



# BetterCrypto · org

## Applied Crypto Hardening

David Durvaux  
Aaron Kaplan

Brussels, 9<sup>th</sup> June 2014

# This talk is public and recorded



**Why better crypto?**



**The NSA**

*The only part of government  
that actually listens.*

# But of course...

- It is not only the NSA, who intercepts
- Other nations now have a blueprint (thanks to Snowden) in case they did not have the technical skills yet
- Criminals now have a blueprint,...
- Everyone has!
- So, what can we do?

**Don't give them anything for free**

*It's your home, you fight*

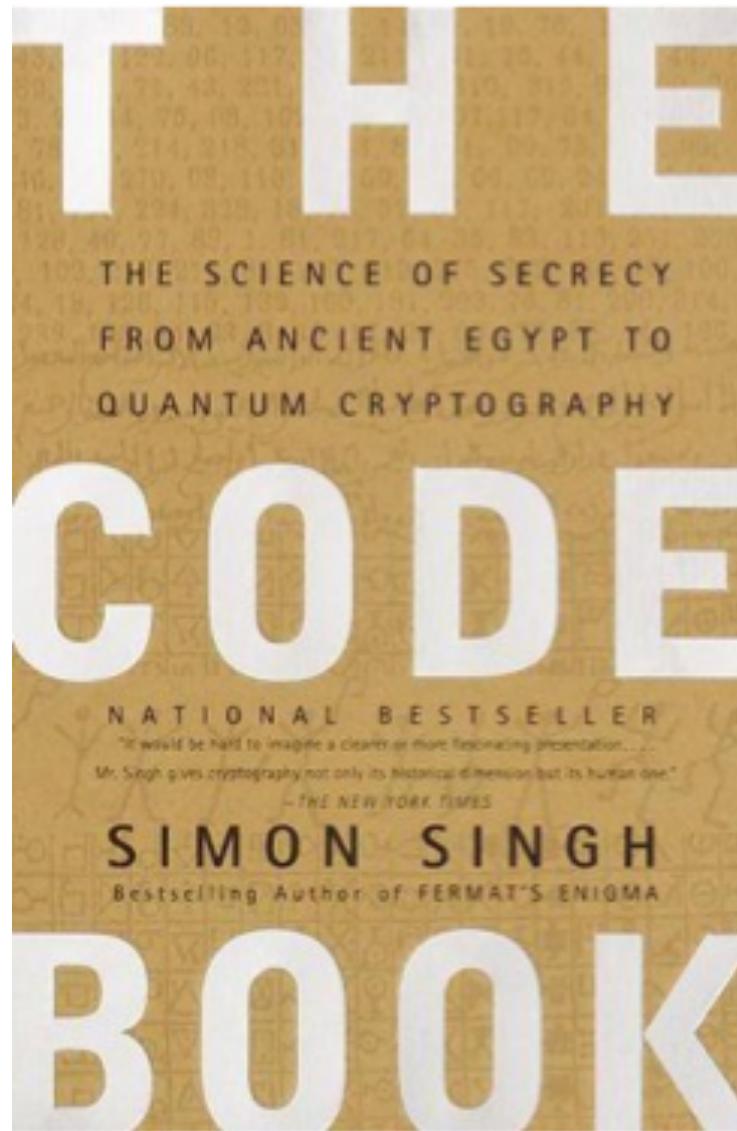
# Who (authors of bettercrypto)

Wolfgang Breyha (uni VIE),  
David Durvaux (CERT.be),  
Tobias Dussa (KIT-CERT),  
L. Aaron Kaplan (CERT.at),  
Florian Mendel (IAIK/A-Sit)  
Christian Mock (coretec),  
Daniel Kovacic (A-Trust),  
Manuel Koschuch (FH Campus Wien),  
Adi Kriegisch (VRVis),  
Ramin Sabet (A-Trust),  
Aaron Zauner (azet.org),  
Pepi Zawodsky (maclemon.at),

*And many other contributors!!*

# Agenda

- Pieces of History
- Introduction to BetterCrypto project
- Cryptography in a nutshell
- Practical Settings
- Testing
- Demo
- Conclusion



# Pieces of History

# Caesar

- Vigenère Cipher



# Mary Queen of Scots



- Trial against Queen Elizabeth
- Was executed after her code was broken (1587)

# Enigma

- Secret in code book



Bundesarchiv, Bild 101I-241-2173-09  
Foto: Grupp | 1943/1944



# BetterCrypto

# Why?

- Crypto is cryptic
- A lot of difficult concepts
- A lot of algorithms
- A lot of parameters
- ...

# The Idea

- Really difficult for systems administrators
  - A “cookbook” can help!
    - That’s BetterCrypto

# That's not...

- A crypto course
- A static document

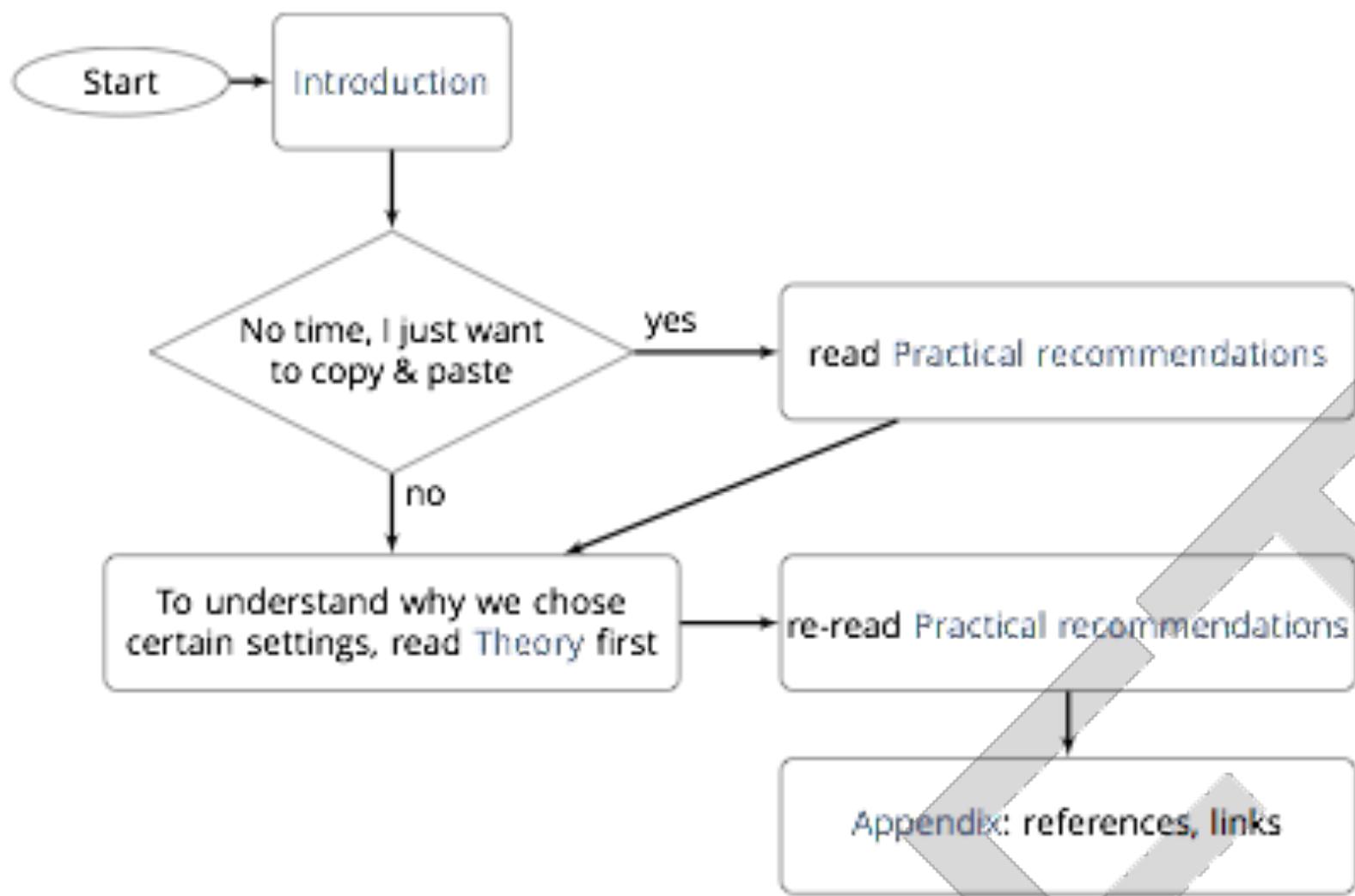
# In short

- Community effort to produce best common practices
- Continuous effort
- Mixed expertises
- Open Source
- Open to comments / suggestions / improvements

# 2 parts

- First part = configurations
  - The most important part
  - Cover as many tools as possible
- Second part = theory
  - Explain and justify choose we made
    - Transparency

# How to use?



A CRYPTO NERD'S  
IMAGINATION:

HIS LAPTOP'S ENCRYPTED.  
LET'S BUILD A MILLION-DOLLAR  
CLUSTER TO CRACK IT.

NO GOOD! IT'S  
4096-BIT RSA!

BLAST! OUR  
EVIL PLAN  
IS FOILED!



WHAT WOULD  
ACTUALLY HAPPEN:

HIS LAPTOP'S ENCRYPTED.  
DRUG HIM AND HIT HIM WITH  
THIS \$5 WRENCH UNTIL  
HE TELLS US THE PASSWORD.

GOT IT.



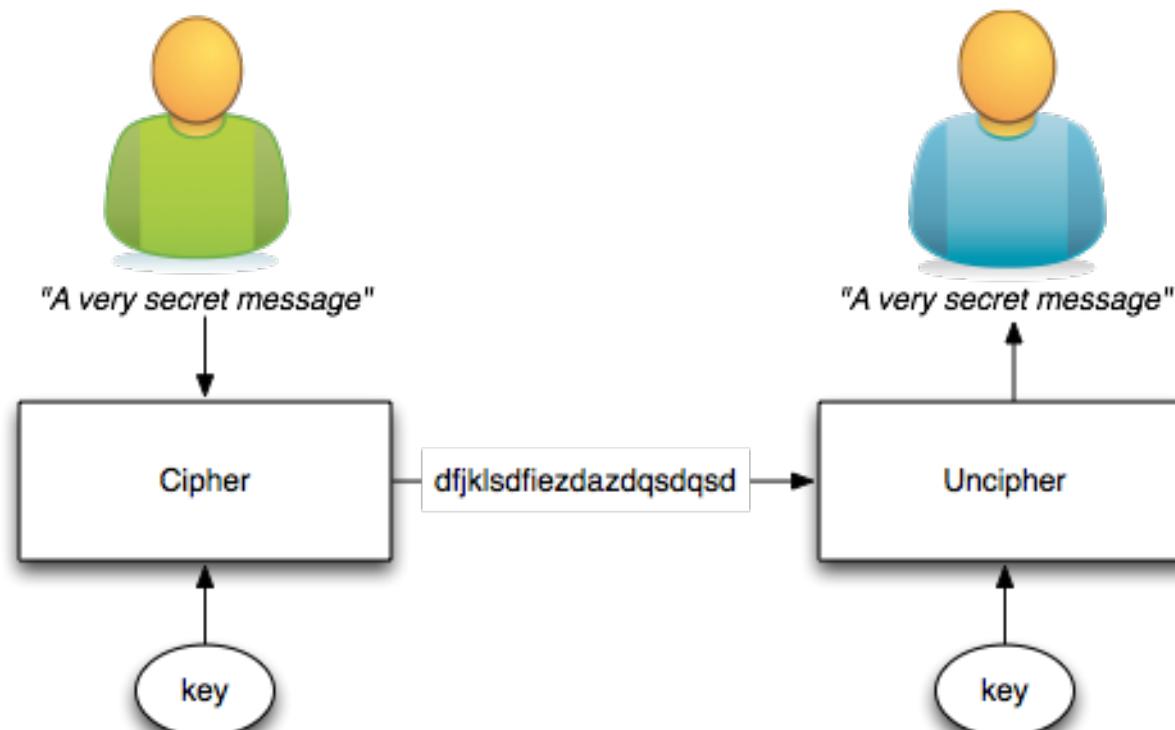
# Crypto in a nutshell

# Goals

- 2 types of goals:
- protect the content of the message
  - Eavesdropping
  - Alteration
- identify the author (signatures)
  - At least the one who controls the key
- Can be combined

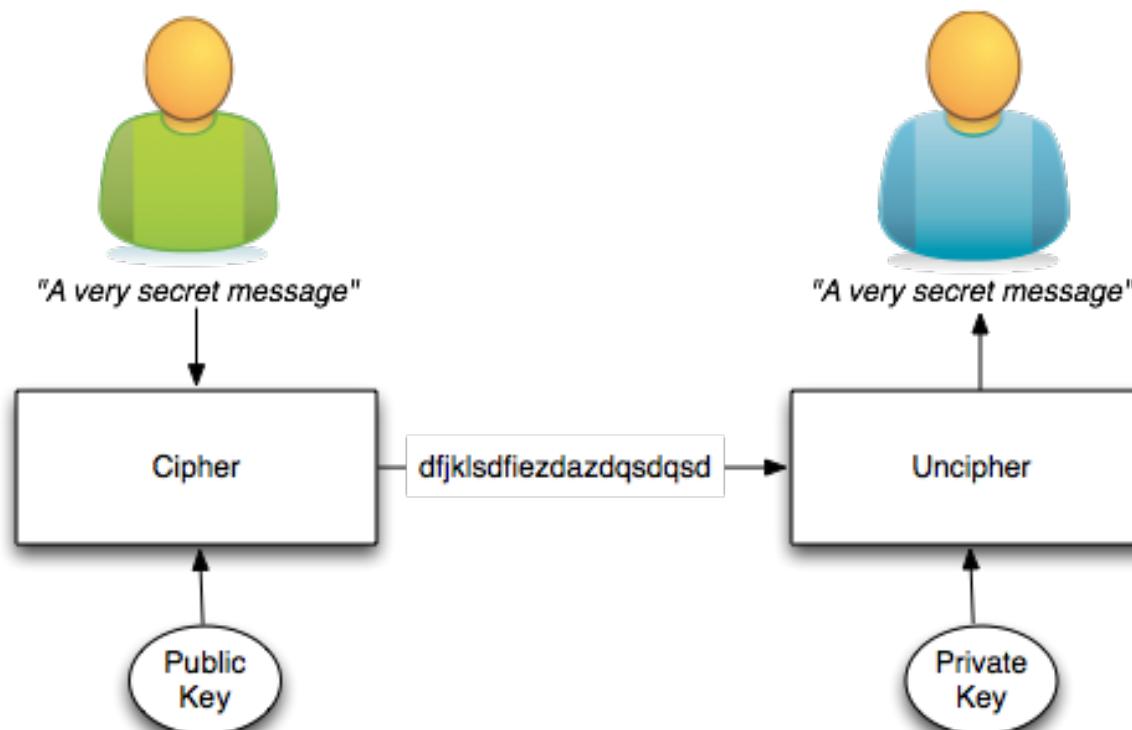
# Symmetric Ciphering

- The key is shared



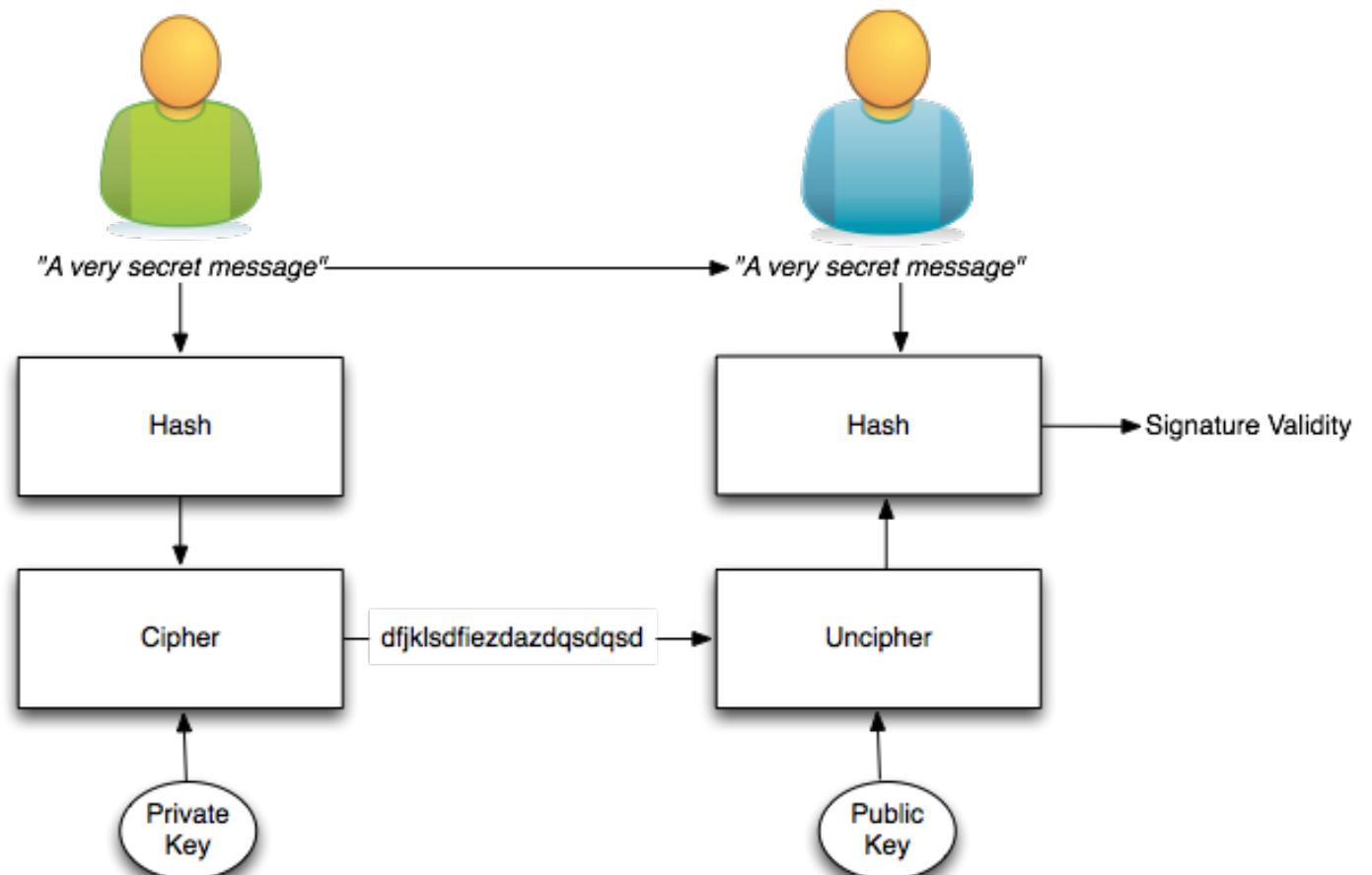
# Asymmetric Ciphering

- Public key is published
- Private key **HAS** to be secured



# Signing

- Author identity is proved
  - Signed with the **private** key

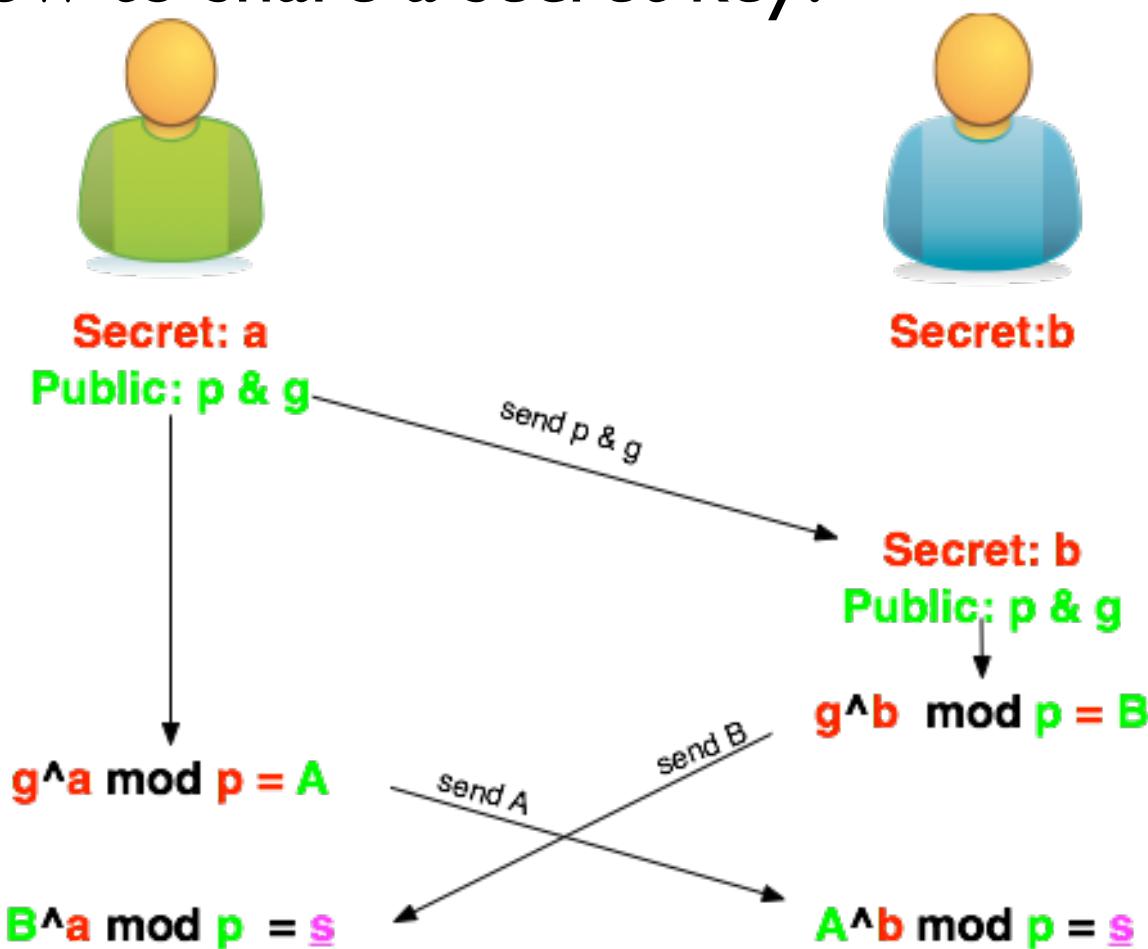


# The asymmetric magic

- RSA “formula” :  $c = m^e \text{ mod}(n)$ 
  - with
    - $c$  which is the ciphertext
    - $m$  is the cleartext message
    - $e$  and  $n$  are the public key
- Uncipher with  $m = c^d \text{ mod}(n)$ 
  - $d$  being the private key

# Diffie-Hellman

- How to share a secret key?



# Ephemeral Diffie-Hellman

- Regular mode
  - Public and private keys are kept
- Ephemeral mode
  - New keys are generated each time
    - By one of the parties at least

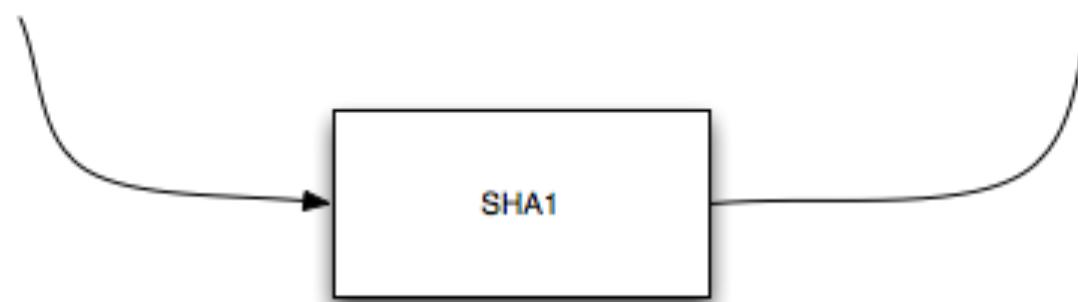
# Hashing

- Take long piece of data and produce a probably unique signature
- Probability of collision for SHA1:
  - 1 over  
**146150163733090291820368483271628  
3019655932542976**

This is a really long text.

I can even put a full book over there  
but it would be too long for my small  
schema ;).

5b833db762858ed42050809816e4028421b6e2a3



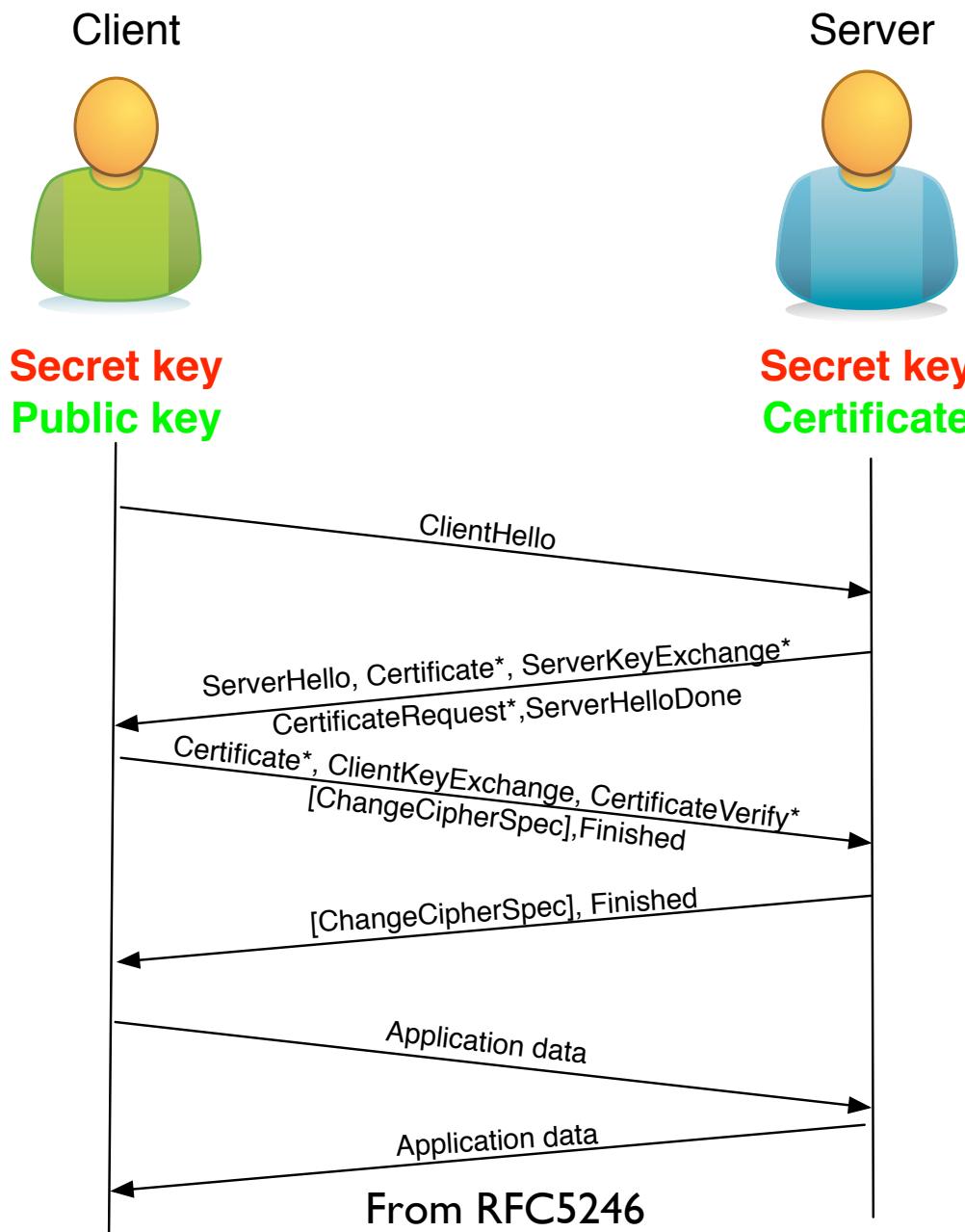
# ECC

- Elliptic curve cryptography (ECC)
- Finding the discrete logarithm of a random elliptic curve element
  - Only knowing a base point
  - Assumed to be infeasible
- Reduced key length

# Some thoughts on ECC

- Currently this is under heavy debate
- Trust the Math
- “Nothing Up My Sleeve Numbers”
- eg. NIST P-256 (<http://safecurves.cr.yp.to/rigid.html>)
- Coefficients generated by hashing the unexplained seed  
c49d3608 86e70493 6a6678e1 139d26b7 819f7e90.
- Might have to change settings tomorrow
- Most Applications only work with NIST-Curves

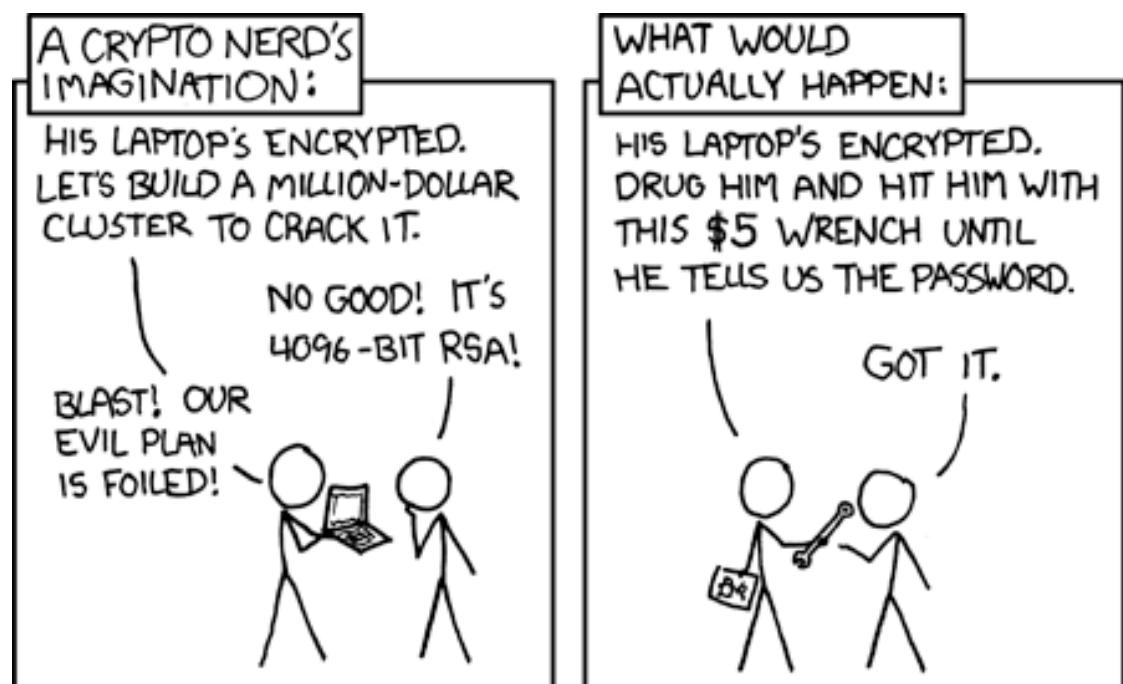
# TLS



- Hello includes
  - Random number
  - Cipher suite
- Finished
  - 1st cipher message with negotiated parameters
  - Algorithm
  - Key
  - Secret

# Forward Secrecy-Motivation:

- Three letter agency (TLA) stores all ssl traffic
- Someday TLA gains access to ssl-private key  
(Brute Force, Physical Force)
- TLA can decrypt all stored traffic



# Perfect Forward Secrecy

- DHE: Diffie Hellman Ephemeral
- Ephemeral: new key for each execution of a key exchange process
- SSL private-Key only for authentication
- Alternative new ssl private key every x days months
- Pro:
  - Highest Security against future attacks
- Contra:
  - Elliptic Curve
  - Processing costs

# Stream vs Block Cipher

- Stream cipher
  - Generate an “infinite” key stream
  - Difficult to correctly use
    - Re-use of keys
  - Faster
- Block cipher
  - Cipher by block with padding
  - Could include integrity protection

# RNGs

- RNGs are *important*.
- Nadia Heninger et al / Lenstra et al

	Our TLS Scan	Our SSH Scans
Number of live hosts	12,828,613 (100.00%)	10,216,363 (100.00%)
... using repeated keys	7,770,232 (60.50%)	6,642,222 (65.00%)
... using vulnerable repeated keys	714,243 (5.57%)	981,166 (9.60%)
... using default certificates or default keys	670,391 (5.23%)	
... using low-entropy repeated keys	43,852 (0.34%)	
... using RSA keys we could factor	64,081 (0.50%)	2,459 (0.03%)
... using DSA keys we could compromise		105,728 (1.03%)
... using Debian weak keys	4,147 (0.03%)	53,141 (0.52%)
... using 512-bit RSA keys	123,038 (0.96%)	8,459 (0.08%)
... identified as a vulnerable device model	985,031 (7.68%)	1,070,522 (10.48%)
... model using low-entropy repeated keys	314,640 (2.45%)	

- Entropy after startup: embedded devices

# RNGs

- Weak RNG
  - Dual EC\_DRBG is BROKEN (backdoored, used in RSA-toolkit)
  - Intel RNG ? Recommendation: add System-Entropy (Network). Entropy only goes up.
- Tools (eg. HaveGE <http://dl.acm.org/citation.cfm?id=945516>)
- RTFM
  - when is the router key generated
  - Default Keys ?
- Re-generate keys from time to time

# Some algorithms

- Symetric Ciphering
  - AES (Rijndael)
  - Camellia
- Asymmetric Ciphering
  - RSA
  - PGP (GPG)

# Some algorithms

- Hash
  - SHA1
  - SHA256
  - SHA512
- Key Exchange
  - Diffie Hellman

# Algorithm vs Implementation!

- Heartbeat

# Heartbeat

```
/* Enter response type, length and copy payload */
    *bp++ = TLS1_HB_RESPONSE;
        s2n(payload, bp);
memcpy(bp, pl, payload);
```

- payload (pl) and payload\_length (payload) are controlled by attacker
- memcpy will copy a part of the victim memory to the reply...

# Cost of encryption

```
$ time openssl enc -e -a -aes-128-cbc -in ./rfc791.txt \  
-out /tmp/rfc.aes -k "Super Key" -S 01EF
```

real 0m0.014s

user 0m0.004s

sys 0m0.003s

```
$ time gpg -a -u 57AB3358 -r 77659F3E -e ./rfc791.txt
```

real 0m0.069s

user 0m0.048s

sys 0m0.008s

GPG is 4,93 time slower!

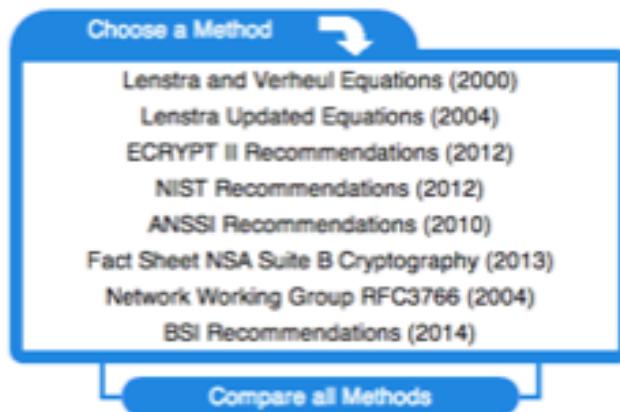
# Keylengths

*On the choice between AES256 and AES128: I would never consider using AES256, just like I don't wear a helmet when I sit inside my car. It's too much bother for the epsilon improvement in security.”*

— Vincent Rijmen in a personal mail exchange Dec 2013

# Keylengths

- <http://www.keylength.com/>
- Recommended Keylengths, Hashing algorithms, etc.
- Currently:
  - RSA:  $\geq$  3248 bits (Encrypt II)
  - ECC:  $\geq$  256
  - SHA 2+ (SHA 256,...)
  - AES 128 is good enough



### 1 Reference for the comparison

You can enter the year until when your system should be protected and see the corresponding key sizes or you can enter a key/hash/group size and see until when you would be protected.

Enter an elliptic curve key size:  bits

### 2 Compare

Method	Date	Symmetric	Asymmetric	Discrete Logarithm Key	Elliptic Curve	Hash
[1] Lenstra / Verheul	2084	135	7813 6816	241	7813	257 269
[2] Lenstra Updated	2090	128	4440 6974	256	4440	256 256
[3] ECRYPT II	2031 - 2040	128	3248	256	3248	256 256
[4] NIST	> 2030	128	3072	256	3072	256 256
[5] ANSSI	> 2020	128	4096	200	4096	256 256
[6] NSA	-	128	-	-	-	256 256
[7] RFC3766	-	136	3707	272	3707	257 -
[8] BSI (signature only)	> 2020	-	1976	256	2048	250 256

# BetterCrypto CipherSuite

- 2 cipher suites
  - version A
    - stronger
    - less supported client
  - version B
    - weaker
    - more “universal”

# Some general thoughts on settings

- General
  - Disable SSL 2.0 (weak algorithms)
  - Disable SSL 3.0 (BEAST vs IE/XP)
  - Enable TLS 1.0 or preferably better
  - Disable TLS-Compression (SSL-CRIME Attack)
  - Implement HSTS (HTTP Strict Transport Security)

# Cipher Suite A

- TLS 1.2
- Perfect forward secrecy / ephemeral Diffie Hellman
- Strong MACs (SHA-2) or
- GCM as Authenticated Encryption scheme

ID	OpenSSL Name	Version	KeyEx	Auth	Cipher	MAC
0x009F	DHE-RSA-AES256-GCM-SHA384	TLSv1.2	DH	RSA	AESGCM(256)	AEAD
0x006B	DHE-RSA-AES256-SHA256	TLSv1.2	DH	RSA	AES(256) (CBC)	SHA256
0xC030	ECDHE-RSA-AES256-GCM-SHA384	TLSv1.2	ECDH	RSA	AESGCM(256)	AEAD
0xC028	ECDHE-RSA-AES256-SHA384	TLSv1.2	ECDH	RSA	AES(256) (CBC)	SHA384

# CiperSuite B

- TLS 1.2, TLS 1.1, TLS 1.0
- Allowing SHA-1

# Cipher Suite B

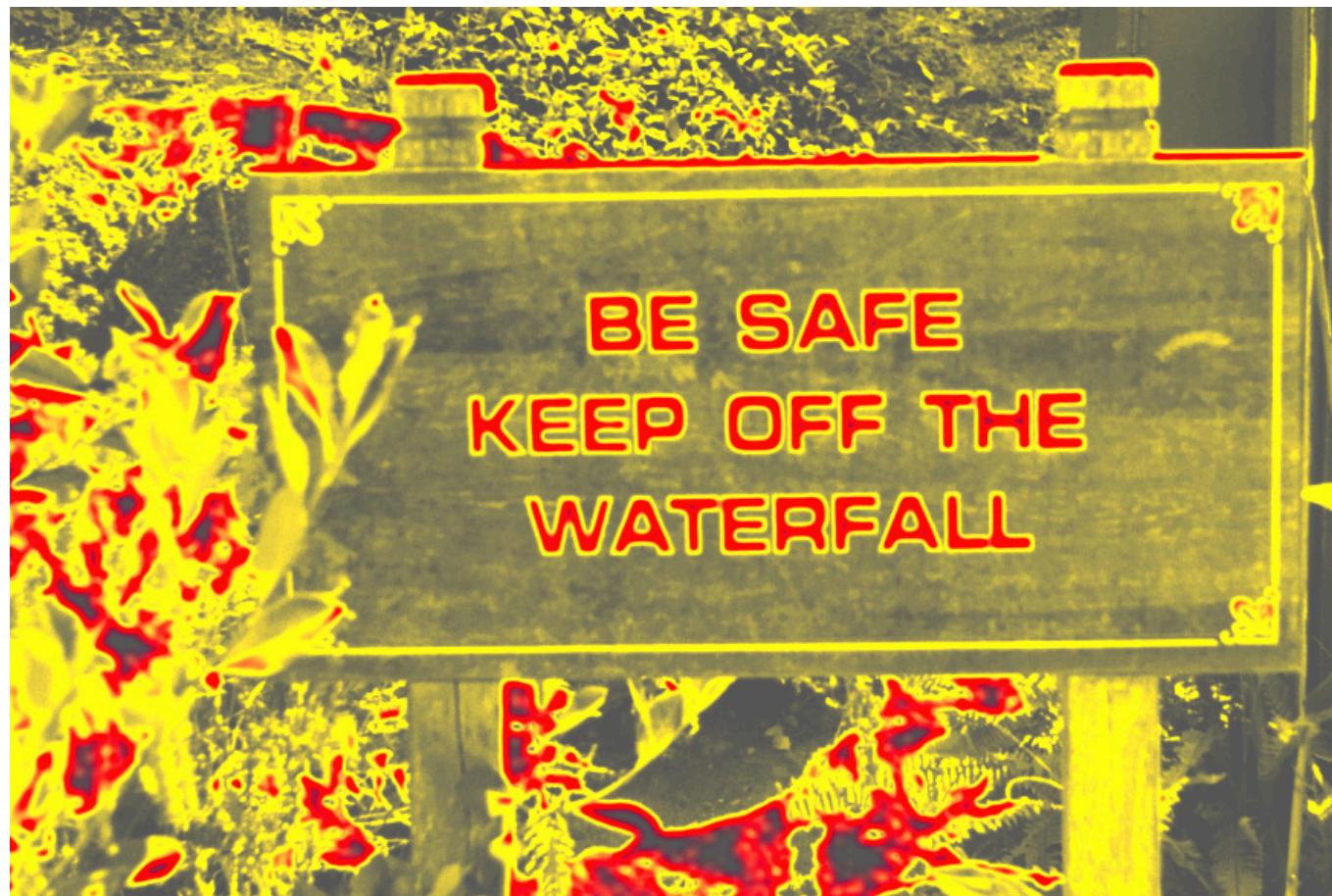
ID	OpenSSL Name	Version	KeyEx	Auth	Cipher	MAC
0x009F	DHE-RSA-AES256-GCM-SHA384	TLSv1.2	DH	RSA	AESGCM(256)	AEAD
0x006B	DHE-RSA-AES256-SHA256	TLSv1.2	DH	RSA	AES(256)	SHA256
0xC030	ECDHE-RSA-AES256-GCM-SHA384	TLSv1.2	ECDH	RSA	AESGCM(256)	AEAD
0xC028	ECDHE-RSA-AES256-SHA384	TLSv1.2	ECDH	RSA	AES(256)	SHA384
0x009E	DHE-RSA-AES128-GCM-SHA256	TLSv1.2	DH	RSA	AESGCM(128)	AEAD
0x0067	DHE-RSA-AES128-SHA256	TLSv1.2	DH	RSA	AES(128)	SHA256
0xC02F	ECDHE-RSA-AES128-GCM-SHA256	TLSv1.2	ECDH	RSA	AESGCM(128)	AEAD
0xC027	ECDHE-RSA-AES128-SHA256	TLSv1.2	ECDH	RSA	AES(128)	SHA256
0x0088	DHE-RSA-CAMELLIA256-SHA	SSLv3	DH	RSA	Camellia(256)	SHA1
0x0039	DHE-RSA-AES256-SHA	SSLv3	DH	RSA	AES(256)	SHA1
0xC014	ECDHE-RSA-AES256-SHA	SSLv3	ECDH	RSA	AES(256)	SHA1
0x0045	DHE-RSA-CAMELLIA128-SHA	SSLv3	DH	RSA	Camellia(128)	SHA1
0x0033	DHE-RSA-AES128-SHA	SSLv3	DH	RSA	AES(128)	SHA1
0xC013	ECDHE-RSA-AES128-SHA	SSLv3	ECDH	RSA	AES(128)	SHA1
0x0084	CAMELLIA256-SHA	SSLv3	RSA	RSA	Camellia(256)	SHA1
0x0035	AES256-SHA	SSLv3	RSA	RSA	AES(256)	SHA1
0x0041	CAMELLIA128-SHA	SSLv3	RSA	RSA	Camellia(128)	SHA1
0x002F	AES128-SHA	SSLv3	RSA	RSA	AES(128)	SHA1

# Compatibility (B suite)



## Handshake Simulation

<a href="#">Bing Oct 2013</a>	TLS 1.0	TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x39) FS	256
<a href="#">Chrome 31 / Win 7</a>	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	256
<a href="#">Firefox 10.0.12 ESR / Win 7</a>	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	256
<a href="#">Firefox 17.0.7 ESR / Win 7</a>	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	256
<a href="#">Firefox 21 / Fedora 19</a>	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	256
<a href="#">Firefox 24 / Win 7</a>	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	256
<a href="#">Googlebot Oct 2013</a>	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	256
<a href="#">IE 6 / XP No FS<sup>1</sup> No SNI<sup>2</sup></a>			Fai <sup>3</sup>
<a href="#">IE 7 / Vista</a>	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	256
<a href="#">IE 8 / XP No FS<sup>1</sup> No SNI<sup>2</sup></a>			Fai <sup>3</sup>
<a href="#">IE 8-10 / Win 7</a>	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	256
<a href="#">IE 11 / Win 7</a>	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xe014) FS	256
<a href="#">IE 11 / Win 8.1</a>	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xe014) FS	256
<a href="#">Java 6u45 No SNI<sup>2</sup></a>			Fai <sup>3</sup>
<a href="#">Java 7u25</a>			Fai <sup>3</sup>
<a href="#">OpenSSL 0.9.8y</a>	TLS 1.0	TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x39) FS	256
<a href="#">OpenSSL 1.0.1e</a>	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xd030) FS	256
<a href="#">Opera 17 / Win 7</a>	TLS 1.2	TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 (0x6b) FS	256
<a href="#">Safari 5.1.9 / OS X 10.6.8</a>	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xe014) FS	256
<a href="#">Safari 6 / iOS 6.0.1</a>	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xe028) FS	256
<a href="#">Safari 6.0.4 / OS X 10.8.4</a>	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xe014) FS	256
<a href="#">Safari 7 / OS X 10.9</a>	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xe028) FS	256
<a href="#">Tor 17.0.9 / Win 7</a>	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	256
<a href="#">Yahoo Slurp Oct 2013</a>	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	256



# Practical Settings

# Tools covered

- Webservers
  - Apache
  - lighttpd
  - nginx
  - Microsoft IIS

# Tools covered

- SSH
  - Open SSH
  - Cisco ASA
  - Cisco IOS

# Tools covered

- Mail servers
  - Dovecot
  - cyrus-imapd
  - Postfix
  - Exim

# Tools covered

- VPN
- IPSec
- CheckPoint Firewall- I
- OpenVPN
- PPPTP
- Cisco ASA
- OpenSWAN
- tinc

# Tools covered

- PGP/GPG
- IPMI/ILO
- Instant Messaging
  - ejabberd
  - OTR
  - Charybdis
  - SILC

# Tools covered

- Database systems
  - Oracle
  - MySQL
  - DB2
  - PostgreSQL

# Tools covered

- Proxy
  - squid
  - Bluecoat
  - Pound
- Kerberos

# Mail Encryption

- GPG / PGP – end to end protection
  - Use public / private crypto to protect your emails
  - Chain of trust
  - Independent of the mail client / transport layer
  - Can be used to verify author and/or protect content
- STARTTLS for SMTP – in transit

# Let's have a look

Draft revision: e516f3c (2014-03-24 12:43:28 +0100) Ulrich



bettercrypto.org

---

## Applied Crypto Hardening

Wolfgang Breyha, David Durvaux, Tobias Dussa, L. Aaron Kaplan, Florian Mendel, Christian Mock, Manuel Koschuch, Adi Kriegisch, Ulrich Pöschl, Ramin Sabet, Berg San, Ralf Schlatterbeck, Thomas Schreck, Alexander Würstlein, Aaron Zauner, Pepi Zawodsky

(University of Vienna, CERT.be, KIT-CERT, CERT.at, A-SIT/IAIK, coretec.at, FH Campus Wien, VRVis, MILCERT Austria, A-Trust, Runtux.com, Friedrich-Alexander University Erlangen-Nuremberg, azet.org, maclemon.at)

---

March 26, 2014

DRAFT

Draft revision: e516f3c (2014-03-24 12:43:28 +0100) Ulrich

# Apache

## Selecting cipher suites:

```
SSLProtocol All -SSLv2 -SSLv3
SSLHonorCipherOrder On
SSLCompression off
# Add six earth month HSTS header for all users...
Header add Strict-Transport-Security "max-age=15768000"
# If you want to protect all subdomains, use the following header
# ALL subdomains HAVE TO support https if you use this!
# Strict-Transport-Security: max-age=15768000 ; includeSubDomains

SSLCipherSuite 'EECDH+aRSA+AESGCM:EECDH+aRSA+SHA384:EECDH+aRSA+SHA256:EDH
+CAMELLIA256:EECDH:EDH+aRSA:+SSLv3:!aNULL:!eNULL:!LOW:!3DES:!MD5:!EXP
:!PSK:!SRP:!DSS:!RC4:!SEED:!AES128:!CAMELLIA128:!ECDHE:AES256-SHA'
```

```
<VirtualHost *:80>
#...
RewriteEngine On
    RewriteRule ^.*$ https:// %{SERVER_NAME} %{REQUEST_URI} [L,R=
        permanent]
#...
</VirtualHost>
```



# TESTING

I FIND YOUR LACK OF TESTS DISTURBING.

DIY.DESPAIR.COM

# Testing

# How to test? - Tools

- `openssl s_client` (or `gnutls-cli`)
- [ssllabs.com](https://ssllabs.com): checks for servers as well as clients
- [xmpp.net](http://xmpp.net)
- `sslscan`
- `SSLyze`

# Tools: openssl s\_client

```
New, TLSv1/SSLv3, Cipher is ECDHE-RSA-AES256-GCM-SHA384
Server public key is 4096 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol : TLSv1.2
    Cipher   : ECDHE-RSA-AES256-GCM-SHA384
    Session-ID: 53D90B7D9D1FFC7EA98C105A2FC27F752B9CE9026CDAB57F4A7D4491C3C5ECC6
    Session-ID-ctx:
    Master-Key: 8F06DE9669BD6BF9628A38DF4F92C2CEBA6B7EA91F465164440CF31F7E8F55F2A67E7320B388D6E7AC4BC141C2FF3F68
    Key-Ag  : None
    PSK identity: None
    PSK identity hint: None
    SRP username: None
    TLS session ticket lifetime hint: 300 (seconds)
    TLS session ticket:
0000 - fe 5b 93 84 a8 c6 ab 4a-74 b8 59 81 dc 3e 52 40 .[.....Jt.Y..>R@
0010 - 0e dd f6 59 b4 a1 d2 54-65 df 9a 1b c9 fb 0d 2e ...Y...Te.....
0020 - 64 9c 65 cf 1c 0d d9 19-57 a6 cd 50 a5 d9 16 a4 d.e.....W..P....
0030 - 17 b6 e8 38 ac e5 76 15-a4 9d d5 62 ee 51 55 09 ...8..v....b.QU.
0040 - 52 36 58 84 04 0f 93 94-7b a9 dc e3 6f 8e 2f 7a R6X.....{...o./z
0050 - 9f bf 3d 4f a1 e1 bb 83-21 0f 7d f2 bd 02 48 a6 ..=0.....!..}..H.
0060 - 5a 96 82 fd dc a6 5a 55-77 b3 9f fb 60 0d 86 66 Z.....ZUw....`..f
0070 - f1 68 42 e2 90 93 8b f6-25 aa 85 cf 08 07 c6 76 .hB.....%.....v
0080 - 06 62 37 32 09 4f ac 23-28 9c db b9 29 c0 23 1b .b72.0.#(...).#.
0090 - e4 c3 d2 a3 a4 b4 87 b5-0e 5c 68 16 73 07 96 90 .....\\h.s...

Start Time: 1385118946
Timeout   : 300 (sec)
Verify return code: 21 (unable to verify the first certificate)
---
```

# Tools: ssllscan

```
ssllscan
Version 1.8.2
http://www.titania.co.uk
Copyright Ian Ventura-Whiting 2009

Testing SSL server git.bettercrypto.org on port 443

Supported Server Cipher(s):
Failed   SSLv2  168 bits  DES-CBC3-MD5
Failed   SSLv2  128 bits  IDEA-CBC-MD5
Failed   SSLv2  128 bits  RC2-CBC-MD5
Failed   SSLv2  128 bits  RC4-MD5
Failed   SSLv2  56 bits   DES-CBC-MD5
Failed   SSLv2  40 bits   EXP-RC2-CBC-MD5
Failed   SSLv2  40 bits   EXP-RC4-MD5
Failed   SSLv3  256 bits  ECDHE-RSA-AES256-GCM-SHA384
Failed   SSLv3  256 bits  ECDHE-ECDSA-AES256-GCM-SHA384
Failed   SSLv3  256 bits  ECDHE-RSA-AES256-SHA384
Failed   SSLv3  256 bits  ECDHE-ECDSA-AES256-SHA384
Rejected SSLv3  256 bits  ECDHE-RSA-AES256-SHA
Rejected SSLv3  256 bits  ECDHE-ECDSA-AES256-SHA
Rejected SSLv3  256 bits  SRP-DSS-AES-256-CBC-SHA
Rejected SSLv3  256 bits  SRP-RSA-AES-256-CBC-SHA
Failed   SSLv3  256 bits  DHE-DSS-AES256-GCM-SHA384
Failed   SSLv3  256 bits  DHE-RSA-AES256-GCM-SHA384
Failed   SSLv3  256 bits  DHE-RSA-AES256-SHA256
Failed   SSLv3  256 bits  DHE-DSS-AES256-SHA256
Rejected SSLv3  256 bits  DHE-RSA-AES256-SHA
Rejected SSLv3  256 bits  DHE-DSS-AES256-SHA
Rejected SSLv3  256 bits  DHE-RSA-CAMELLIA256-SHA
Rejected SSLv3  256 bits  DHE-DSS-CAMELLIA256-SHA
Rejected SSLv3  256 bits  AECDH-AES256-SHA
Rejected SSLv3  256 bits  SRP-AES-256-CBC-SHA
Failed   SSLv3  256 bits  ADH-AES256-GCM-SHA384
Failed   SSLv3  256 bits  ADH-AES256-SHA256
Rejected SSLv3  256 bits  ADH-AES256-SHA
Rejected SSLv3  256 bits  ADH-CAMELLIA256-SHA
```

# Tools: ssllabs

You are here: [Home](#) > [Projects](#) > [SSL Server Test](#) > [git.bettercrypto.org](#)

## SSL Report: git.bettercrypto.org (213.129.229.244)

Assessed on: Fri Nov 22 07:41:58 UTC 2013 | [Clear cache](#) [Scan Another »](#)

### Summary

Overall Rating A

	Score
Certificate	100
Protocol Support	95
Key Exchange	100
Cipher Strength	100

Documentation: [SSL/TLS Deployment Best Practices](#), [SSL Server Rating Guide](#), and [OpenSSL Cookbook](#).

This site works only in browsers with SNI support.

This server provides robust [Forward Secrecy](#) support.

# ssllabs (2)

## Configuration



### Protocols

TLS 1.2	Yes
TLS 1.1	Yes
TLS 1.0	Yes
SSL 3	No
SSL 2	No



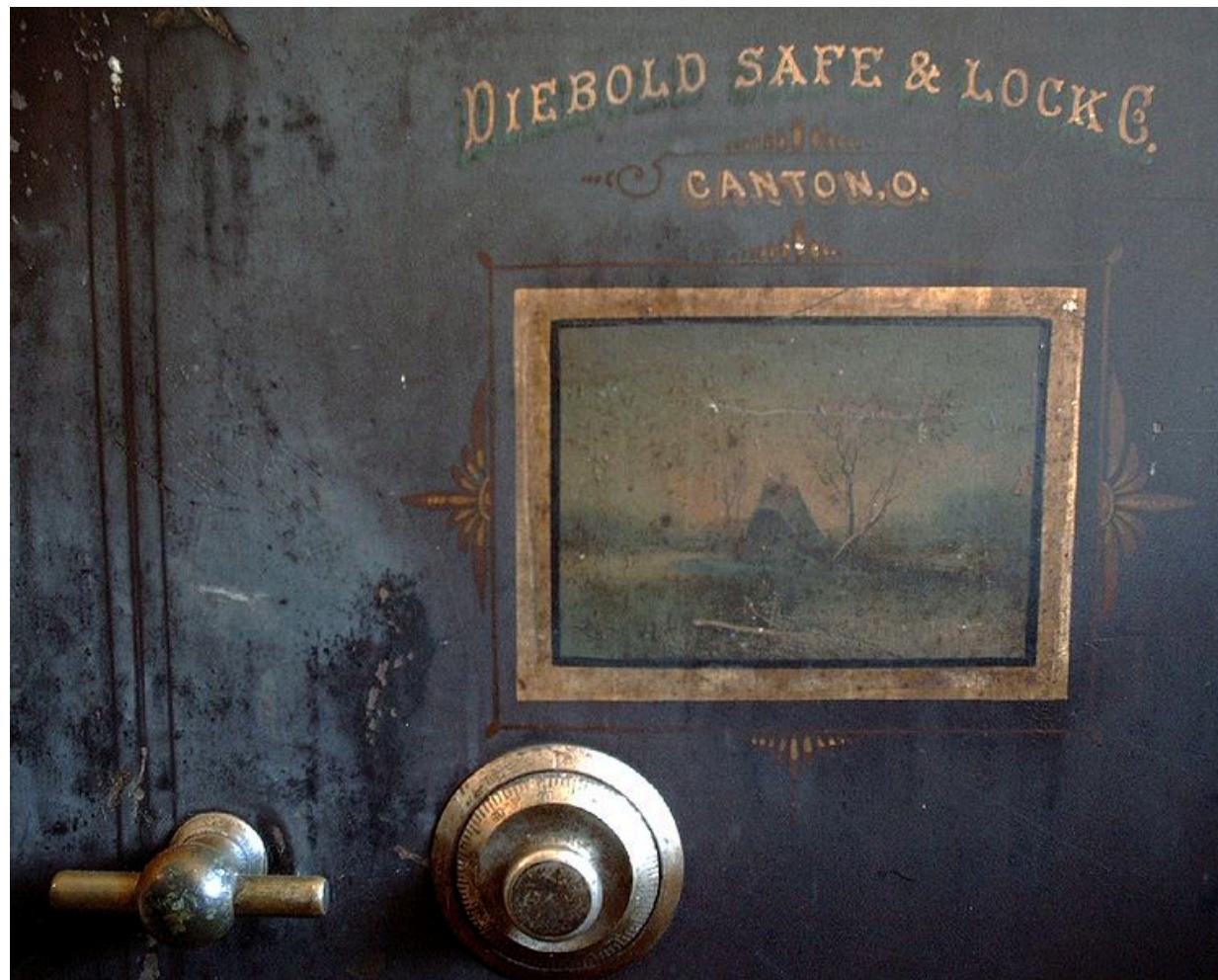
### Cipher Suites (SSL 3+ suites in server-preferred order, then SSL 2 suites where used)

TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)	ECDH 256 bits (eq. 3072 bits RSA)	FS	256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028)	ECDH 256 bits (eq. 3072 bits RSA)	FS	256
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (0x9f)	DH 4096 bits (p: 512, g: 1, Ys: 512)	FS	256
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 (0x6b)	DH 4096 bits (p: 512, g: 1, Ys: 512)	FS	256
TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88)	DH 4096 bits (p: 512, g: 1, Ys: 512)	FS	256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)	ECDH 256 bits (eq. 3072 bits RSA)	FS	256
TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x39)	DH 4096 bits (p: 512, g: 1, Ys: 512)	FS	256
TLS_RSA_WITH_AES_256_CBC_SHA (0x35)			256



## Handshake Simulation

<a href="#">Bing Oct 2013</a>	TLS 1.0	TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x39) FS	256
<a href="#">Chrome 31 / Win 7</a>	<b>TLS 1.2</b>	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	256
<a href="#">Firefox 10.0.12 ESR / Win 7</a>	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	256
<a href="#">Firefox 17.0.7 ESR / Win 7</a>	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	256
<a href="#">Firefox 21 / Fedora 19</a>	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	256
<a href="#">Firefox 24 / Win 7</a>	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	256
<a href="#">Googlebot Oct 2013</a>	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	256
<a href="#">IE 6 / XP No FS<sup>1</sup> No SNI<sup>2</sup></a>			Fail <sup>3</sup>
<a href="#">IE 7 / Vista</a>	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	256
<a href="#">IE 8 / XP No FS<sup>1</sup> No SNI<sup>2</sup></a>			Fail <sup>3</sup>
<a href="#">IE 8-10 / Win 7</a>	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	256
<a href="#">IE 11 / Win 7</a>	<b>TLS 1.2</b>	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	256
<a href="#">IE 11 / Win 8.1</a>	<b>TLS 1.2</b>	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	256
<a href="#">Java 6u45 No SNI<sup>2</sup></a>			Fail <sup>3</sup>
<a href="#">Java 7u25</a>			Fail <sup>3</sup>
<a href="#">OpenSSL 0.9.8y</a>	TLS 1.0	TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x39) FS	256
<a href="#">OpenSSL 1.0.1e</a>	<b>TLS 1.2</b>	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030) FS	256
<a href="#">Opera 17 / Win 7</a>	<b>TLS 1.2</b>	TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 (0x6b) FS	256
<a href="#">Safari 5.1.9 / OS X 10.6.8</a>	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	256
<a href="#">Safari 6 / iOS 6.0.1</a>	<b>TLS 1.2</b>	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028) FS	256
<a href="#">Safari 6.0.4 / OS X 10.8.4</a>	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	256
<a href="#">Safari 7 / OS X 10.9</a>	<b>TLS 1.2</b>	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028) FS	256
<a href="#">Tor 17.0.9 / Win 7</a>	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	256
<a href="#">Yahoo Slurp Oct 2013</a>	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	256



# Demo

# GPG - Ciphering

```
echo "This is a really secret" \
| gpg -a -u <your id>-r <his id> -e
```

```
-----BEGIN PGP MESSAGE-----
Version: GnuPG/MacGPG2 v2.0.22 (Darwin)
Comment: GPGTools - http://gpgtools.org

hQQMA5/oT2jaVoa7AR//bNn0Zfw8Ci+AmzPa0MjQDLxNIPTSaVa30/9fm692XpYu
ynp/iRUFhcq2DPRoFWq71V9BABAhzBzQA0rGZ6bZGTziVFkAfm4+79NvHGkE0aji
pf4CmlpyRE7yJWgIUd7TbIrKiC18yzA229YsC5QuTxqq+TFRiUpji/H3XVp/bTFu
tgHfAEagv5vdYCwlBmPuLc4oUSQ+zDoLEFw8AFaHaGMsCF/Pmq23k08uj6Ku7PJQ
FYd/HHmvQGlXrBuQXcZMml3q1W4sz4T2ZZW8HTF00J2s+0XwIK2rptHgU+f82Cw6
TFZmG9dLJoJeYr20YK3Yv0sIxj0hueqiNdTmVMTL5PiL96xXvIMmlRCRpQFOX0EL
iToSbl/M340jh0DN8t2NgI2XXqsN5lB1NZPyeTb84Uf8ZpBrkKC3gf8L8V+8B0C0
hDT/aMKhe2BoeaPUeA/+rxb4IUo7oYgPl8PLT8bJStAG8EwsIKI6yUXl6vNSYBqF
eLofD3YHWg03pk4wgaQ2u5xjPz7216MhSr/33Vel/DDwTd+yfUdT4SGXVmPqvEbC
hxW+Dq7W0WGwGr8Ed3yMIfrJcq7NEFwnz7DXyZR3LPdi1TYvvr+S8hbKomppo9S6
8uM5oBBD2LFnHPXgU7aWGGksC+jZFvrbNmeDGJWegaITEAe7rKvKM1joirHLNWYk
jnTySr+PKFHLwMMHCr8tfe2BTuRkdC4MQB0f3SL3ic0tLI1zUFpKXozN+H46EMbl
5o+rlgk+Kpb7UM5tbsBo+5E9e812TPp10fk63IMpUlxCd/IvQ4G4za3b60A/ev
DyRMs2wFaKKLV6g+cRUw8vmIbSFdBJlbodMc0wje8masAkk7kR/lM4IFZgu5v/xy
o3ne8khiz7v0Dq49s6GLBXC51YYhKnIDcaJzebDFpopp7y2DisjPWbGkWVqs/QH8
mihe/7TbaHH0bjmoFGPGb29IX2wuejYZE3Czm02J9rZ9dFFTijU7A0C9PnpzDT3p
Nde5iH4l4crMJ8GuXyh2rmwCXWN6BHnCeI69BIIs77yxP+CWOewIM2arPzYeIztNH
```

# GPG - Unciphering

- Let's save the ciphered text to msg.asc
- Then uncipher...

```
gpg -d msg.asc
```

```
You need a passphrase to unlock the secret key for
user: "David Durvaux <david.durvaux@belnet.be>"
8192-bit RSA key, ID DA5686BB, created 2010-12-04 (main key ID E84A32A0)
```

```
gpg: encrypted with 8192-bit RSA key, ID DA5686BB, created 2010-12-04
      "David Durvaux <david.durvaux@belnet.be>"
```

```
This is a really secret
```

# GPG - Signing

```
echo "This is a really secret" \
| gpg -a -u <your id>-r <his id>-s
```

```
You need a passphrase to unlock the secret key for
user: "David Durvaux <david.durvaux@cert.be>"
8192-bit RSA key, ID 57AB3358, created 2010-12-06
```

```
-----BEGIN PGP MESSAGE-----
```

```
Version: GnuPG/MacGPG2 v2.0.22 (Darwin)
Comment: GPGTools - http://gpgtools.org
```

```
owEBUASv+5ANAwACAcIpLB1XqzNYAcsgYgBThBnfVGhpcyBpcyBhIHJ1YWxseSBz
ZWNyZXQgIAqJBBwEAAECAYFA10EGd8ACgkQwiksHVerM1jHGiaA2LHHYXTKvSDJ
iKKA0Cc5P7YMRuodTYkHGPq3FGkR0k0yrXsFd3VwxyUoqWFodE0T675DhT/fJ2+t
eksFCrsY9jtq440QJdxTSahA2FwlwloE+1x27oAPtKdzKF0TMzGxwVa+wPnly51s
5Vj51n16oMMhigSLZNU/aPe1B8GLI9RXiKeLAv6SW65cq5CU2YhfMhnwKsLk3f1U
M/wXb7403NAqvCsRrdL8DWDxq7dvTmh1xFRJE53d5NxYI8l7K4SGwbIVs17e71a/
ZtPhrhac0WlYmwRXcS9qVewKt7Ri3DPdrsZ8CNVj1tJ9UcAE8fbe8RPlfJgVUF3i
Ai0IJJZZgWVoVdknAoFjXpH3rHvQWXZicSRLmU33K/yoE5WA0bSRQLMG18+EM1dc
MU4yFlUcRj0ZEkqAuwxgmFUeLiBF9fupqtBaJ+MiqctbFEZgbPamVcWfDS3ibt7G
3MGBDR/RLUvhDwqlg7nZxRVlEBPBNNB+rxT7sxDJTRNpqu2X67hITi9c4kI0n/jz
NEuhHNHyy6cJtAf2nDWI7i8NnMf5znGja3nJlZzvC7RDcwFbXswHSIXZXZJYIjt
CPdKB7u5Sxzap4+SksnxIW+9Ny8mY3vK+VkJ9FF42EfbpV007onJ3VwSegPZdsK8
kp670hHlwlb2c/GJTY0ZuoV+olpbbJwYwSS5rHYFoJUAMh5z2l0zKiNQSA3ZL0X7
SC63BrpK64sch0yJueEwlK1vVG5DWKXANV6x6rBEprzN6whc3+dYWe7ybF8ETuGD
T1oVnFTwkBPNE+PEHQpUSHCRKx0kSxWF+nUYbWIbsPpMPl+obr56oa0ErvR0CT53
```

# GPG – Check Signature

```
gpg --verify sig.asc
```

```
gpg: Signature made Tue May 27 06:51:43 2014 CEST using RSA key ID 57AB3358
gpg: Good signature from "David Durvaux <david.durvaux@cert.be>"
gpg:                               aka "David Durvaux <david@durvaux.be>"
gpg:                               aka "David Durvaux <david@durvaux.net>"
gpg:                               aka "David Durvaux <david.durvaux@gmail.com>"
gpg:                               aka "David Durvaux <ddurvaux@people.ops-trust.net>"
gpg:                               aka "David Durvaux <david.autopsit.org>"
```

Differents way to sign / verify:

<https://www.gnupg.org/gph/en/manual/x135.html>

Other techniques

- Clearsinged Documents
- Detached Signatures

```
gpg (GnuPG/MacGPG2) 2.0.22; Copyright (C) 2013 Free Software Foundation, Inc.  
This is free software: you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law.
```

```
Please select what kind of key you want:  
 (1) RSA and RSA (default)  
 (2) DSA and Elgamal  
 (3) DSA (sign only)  
 (4) RSA (sign only)  
Your selection? 1  
RSA keys may be between 1024 and 8192 bits long.  
What keysize do you want? (2048) 4096  
Requested keysize is 4096 bits  
Please specify how long the key should be valid.  
    0 = key does not expire  
  <n> = key expires in n days  
  <n>w = key expires in n weeks  
  <n>m = key expires in n months  
  <n>y = key expires in n years  
Key is valid for? (0) 90  
Key expires at Mon Aug 25 18:02:41 2014 CEST  
Is this correct? (y/N) y
```

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

```
Real name: Demo Key  
Email address: demo@localhost  
Comment: Demo  
You selected this USER-ID:  
  "Demo Key (Demo) <demo@localhost>"
```

```
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O  
You need a Passphrase to protect your secret key.
```

```
We need to generate a lot of random bytes. It is a good idea to perform  
some other action (type on the keyboard, move the mouse, utilize the  
disks) during the prime generation; this gives the random number  
generator a better chance to gain enough entropy.
```

```
We need to generate a lot of random bytes. It is a good idea to perform  
some other action (type on the keyboard, move the mouse, utilize the  
disks) during the prime generation; this gives the random number  
generator a better chance to gain enough entropy.
```

```
gpg: key 77659F3E marked as ultimately trusted  
public and secret key created and signed.
```

```
gpg: checking the trustdb  
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model  
gpg: depth: 0 valid: 7 signed: 21 trust: 0-, 0q, 0n, 0m, 0f, 7u  
gpg: depth: 1 valid: 21 signed: 30 trust: 21-, 0q, 0n, 0m, 0f, 0u  
gpg: next trustdb check due at 2014-08-11  
pub 4096R/77659F3E 2014-05-27 [expires: 2014-08-25]  
      Key fingerprint = 8D0B B43D 5F97 6AC0 66FF 1DEF DC5B 4FF1 7765 9F3E  
uid          Demo Key (Demo) <demo@localhost>  
sub 4096R/A77F0BAA 2014-05-27 [expires: 2014-08-25]
```

# GPG

# Key generation

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

- Kind of Key
- Keylength
- Expiration Period

# GPG – Key signing

```
gpg --sign-key -u <your ID> <his id>
```

```
pub 4096R/77659F3E created: 2014-05-27 expires: 2014-08-25 usage: SC
      trust: ultimate validity: ultimate
sub 4096R/A77F0BAA created: 2014-05-27 expires: 2014-08-25 usage: E
[ultimate] (1). Demo Key (Demo) <demo@localhost>

pub 4096R/77659F3E created: 2014-05-27 expires: 2014-08-25 usage: SC
      trust: ultimate validity: ultimate
Primary key fingerprint: 8D0B B43D 5F97 6AC0 66FF 1DEF DC5B 4FF1 7765 9F3E

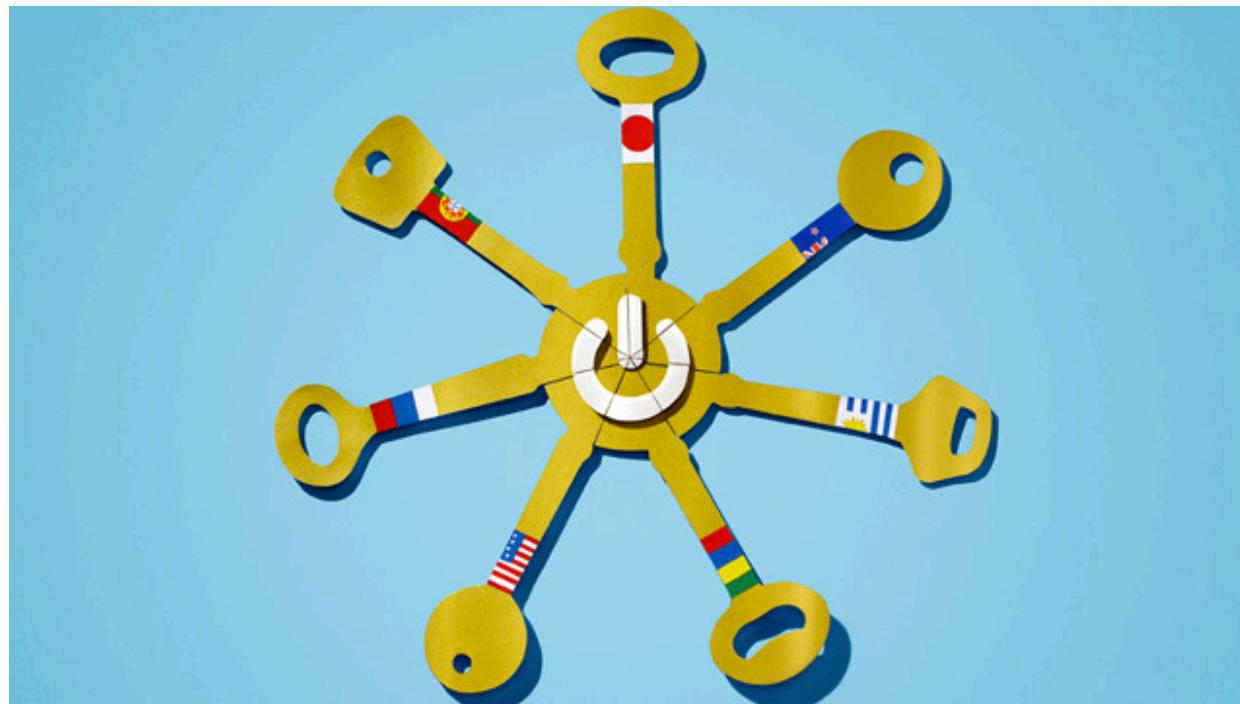
      Demo Key (Demo) <demo@localhost>

This key is due to expire on 2014-08-25.
Are you sure that you want to sign this key with your
key "David Durvaux <david.durvaux@belnet.be>" (E84A32A0)

Really sign? (y/N) y

You need a passphrase to unlock the secret key for
user: "David Durvaux <david.durvaux@belnet.be>"
8192-bit RSA key, ID E84A32A0, created 2010-12-04
```

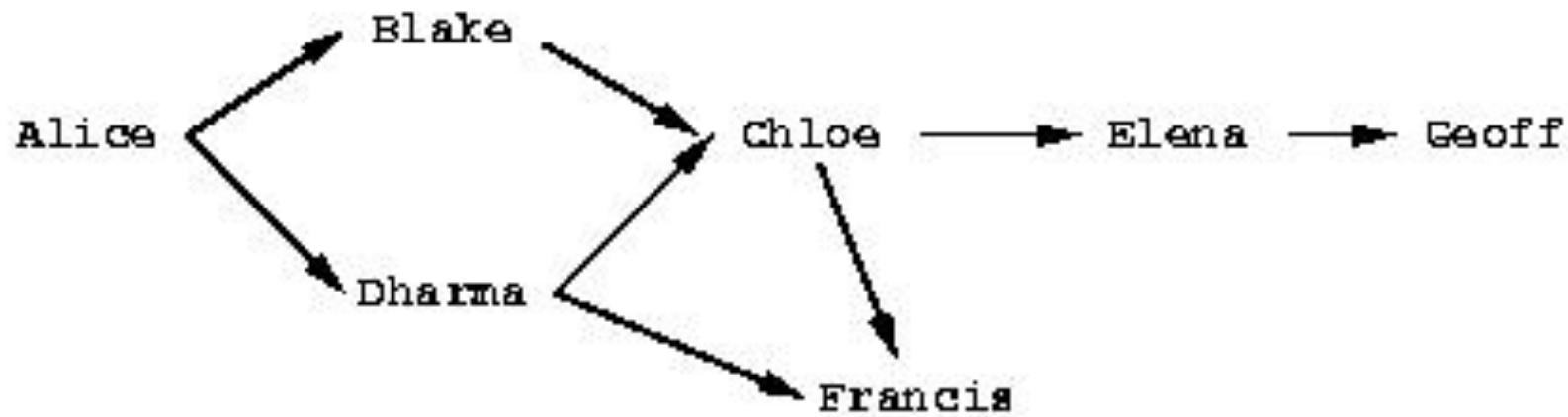
# GPG – Let's do it!



Let's do a key party!

# GPG – Sending key

```
gpg --send-keys <key id>
```



# GPG - Integration

- Enigmail (Thunderbird)
- GPGMail (Apple Mail)
- Symantec PGP
- ...

# Other nice user tools

- Ciphered containers:
  - TrueCrypt → might want to switch now?
  - Apple's FileVault2
- Password containers
  - KeePass
  - LastPass
- ...

## The Conclusion.

The Feavour works up towards Madness,  
and will scarcely endure to be touch'd.  
And what hope is there of Health when  
the Patient strikes in with the Disease,  
and flies in the Face of the Remedy? Can  
Religion retrieve us? Yes, when we don't  
despise it. But while our *Notions* are  
naught, our *Lives* will hardly be other-  
wise. What can the Assistance of the  
Church signify to those who are more  
than Preachers, than Practise

# Conclusion

# Futur / Idea

- Configuration Generator (online)
- Other tools
- Other protocols

# Current state as of 2014/05/31

- ✓ Solid basis with Variant (A) and (B)
- ✓ Public draft was widely presented at the CCC, RIPE meeting, IETF Strint workshop, Linuxdays, ..., M3AAWG
- Section „cipher suites“ still a bit messy, needs more work
- Need to convert to HTML

# How to participate

- We need: cryptologists, sysadmins, hackers
- Read the document, find bugs
- Subscribe to the mailing list
- Understand the cipher strings Variant (A) and (B) before proposing some changes
- If you add content to a subsection, make a sample config with variant (B)
- Git repo is world-readable
- We need:
  - Add content to an subsection from the TODO list  
→ send us diffs
  - **Reviewers!**

# Thank you!

- BetterCrypto.org
- <https://git.bettercrypto.org/ach-master.git>
- <http://lists.cert.at/cgi-bin/mailman/listinfo/ach>
- Contact
  - [kaplan@cert.at](mailto:kaplan@cert.at) — @kaplan\_CERTat
  - [david@autopsit.org](mailto:david@autopsit.org) — @ddurvaux