girlfriends / boyfriends?)

Secure Communication (email)

- **Most email is archived on the mail server**
  - Sitting in plain-text, waiting to be read by... someone
  - Easy to search by anyone with access
  - Easy to scan for marketing

Secure Communication (email)

- **For end-to-end encryption the sender and receiver both need their own public key and private key on the client**
- **Client side software:**
  - Thunderbird: **Enigmail** [www.enigmail.net]
  - Linux: **GnuPG** [package: gnupg]
  - OS X (Mail): **GPGTools** [gpgtools.org]
  - Windows: **GPG4Win** [www.gpg4win.org]

Secure Communication (email)

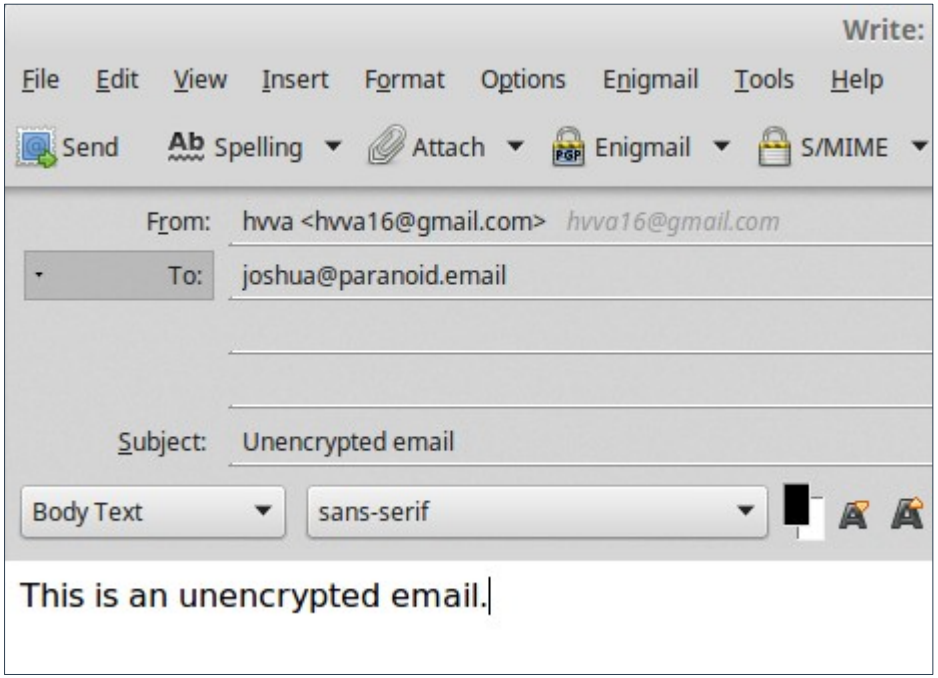
- **Problem: very few people do this**
  - I've had “please encrypt” in my email signature for over 4 years
  - I've received 3 encrypted messages...
  - The most sensitive information was never encrypted or properly secured

Secure Communication (email)

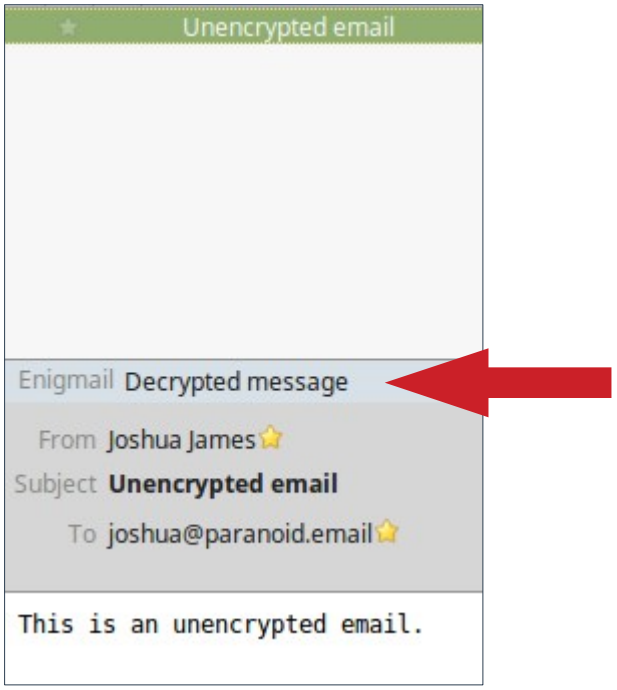
- **Solution: http://paranoid.EMAIL**
- **The email server will encrypt unencrypted messages automatically**
  - If someone sends me bank account information in plain-text, once it reaches the paranoid.EMAIL server, the message will be encrypted with my public key
  - Only my private key can decrypt the message

Secure Communication (email)

- **Sending an unencrypted email to a paranoid.EMAIL address**



Composing an email, and NOT using encryption



Message received has been encrypted by the server

Secure Communication (email)

```
Message-ID: <54DD932E.1000103@gmail.com>
Date: Fri, 13 Feb 2015 15:01:18 +0900
From: hvva <hvva16@gmail.com>
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:31.0) Gecko/20100101 Thunderbird/31.4.0
Mime-Version: 1 (Paranoid.email GnuPG mailer)
To: joshua@paranoid.email
Subject: Unencrypted email
Content-Transfer-Encoding: 7bit
X-OPENPGPKEY: Encrypted to key(s): 400E626BE05D2235F41108A89D635AE7606B15C4
Content-Transfer-Encoding: 7bit
X-Mailer: Paranoid.Email V0.2

-----BEGIN PGP MESSAGE-----
Version: GnuPG v1

hQEMA999TJtS1eU1AQf/V2MEEEEfHer6tWT3XQTcGgMhz9NPRIYIxBE3Kbd1Qv7/
YqVcQNec2qCwdfESWaerOEhqcktv1RFemG+e9eA71emuJ1sT5lKhQeS4cAp9YIfu
kYHknStHRQWhtTCKfdjJ+Mk/oSyuRWzT41fKa7CoFlROaslZVRP4g5j5zDGDITFA
8dvchf7kPfdGue46NNHeCCaiCcwZSLZjD7qA2SAVVf4xyXZIl2LThSFVVTh5TixD
MmETCNKaOvqwKj3sL2IcSMdKmB3oNTmqjn/v7TnvwSMEufEgDZVGRpQrLOUDfqsp
fhBuwxKn3svPhpbQsSoGao9Sr+D1nLRmbhVMi1PQ29KuAfbonxdiuV8jESIZZOIX
Y9LpHELFMi+jQjS+XK9eduGMwPjQB+19SWy9dziQuE5l+r5Ru89uFUwGyoMQJ6IS
wR8LqAmYXkqw7Ulb9OJ5vggA2FGZ4VkEtZrAg32Pu5TJRm01XY0Sj6JChSMzS1ZG
KFZKluLoLAYXZdSwwqzmIS2seoIuwX53OZ3q8WIXpnSFyNtEGvMnK4Y0EtsjHQIj
D+Y68OkPVWoxYYYaS5DK
=gPY0
-----END PGP MESSAGE-----
```

Received message

Conclusions

- **Securing communications during transmission and storage is an important and challenging area**
- **More easy-to-use tools need to be developed**
- **Please use signing to verify data and programs you release**
- **Please use gpg for communications, and advertise that you use it**
- **Please use http://paranoid.EMAIL**
  - Currently in **Beta**

Questions?

Joshua I. James
Joshua@cybercrimetech.com
GPG: 0x9D635AE7606B15C4