02. Usable Encryption

Blase Ur and Mainack Mondal
March 28th, 2018
CMSC 23210 / 33210

THE UNIVERSITY OF CHICAGO

Security, Usability, & Privacy Education & Research

This World of Ours

Encryption: A crash course

Encryption basics

- Putting information in code so that unauthorized people can’t read it
- What might you want to encrypt?
  - Email or text message
  - Individual file
  - Hard drive, USB stick
  - Communication with a website
  - Everything

One-way functions

- Easy to compute in one direction, but hard to compute in the other
- Hash function
  - Small change in input → big change in output
  - md5(“blase”) = `12b872adb2588c668d706d847fc1da7e`
- Used for storing passwords
- Current research: make hashing slow
  - (Older and less good) bcrypt: iterated hashes
  - scrypt and Argon2: memory-hard

Trapdoor functions

- Easy to compute in one direction, but hard to compute in the other unless you have some extra information
- Encryption: reversible (if you know secret)
  - "this is a test" → `Xe0yUqyOnY8JskyCQ2cYIg==`
  - `Xe0yUqyOnY8JskyCQ2cYIg==` and `chicago` (secret) → "this is a test"

Two main encryption approaches

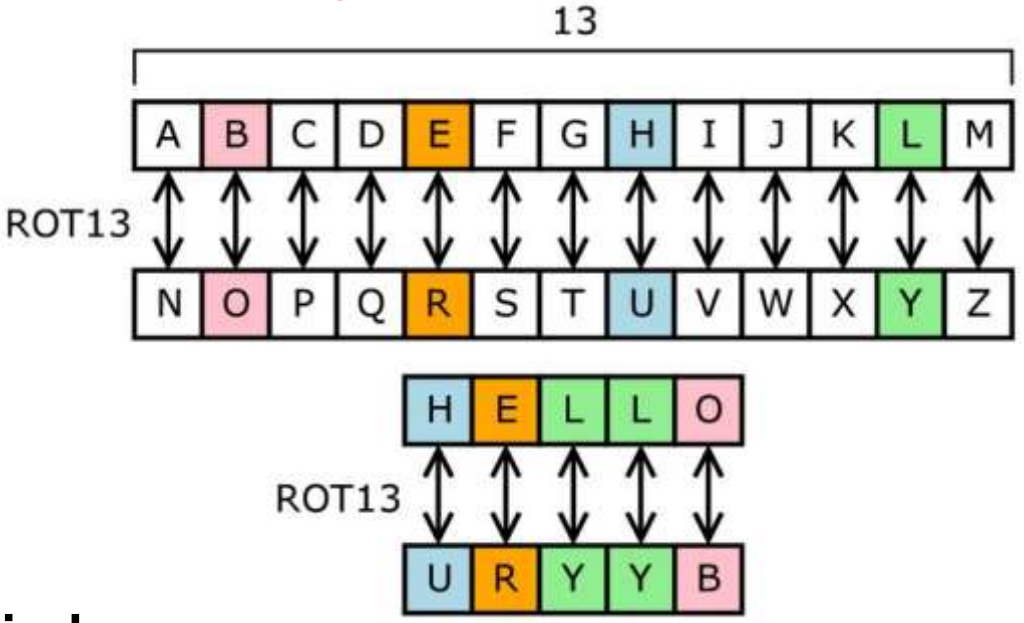
- Symmetric encryption: Same key used for encryption and decryption
  - Requires key exchange (out of band)
  - Prominent examples: AES, DES, etc.
- Asymmetric encryption: Different keys used for encryption and decryption
  - Keypair: public key and private key
  - Prominent examples: RSA, ElGamal, elliptic curve crypto

Properties of encryption

- **Secrecy**: Is Blase the only person who can decrypt my message?
- **Integrity**: Has someone tampered with Blase's message?
- **Authenticity**: Did this message really come from Blase?

Encryption historically

- Caesar shift



- Substitution cipher

CIPHER ALPHABET

A = B	H = A	O = O	V = L
B = V	I = D	P = Y	W = P
C = G	J = Z	Q = F	X = U
D = Q	K = C	R = J	Y = I
E = K	L = W	S = X	Z = R
F = M	M = S	T = H	
G = N	N = E	U = T	

Figure 1

Encryption historically

- Vigenère cipher

- Enigma machine

Block ciphers (symmetric)

- Old (deprecated): DES → 3DES
- AES (Advanced Encryption Standard)
  - Rijndael Cipher (chosen in 2001)
  - 128-bit blocks
  - 128-, 192-, or 256-bit keys
  - https://www.youtube.com/watch?v=evjFwDRTmV0
- No known* feasible* attacks on AES
- Timing side-channel attacks possible

Public key (asymmetric) crypto

- 1970s – Present

Diffie-Hellman key exchange

- 1976: Diffie, Hellman, Merkle
- (Multiplicative group of integers modulus p)
- Generator g
- Prime number p
- Secrets (integers) x and y

Diffie-Hellman key exchange

RSA

- Generate a key:
  - Choose primes p, q
  - Calculate $n = p \cdot q$
  - Private key: $\lambda(n) = lcm(p-1, q-1)$
  - Choose integer e so $1 < e < \lambda(n)$ and e, $\lambda(n)$ coprime
  - Solve for d such that $d \cdot e \equiv 1 \pmod{\lambda(n)}$
  - Release e and n as the public key, but keep d private
- To encrypt: $c \equiv m^e \pmod{n}$
- To decrypt: c^d
  - Equivalent to $m^{ed} \pmod{n}$, which is just m

Usable encryption

Why do user studies?

Purpose	Useful to…
Assess needs	Decide what to build
Evaluate	Determine whether system meets requirements and what needs to be improved
Understand tradeoffs	Decide which features/approaches/systems best fit particular needs
Find root causes	Determine where redesigns or new approaches are needed


What do we Want?
EVIDENCE BASED SCIENCE
When do we Want It?
AFTER PEER REVIEW

User study steps

- Identify research questions, metrics, and use cases
- Decide on type of study and design study protocol
- Develop detailed scripts, surveys, scenarios, incentives, instrumentation, prototypes, recruiting materials, etc.
- Obtain ethics approval
- Pilot and iterate on study design
- Collect data
- Analyze results
- Repeat some or all of these steps as needed

Usable security study challenges

- Keeping it real (ecological validity)
  - Create realistic sense of risk **(but not real risk)**
  - Provide realistic incentives
  - Don’t bias participants
- Measuring the right thing
  - Design the right protocol
  - Control the variables
  - Instrument
- Observing infrequent events and small differences
- Legal, ethical, and practical issues

Why Johnny can’t encrypt

- Why can’t Johnny encrypt?
- Why was it so hard for participants to complete the tasks?
- How did the experimenters motivate the tasks and get participants to care about security?
- What role did attackers play in this user study?

Why Johnny can’t encrypt

- Classic paper in usable security (1999)
- Interfaces are bad
- Metaphors are wrong (and confusing)
- Opaque process
- Key management is difficult

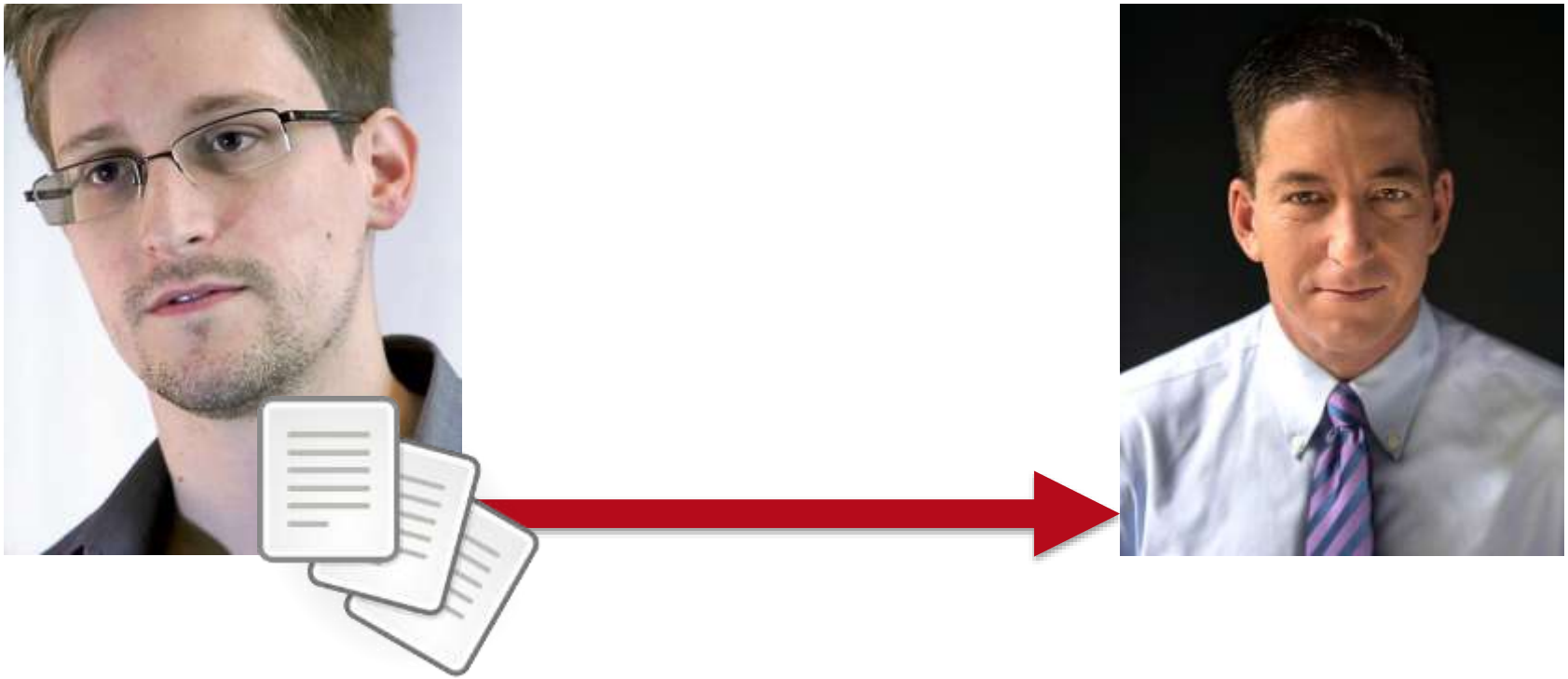
Why Johnny can’t encrypt

- Security principles
  - Unmotivated user
  - Abstraction property
  - Lack of feedback
  - Barn door property
  - Weakest link property
- Cognitive walkthrough vs. user test
- Bad metaphors

Why Glenn couldn’t encrypt

Why Glenn couldn’t encrypt

Why Glenn couldn’t encrypt



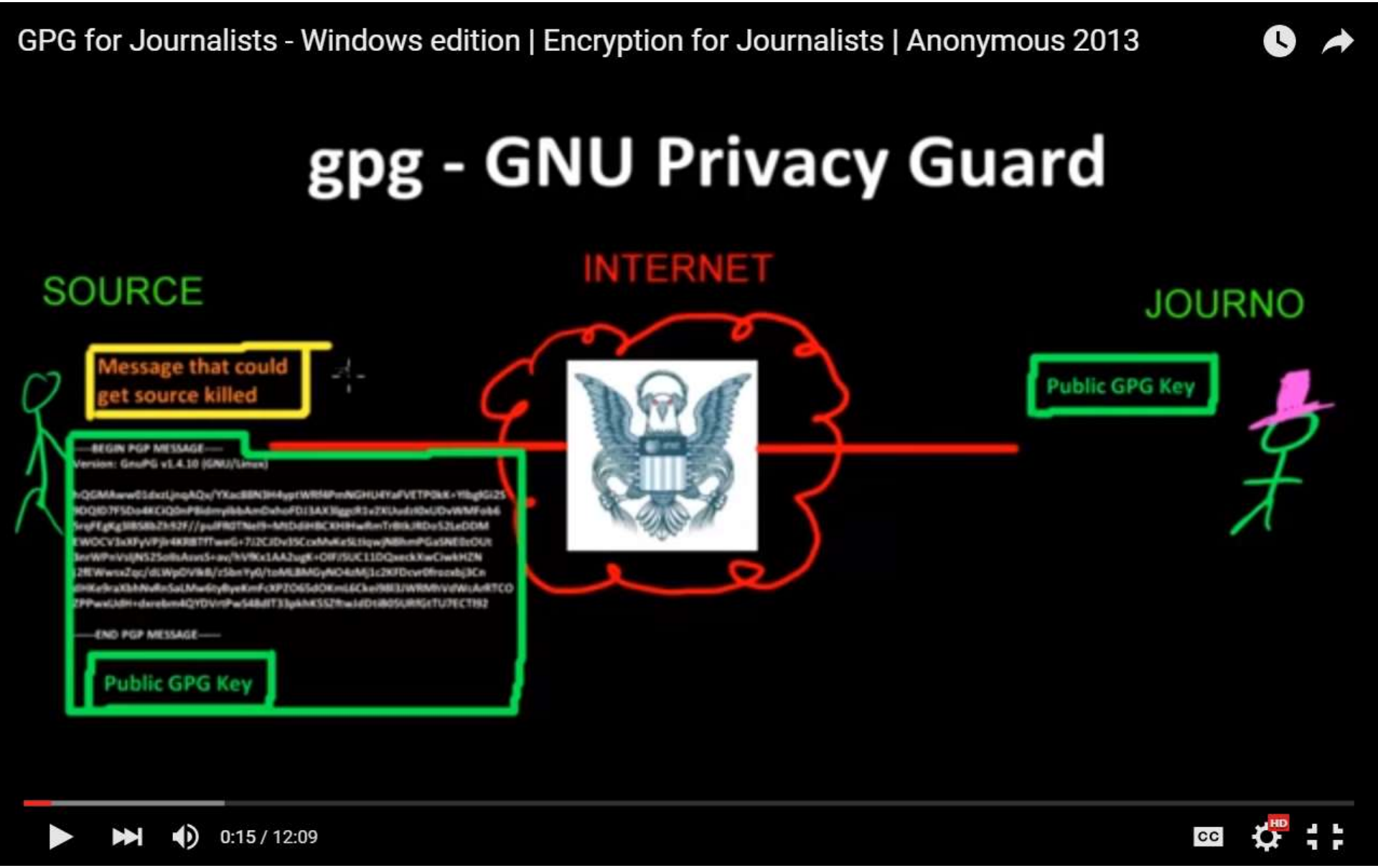
Why Glenn couldn’t encrypt

Why Glenn couldn’t encrypt

- http://vimeo.com/56881481
  - 1:50 – 3:37, 4:10 – 4:58, 11:15 – 11:43
- “And yet, Greenwald still didn't bother learning security protocols. ‘The more he sent me, the more difficult it seemed,’ he says. ‘I mean, now I had to watch a f***ing video . . . ?’”
- Snowden ended up reaching out to Laura Poitras instead

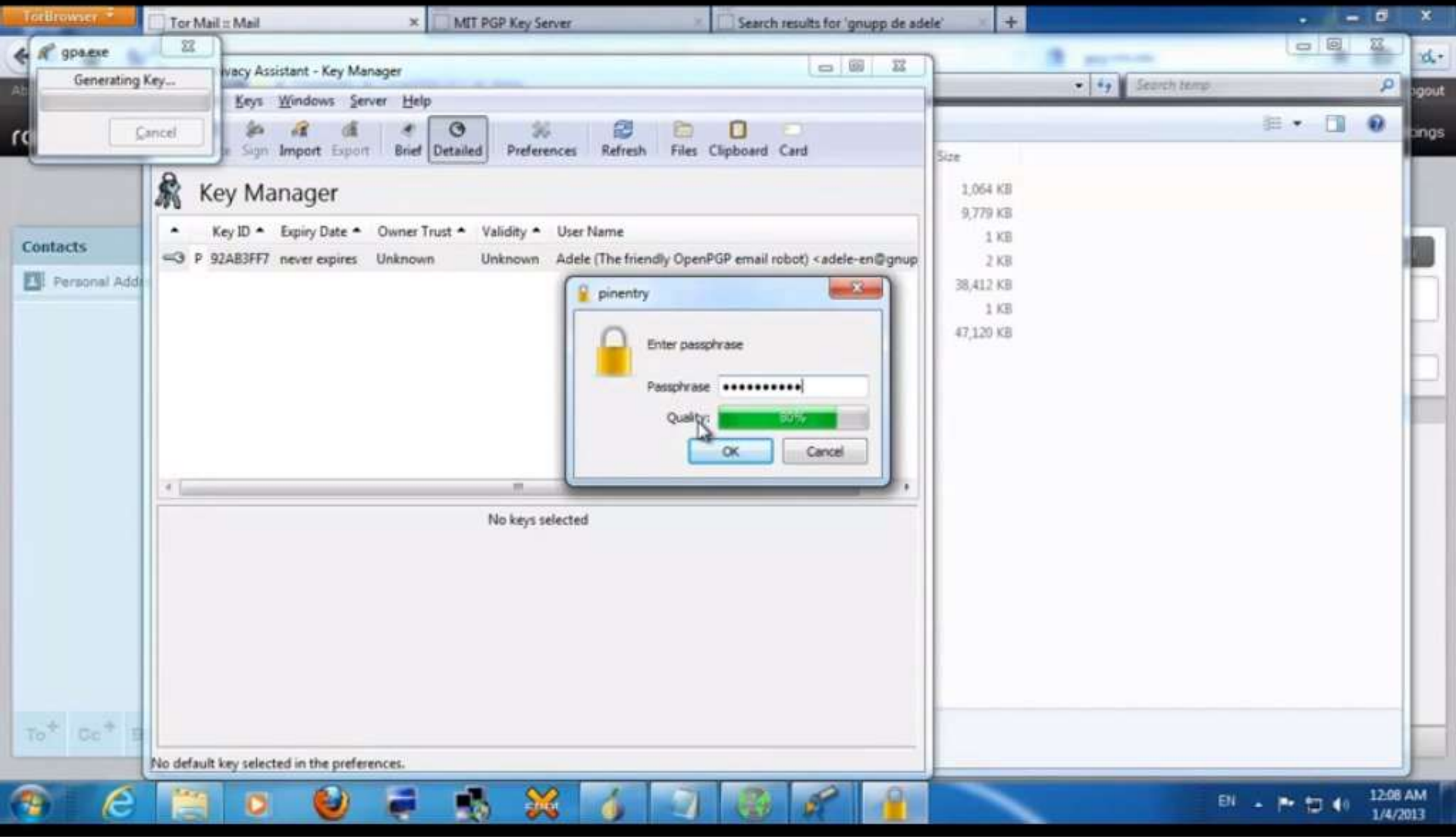
http://www.rollingstone.com/politics/news/snowden-and-greenwald-the-men-who-leaked-the-secrets-20131204
http://www.dailydot.com/politics/edward-snowden-gpg-for-journalists-video-nsa-glenn-greenwald/

GPG for Journalists - Windows edition | Encryption for Journalists | Anonymous 2013

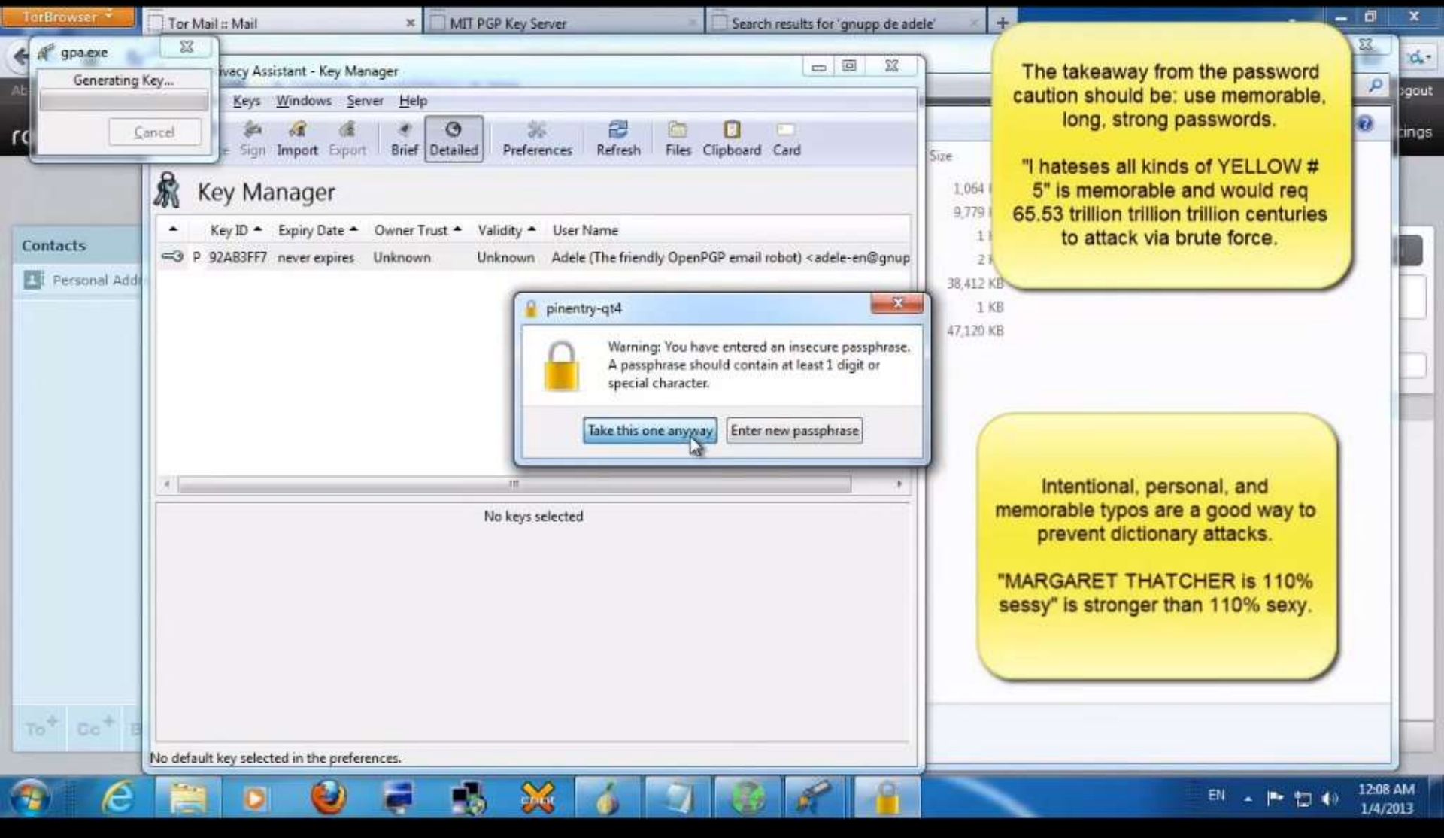


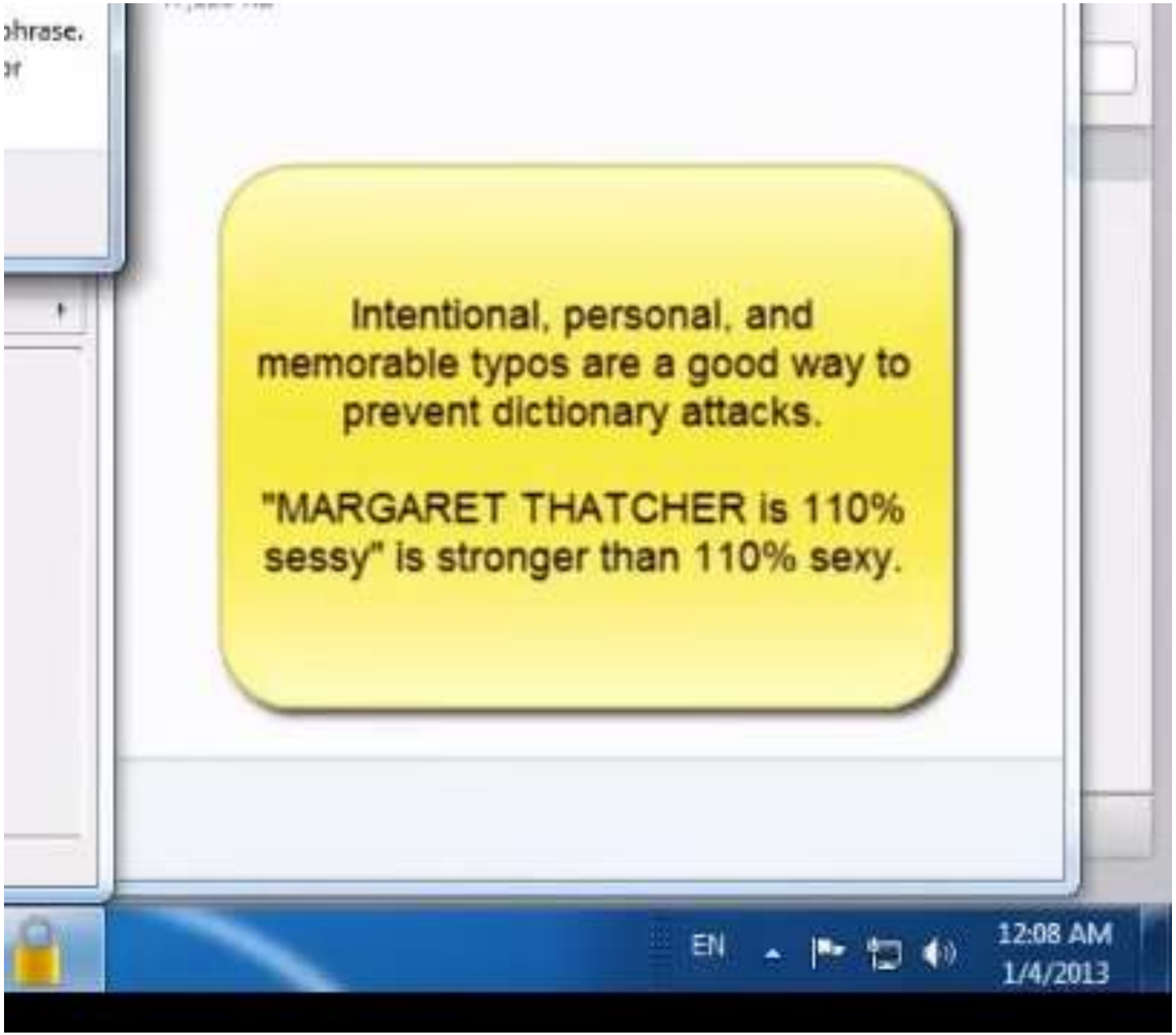
0:15 / 12:09

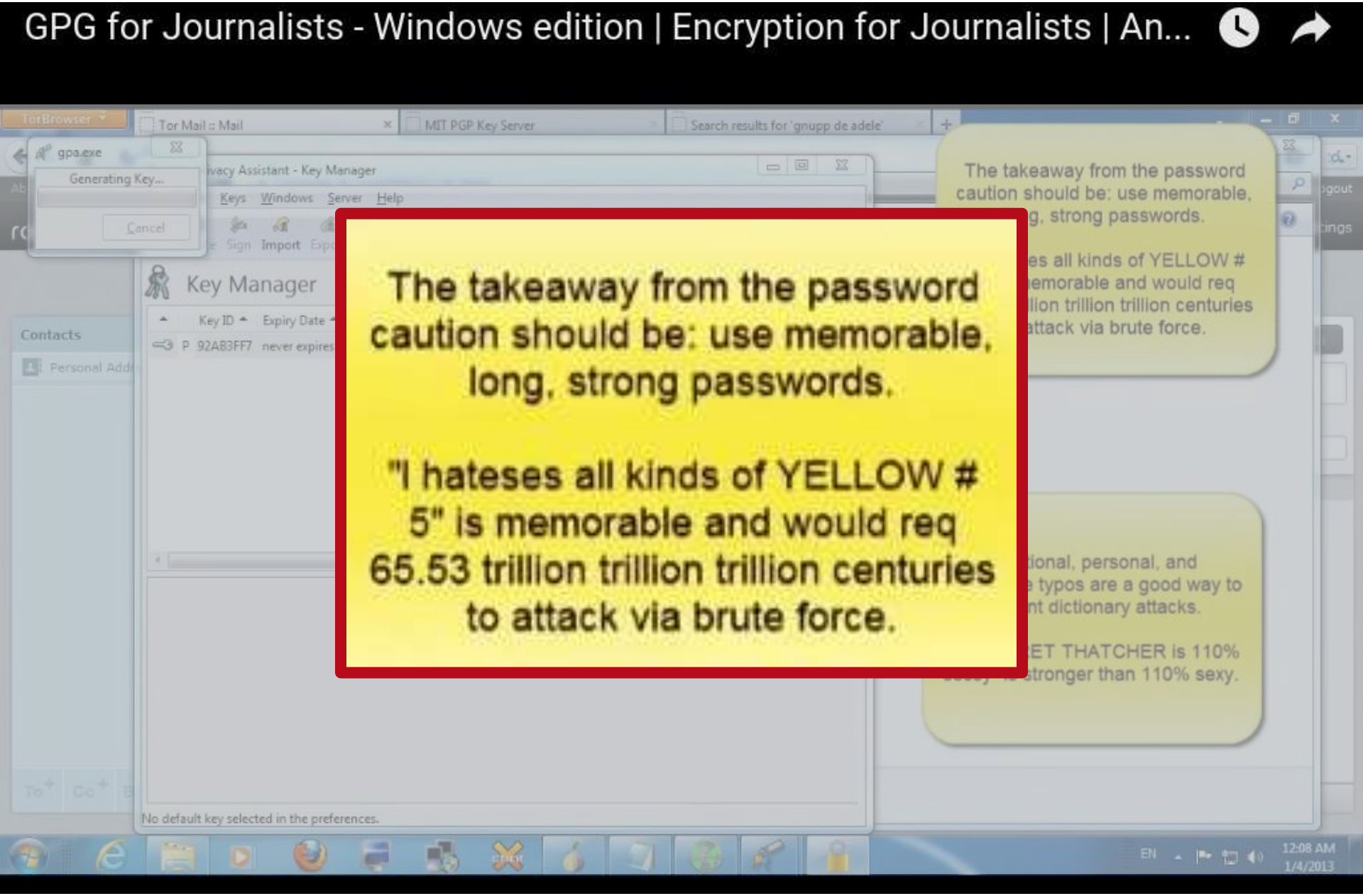
GPG for Journalists - Windows edition | Encryption for Journalists | An...



GPG for Journalists - Windows edition | Encryption for Journalists | An...




Intentional, personal, and memorable typos are a good way to prevent dictionary attacks.
"MARGARET THATCHER is 110% sessy" is stronger than 110% sexy.
EN
12:08 AM
1/4/2013


GPG for Journalists - Windows edition | Encryption for Journalists | An...
The takeaway from the password caution should be: use memorable, long, strong passwords.
"I hateses all kinds of YELLOW # 5" is memorable and would req 65.53 trillion trillion trillion centuries to attack via brute force.
Key Manager
Generating Key...
Cancel
No default key selected in the preferences.

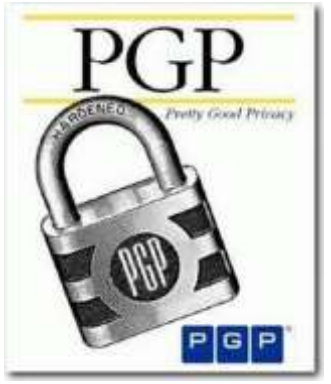
Complexity of asymmetric encryption

- User creates a keypair
  - Public key should be widely distributed
  - Private key should never be distributed
- Private key protected with a password
- Two very different functions:
  - Encrypting (secrecy)
  - Signing (authenticity)
- Need person’s key to communicate

(Just some) usability problems

- Encryption is rarely configured by default
- Public/private key encryption
  - How to get someone's public key?
  - How do I make it work on my phone?
- You often need a good password
  - …and you can't lose it or forget it
- Configuring multiple devices
- "Only paranoid people use encryption"

Open vs. closed ecosystem



- Open standards
  - Don’t require specific ecosystem or software
- Messaging apps (some security-focused)
  - (Sometimes) can minimize configuration
  - More centralized trust

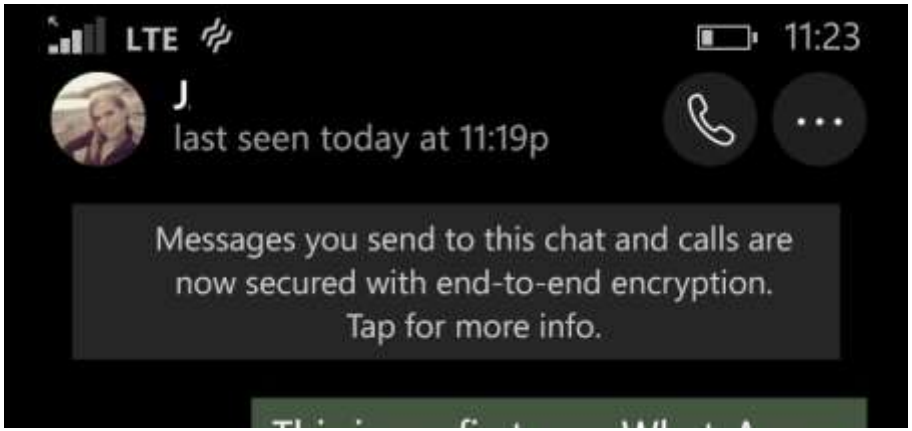
Signal Private Messenger

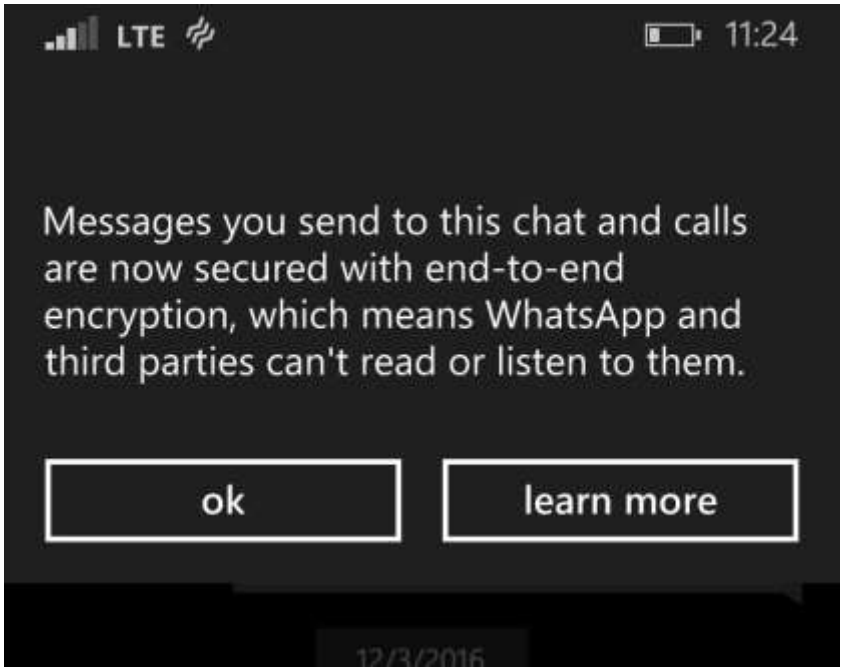
Open Whisper Systems Communication

Everyone

WhatsApp

Whatsapp


LTE
11:23
J
last seen today at 11:19p
Messages you send to this chat and calls are now secured with end-to-end encryption. Tap for more info.


LTE
11:24
Messages you send to this chat and calls are now secured with end-to-end encryption, which means WhatsApp and third parties can't read or listen to them.
ok
learn more

Important distinctions

- **End-to-end encryption**: messages encrypted between sender and recipient
  - In theory, providers can't access
  - **Perfect forward secrecy**: compromise of key does not compromise previous session keys
- As opposed to messages being encrypted between you and the company (e.g., WhatsApp), and then between the company and recipient

Do you have the right key?

- Man-in-the-middle attack

- Ways of trusting a person→key binding:
  - Public-key infrastructure (certifying authorities)
  - Web of trust (someone you trust vouches)
  - Exchange keys out of band
  - Platform provider verifies
  - Key servers: https://pgp.mit.edu/

Verifying you have the right key

Locally verifying key

- Out-of-band channel
- Interaction
  - Bump, button press, shake devices
- Location-limited channel
  - Bluetooth, sounds, wired connection
- Short string comparisons

Remotely verifying key

GnuPG

```
3A70 F9A0 4ECD B5D7 8A89
D32C EDA0 A352 66E2 C53D
```

OpenSSH

```
ef:6d:bb:4c:25:3a:6d:f8:79:d3:a7:90:db:c9:b4:25
```

bubblebabble

```
xucef-masiv-zihyl-bicyr-zalot-cevyt-lusob-
negul-biros-zuhal-cixex
```

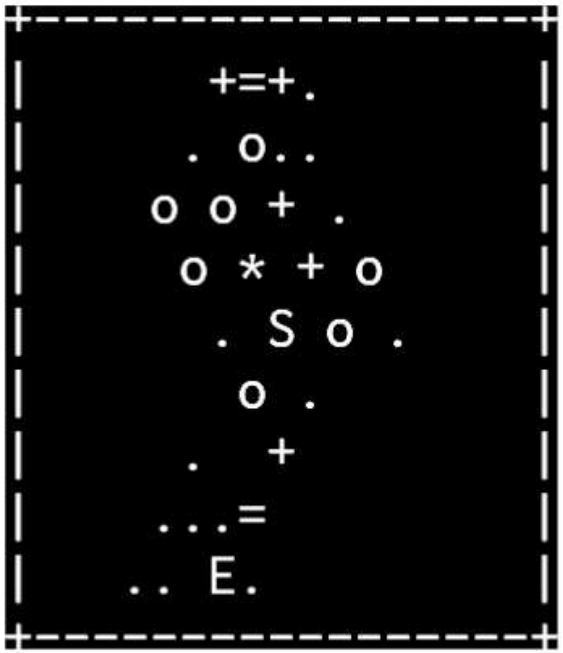
OTR

```
4206EA15 1E029807 C8BA9366 B972A136 C6033804
```

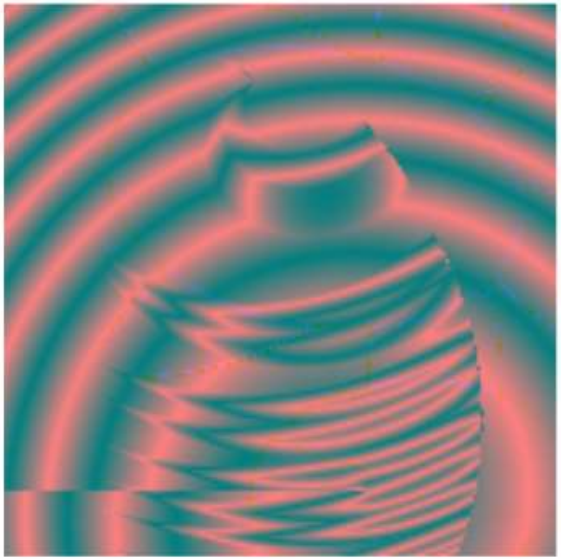
WhatsApp

```
54040 65258 71972 73974
10879 55897 71430 75600
25372 60226 27738 71523
```

Remotely verifying key



(a) OpenSSH Visual Host Key



(b) Vash

(c) Unicorn

Current Research in Cryptography

Fully Homomorphic Encryption

- Homomorphism: transformation between sets that preserves relations between elements
- Allows for computation on encrypted data
- Partially homomorphic systems:

Unpadded RSA [edit]

If the RSA public key is modulus m and exponent e, then the encryption of a message x is given by $\mathcal{E}(x) = x^e \mod m$. The homomorphic property is then

$$\mathcal{E}(x_1) \cdot \mathcal{E}(x_2) = x_1^e x_2^e \mod m = (x_1 x_2)^e \mod m = \mathcal{E}(x_1 \cdot x_2)$$

Paillier [edit]

In the Paillier cryptosystem, if the public key is the modulus m and the base g, then the encryption of a message x is $\mathcal{E}(x) = g^x r^m \mod m^2$, for some random $r \in \{0, \ldots, m-1\}$. The homomorphic property is then

$$\mathcal{E}(x_1) \cdot \mathcal{E}(x_2) = (g^{x_1} r_1^m)(g^{x_2} r_2^m) \mod m^2 = g^{x_1+x_2}(r_1 r_2)^m \mod m^2 = \mathcal{E}(x_1 + x_2)$$

- Fully homomorphic systems (2010 – now)

Searchable Encryption

- Can we have things (e.g., documents) that are encrypted yet still searchable?
  - Conceptualize security roughly as leaking nothing but the results of the search queries
- Can we have a database that is encrypted yet can still be queried?
  - Proxy that rewrites queries and returns decrypted results

Oblivious RAM

- Algorithms can give away information based on their access patterns
- Can we automatically “rewrite” algorithms so that their patterns of memory access don’t leak information?