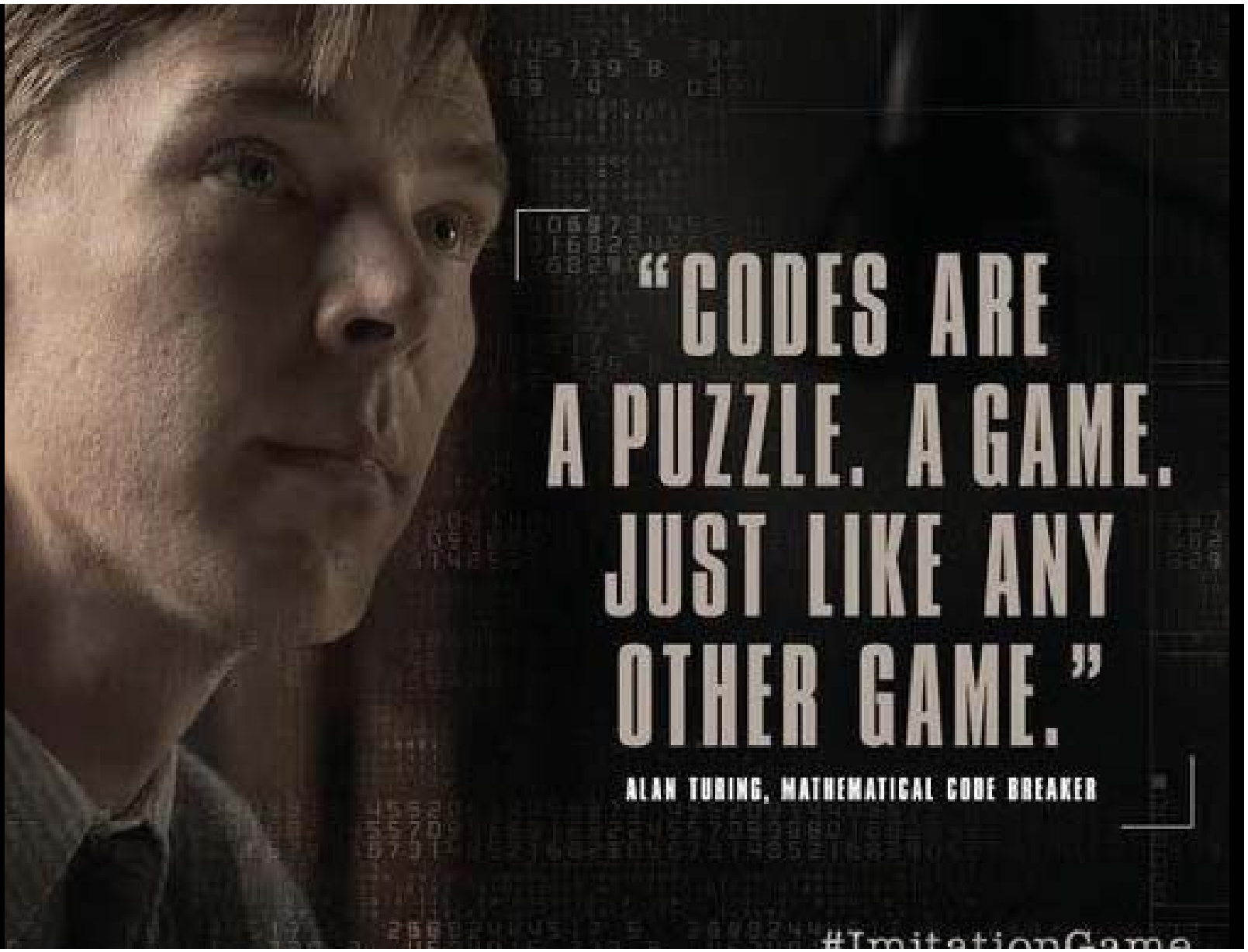

"EVERYONE THINKS ENIGMA IS UNBREAKABLE."
COMMANDER DENNISTON, HEAD OF GOVERNMENT CODE SCHOOL


"CODES ARE
A PUZZLE. A GAME.
JUST LIKE ANY
OTHER GAME."
ALAN TURING, MATHEMATICAL CODE BREAKER


"SOMETIMES IT IS THE PEOPLE NO ONE IMAGINES ANYTHING OF WHO DO THE THINGS THAT NO ONE CAN IMAGINE."
JOAN CLARKE, MATHEMATICIAN

BetterCrypto.org

Applied Crypto Hardening

David Durvaux
Aaron Kaplan
Aaron Zauner

FIRST.org -- Berlin, June 2(

Why better crypto?


NATIONAL SECURITY AGENCY
PEEPING WHILE YOU'RE SLEEPING
The NSA
The only part of government

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?

on’t give them anything for fre

It’s your home, your fight!

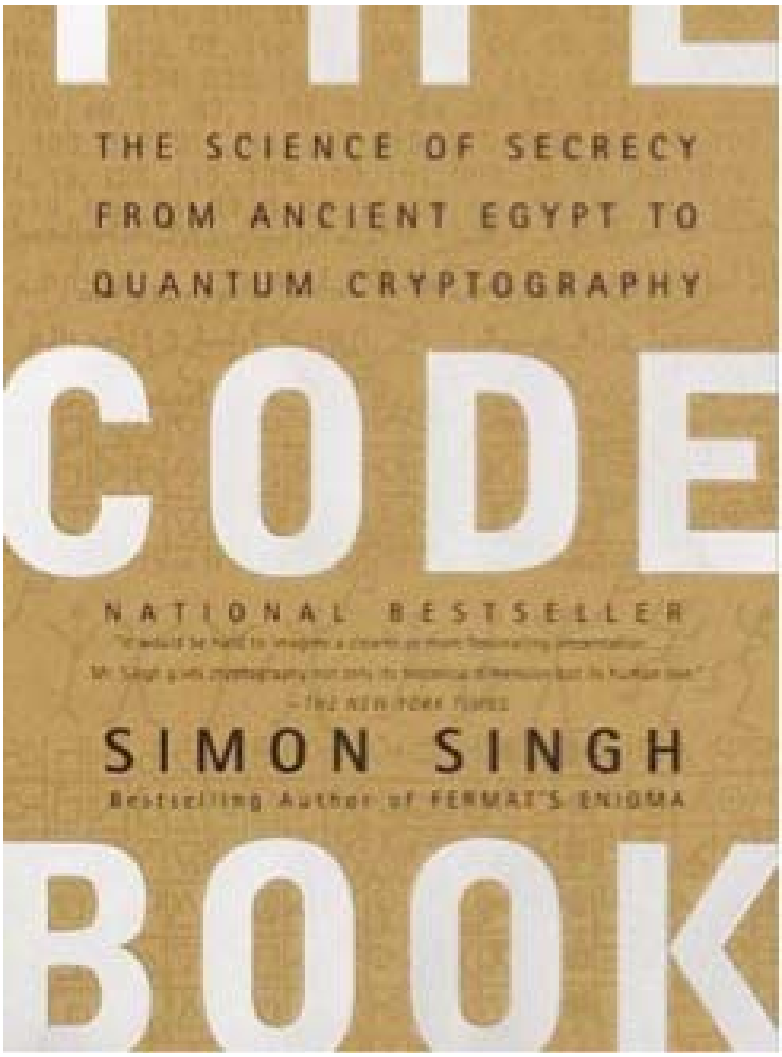
(authors of bettercrypto)

Agenda

- Part 1: BetterCrypto and the crypto world
- Part 2: When Thinks Goes Wrong...

Agenda – Part 1

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



Pieces of Historv

Historic ciphers

- Caesar Cipher
- Vigenère Cipher

Mary Queen of Scots

- Trial against Queen Elizabeth
- Was executed after code was broken (1

Enigma

- Secret in code book

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 BetterCrypo

That’s not…

- [ ] A crypto course
- [ ] A static document

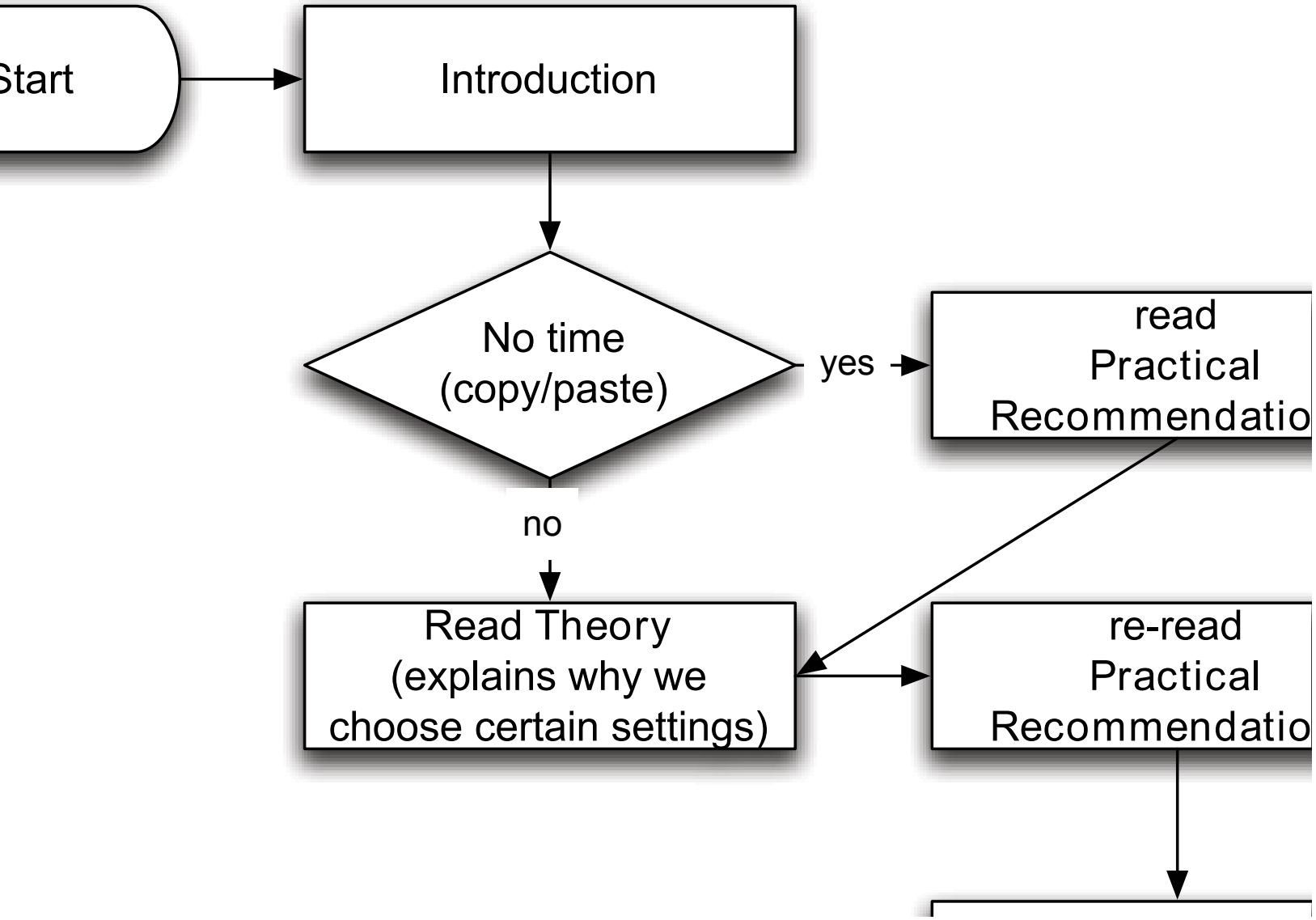
In brief

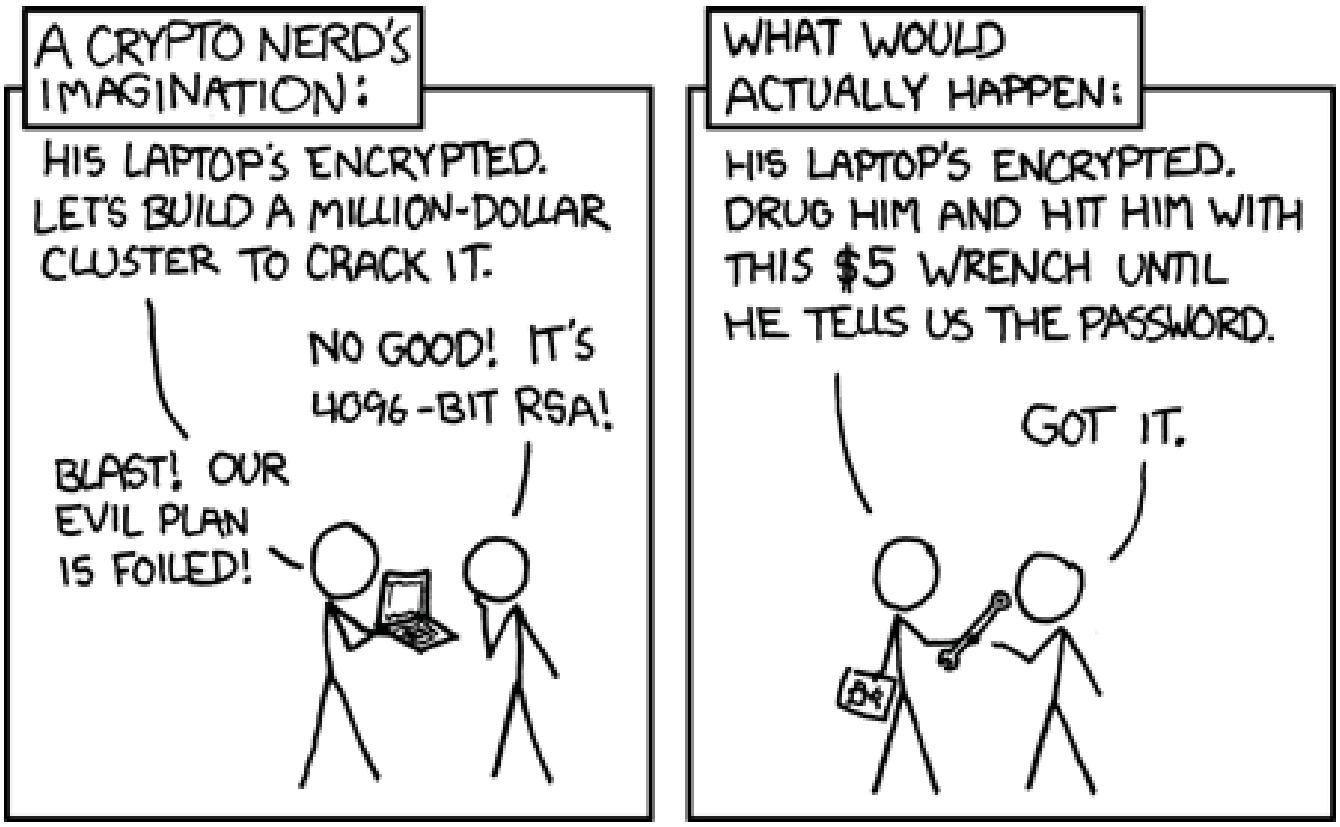
- Community effort to produce best common practices for typical servers
- Continuous effort
- From diverse areas of expertise: sysadmins, cryptologists, developers, IT security pros
- Open Source (CC-BY-SA)
- Open to comments / suggestions / improvements

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

ow to use the better crypto guid





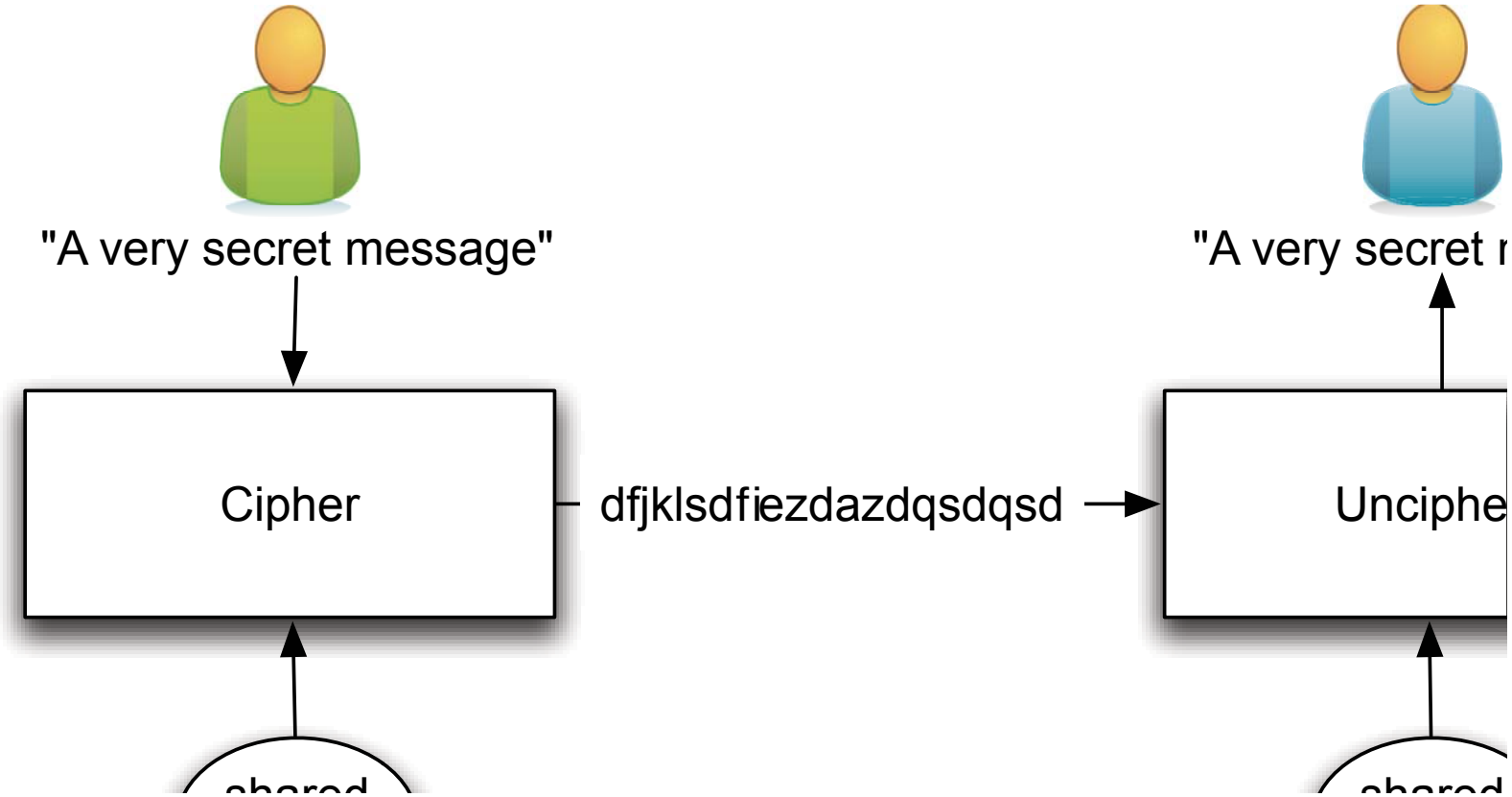
Crypto in a nutshell

Goals

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

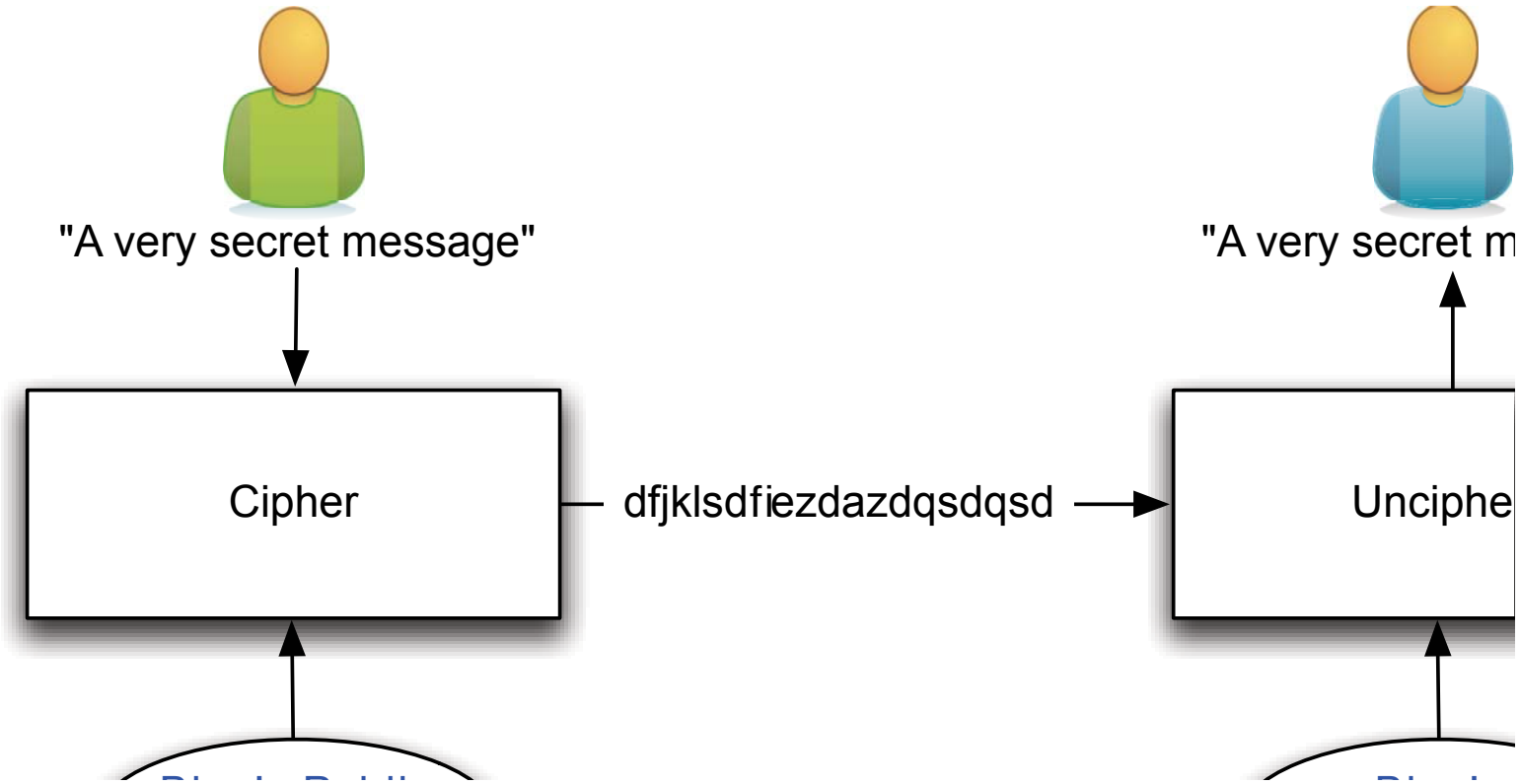
Symmetric Crypto

- The key is shared



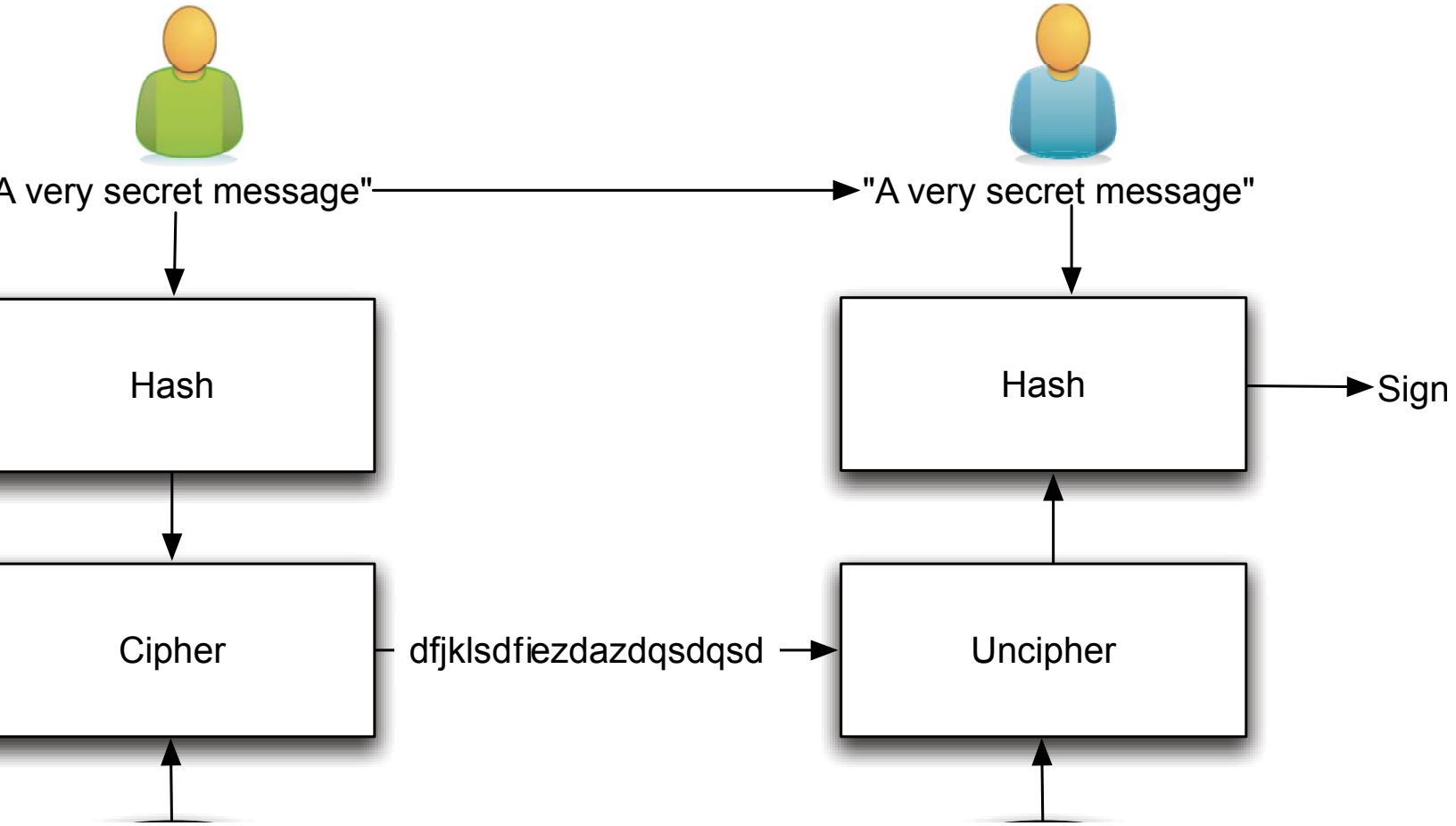
Asymmetric Crypto

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



Signing

- Author's identity is proved
  - Signed with the private key



The asymmetric magic

- RSA "formula" : $c = m^e \mod(n)$
  - with
    - c which is the ciphertext
    - m is the cleartext message
    - e and n are the public key
- Decipher with $m = c^d \mod(n)$
  - d being the private key

Diffie-Hellman

- How to share a secret key?

Secret: a

Public: p & g

send p & g

Secret

Public: p

g^b mod

g^a mod p = A

send A

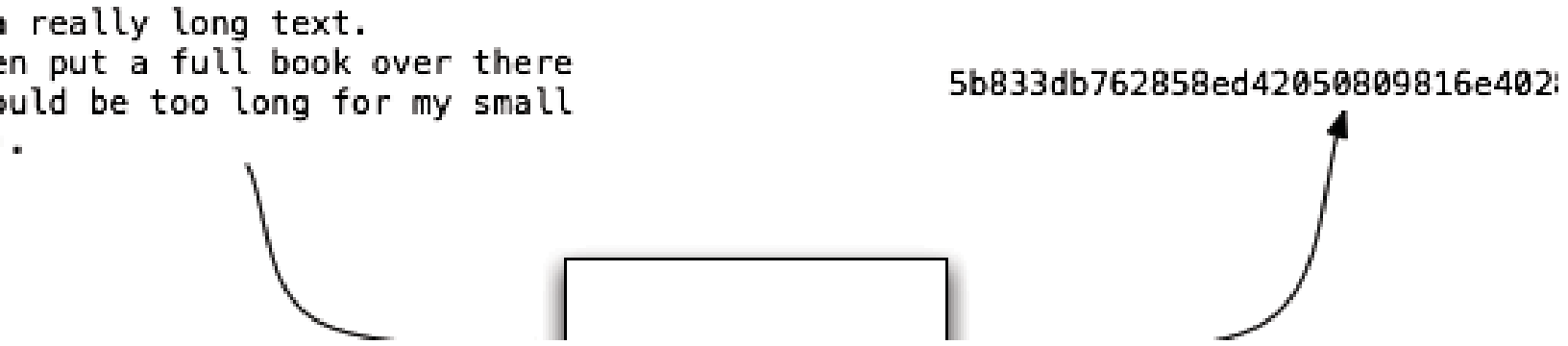
send B

Ephemeral Diffie-Helleman

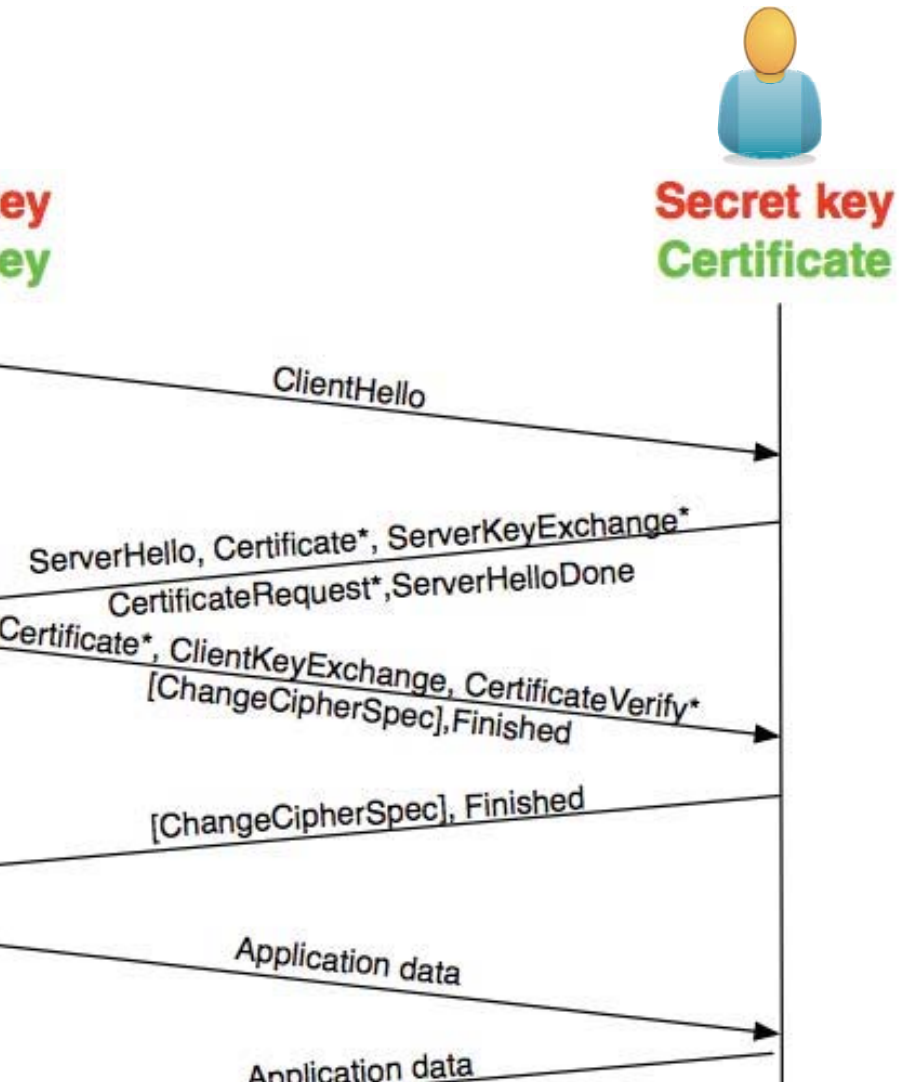
- Regular mode
  - Public and private keys are kept
- Ephemeral mode
  - New keys are generated each time
    - By both parties

Hashing

- Take long piece of data and produce a probably unique fingerprint
- Probability of collision for SHA1:
  - 1 over 1461501637330902918203684832716283019655932542976



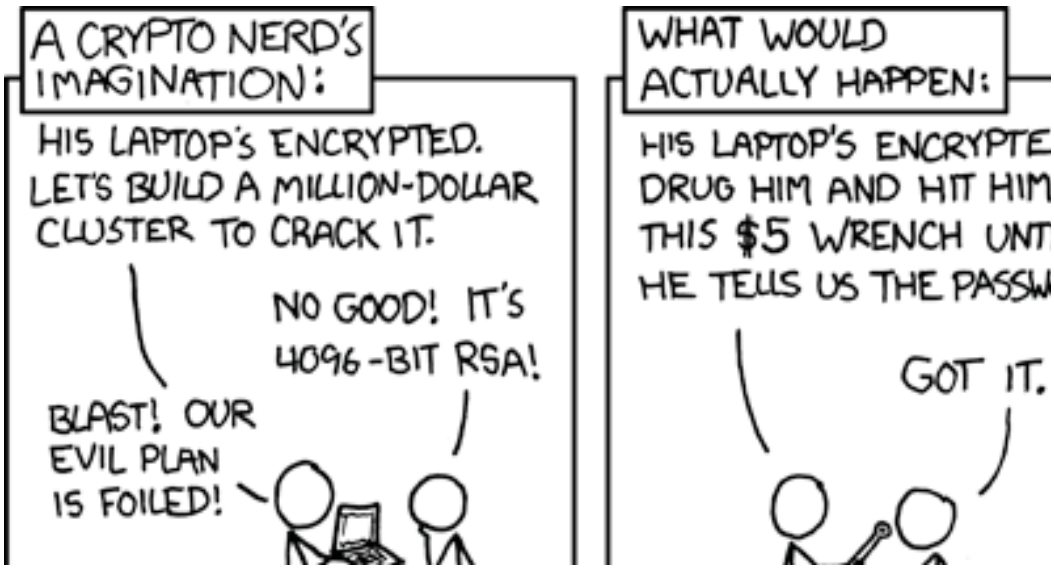
TLS



- Hello includes
  - Random number
  - Cipher suite
- Finished
  - 1st cipher message
  - negotiated parameter
    - Algorithm
    - Key

Forward Secrecy-Motivation

- [ ] Lavabit example
- [ ] 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:

Stream vs Block Cipher

- Stream cipher
  - Generate an “infinite” key stream
  - Difficult to correctly use
    - Re-use of keys
  - Faster
- Block cipher
  - Encrypt by block with padding

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

(p)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
- Asymetric Ciphering
  - RSA
  - PGP (GPG)

Some algorithms

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

Algorithm vs Implementation!

- Heartbleed
- Debian bug in Openssl (randomness was commented out)

Cost of encryption

```
me openssl enc -e -a -aes-128-cbc -in ./rfc791.t
 /tmp/rfc.aes -k "Super Key" -S 01EF

   0m0.014s

   0m0.004s

  0m0.003s


me gpg -a -u 57AB3358 -r 77659F3E -e ./rfc791.tx

   0m0.069s

   0m0.048s

   0m0.008s
```

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: >= 3248 bits (Ecrypt II)
  - ECC: >= 256
  - SHA 2+ (SHA 256,…)
  - AES 128 is good enough

ECRYPT II Recommendations (2012)
NIST Recommendations (2012)
ANSSI Recommendations (2010)
Fact Sheet NSA Suite B Cryptography (2013)
Network Working Group RFC3766 (2004)
BSI Recommendations (2014)

Compare all Methods

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: 256 bits

2 Compare

Method	Date	Symmetric	Asymmetric	Discrete Logarithm Key	Discrete Logarithm Group	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

lethod	Date	Symmetric	Asymmetric		Discrete Logarithm Key	Discrete Logarithm Group	Elliptic Cu
erheul	2014	81	1562	1216	143	1562	152
dated	2014	78	1218	1309	155	1218	155
	2011 - 2015	80	1248		160	1248	160
	2011 - 2030	112	2048		224	2048	224
	2010 - 2020	100	2048		200	2048	200
	-	-	-		-	-	-
	-	-	-		-	-	-
ure only)	2013 - 2015	-	1976		224	2048	224

BetterCrypto CipherSuite

- 2 cipher suites
  - version A
    - stronger
    - fewer supported clients
  - version B
    - weaker

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

OpenSSL Name	Version	KeyEx	Auth	Cipher
DHE-RSA-AES256-GCM-SHA384	TLSv1.2	DH	RSA	AESGCM(256)
DHE-RSA-AES256-SHA256	TLSv1.2	DH	RSA	AES(256) (CBC)

Ciper Suite B

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

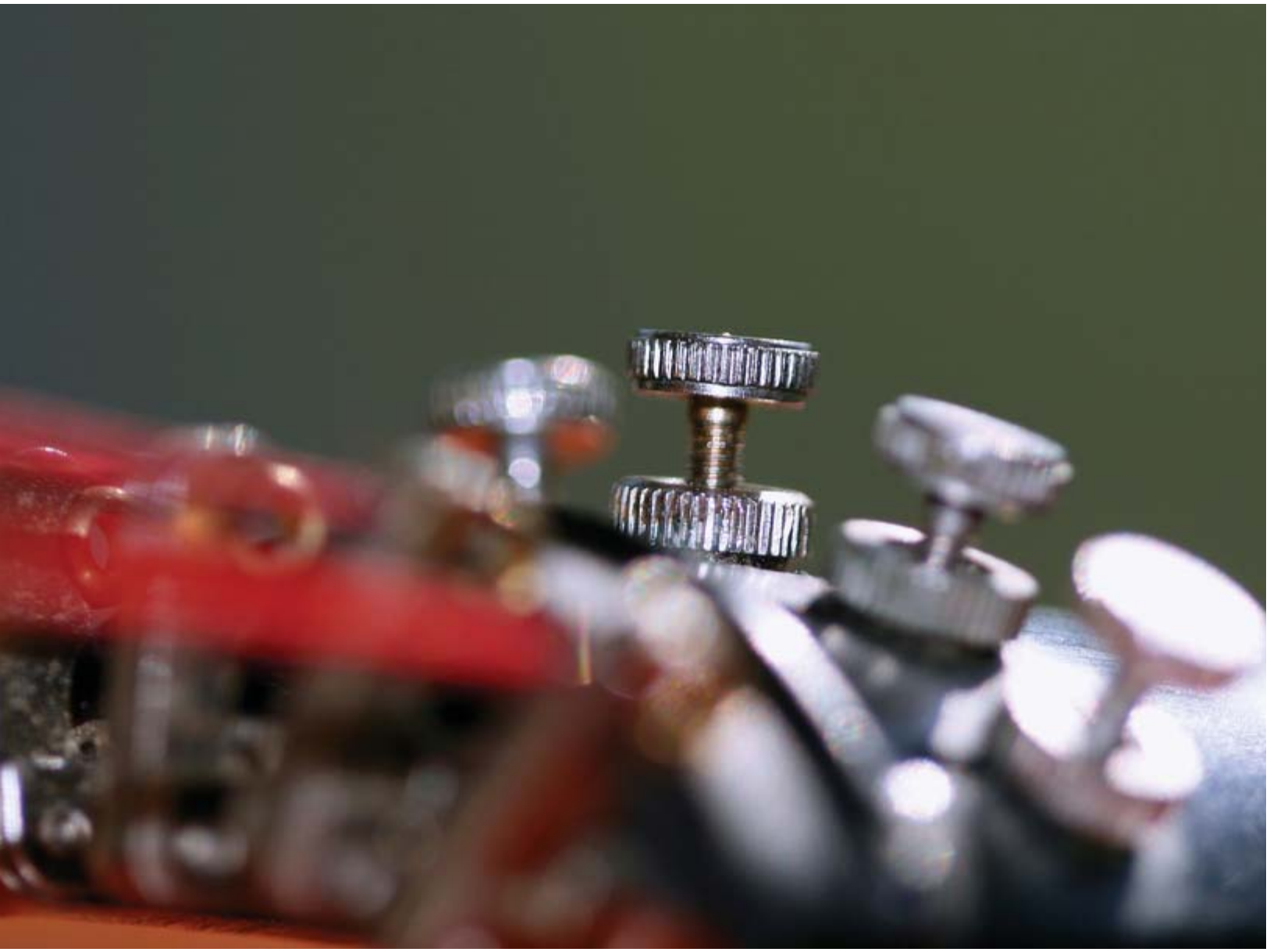
Cipher Suite B

	OpenSSL Name	Version	KeyEx	Auth	Cipher	M
9F	DHE-RSA-AES256-GCM-SHA384	TLSv1.2	DH	RSA	AESGCM(256)	AE
6B	DHE-RSA-AES256-SHA256	TLSv1.2	DH	RSA	AES(256)	SH
30	ECDHE-RSA-AES256-GCM-SHA384	TLSv1.2	ECDH	RSA	AESGCM(256)	AE
28	ECDHE-RSA-AES256-SHA384	TLSv1.2	ECDH	RSA	AES(256)	SH
9E	DHE-RSA-AES128-GCM-SHA256	TLSv1.2	DH	RSA	AESGCM(128)	AE
67	DHE-RSA-AES128-SHA256	TLSv1.2	DH	RSA	AES(128)	SH
2F	ECDHE-RSA-AES128-GCM-SHA256	TLSv1.2	ECDH	RSA	AESGCM(128)	AE
27	ECDHE-RSA-AES128-SHA256	TLSv1.2	ECDH	RSA	AES(128)	SH
88	DHE-RSA-CAMELLIA256-SHA	SSLv3	DH	RSA	Camellia(256)	SH
39	DHE-RSA-AES256-SHA	SSLv3	DH	RSA	AES(256)	SH
14	ECDHE-RSA-AES256-SHA	SSLv3	ECDH	RSA	AES(256)	SH
45	DHE-RSA-CAMELLIA128-SHA	SSLv3	DH	RSA	Camellia(128)	SH
33	DHE-RSA-AES128-SHA	SSLv3	DH	RSA	AES(128)	SH
13	ECDHE-RSA-AES128-SHA	SSLv3	ECDH	RSA	AES(128)	SH
84	CAMELLIA256-SHA	SSLv3	RSA	RSA	Camellia(256)	SH

Compatibility (B suite)

Handshake Simulation

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



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-1
  - OpenVPN
  - PPPTP
  - Cisco ASA
  - OpenSWAN

Tools covered

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

Tools covered

- Database systems
  - Oracle
  - MySQL
  - DB2
  - PostgreSQL

Tools covered

- Proxy
  - squid
  - Bluecoat
  - Pound
- Kerberos

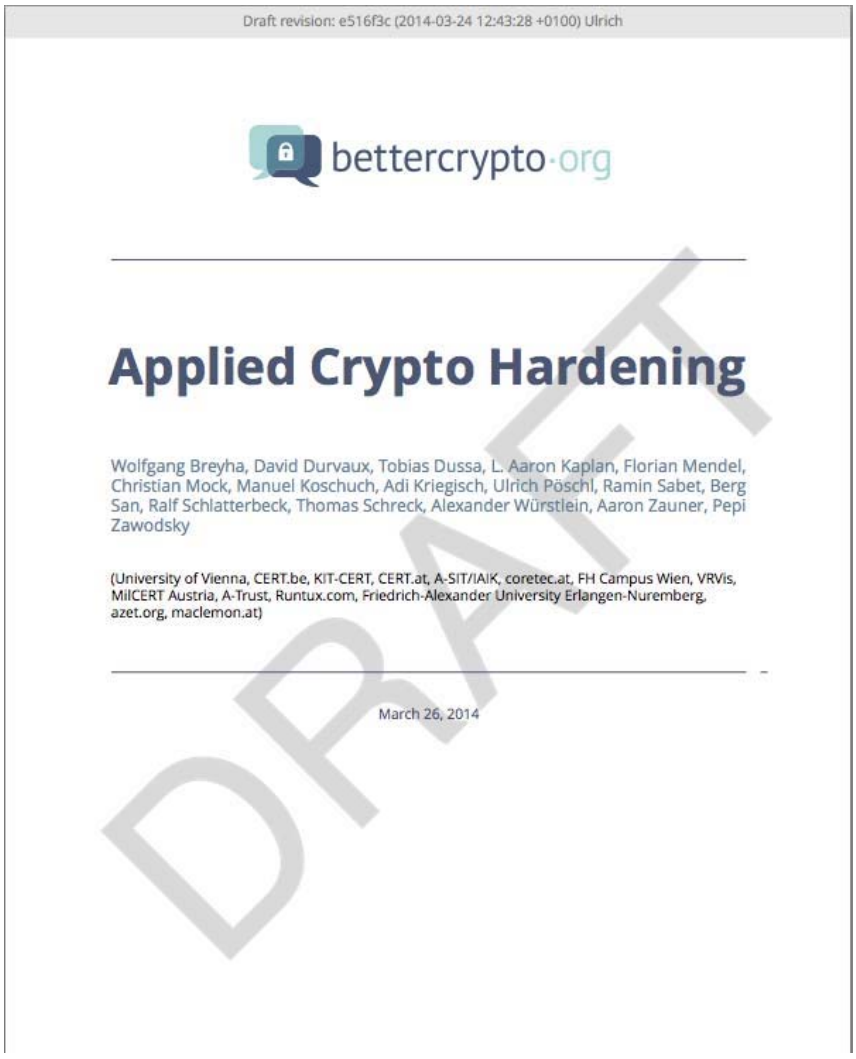
But…

- Microsoft products
  - MS Exchange
  - MS Lynx
  - …
- Other major vendors

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

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

Apache

Selecting cipher suites:

```
tocol All -SSLv2 -SSLv3
orCipherOrder On
pression off
six earth month HSTS header for all users...
 add Strict-Transport-Security "max-age=15768000"
ou want to protect all subdomains, use the following header
subdomains HAVE TO support https if you use this!
ct-Transport-Security: max-age=15768000 ; includeSubDomains

herSuite 'EECDH+aRSA+AESGCM:EECDH+aRSA+SHA384:EECDH+aRSA+SHA
MELLIA256:EECDH:EDH+aRSA:+SSLv3:!aNULL:!eNULL:!LOW:!3DES:!M
SK:!SRP:!DSS:!RC4:!SEED:!AES128:!CAMELLIA128:!ECDSA:AES256-
```

Additionally:

```
alHost *:80>

teEngine On
RewriteRule ^.*$ https://%{SERVER_NAME}%{REQUEST_URI} [L,R=
    permanent]
```

Mail Server

- SMTP make use of opportunistic TLS
- 3 modes for mailservers
  - Mail Submission Agent (MSA)
  - Receiving Mail Transmission Agent (MX)
  - Sending Mail Transmission Agent (SMTP client)

Mail Server

- [ ] Correct DNS configuration without CNAMEs
- [ ] Enable encryption
- [ ] NO self-signed certificates

SMTP client mode

- Hostname used as HELO must match the PTR RR
- Setup a client certificate
- Common name or alternate subject name must match the PTR RR
- Don‘t touch cipher suite

MSA

- [ ] Listen on port 587
- [ ] Enforce SMTP AUTH
- [ ] No SMTP AUTH on unencrypted connections
- [ ] (use recommended cipher suites)

Postfix: MX & SMTP client

- In main.cf
  - Enable opportunistic TLS

```
rameters
s_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
s_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
for Postfix >= 2.9, and 1 for earlier versions
s_loglevel = 0
 opportunistic TLS support in the SMTP server and client
s_security_level = may
_security_level = may
_loglevel = 1
 have authentication enabled, only offer it after STARTTLS
s_auth_only = yes
```

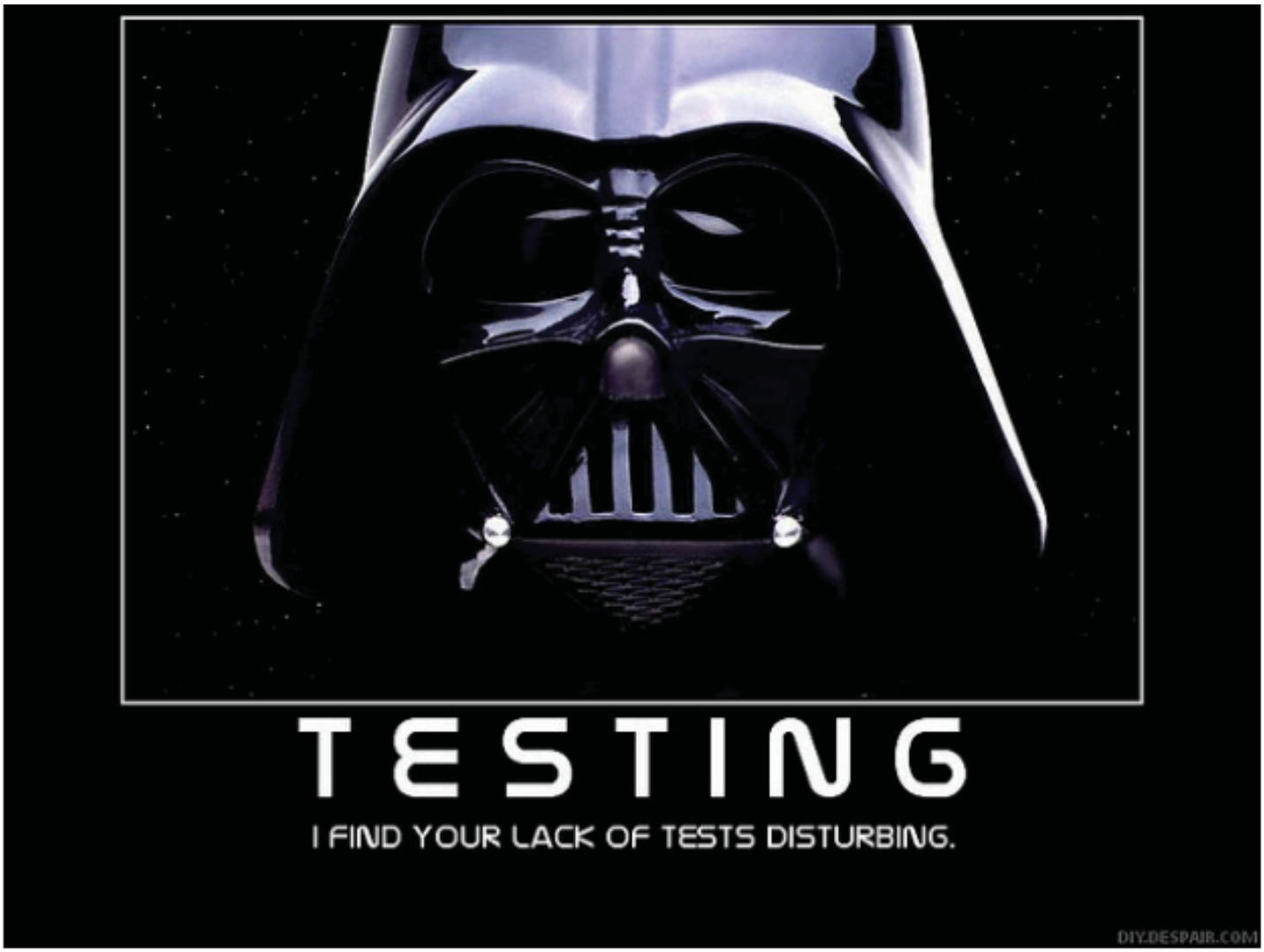
Postfix: MSA

- [ ] Define ciper suite:

```
tls_mandatory_protocols = !SSLv2, !SSLv3
tls_mandatory_ciphers=high
gh_cipherlist=EDH+CAMELLIA:EDH+aRSA:EECDH+aRSA+AESGCM:EECDH+aRSA+SHA38
DH+aRSA+SHA256:EECDH:+CAMELLIA256:+AES256:+CAMELLIA128:+AES128:+SSLv3:
NULL:!LOW:!3DES:!MD5:!EXP:!PSK:!DSS:!RC4:!SEED:!ECDSA:CAMELLIA256-SHA:
A:CAMELLIA128-SHA:AES128-SHA
```

- [ ] Configure MSA SMTP:

```
sion inet n - - - - smtpd
tpd_tls_security_level=encrypt
s_preempt_cipherlist=yes
```

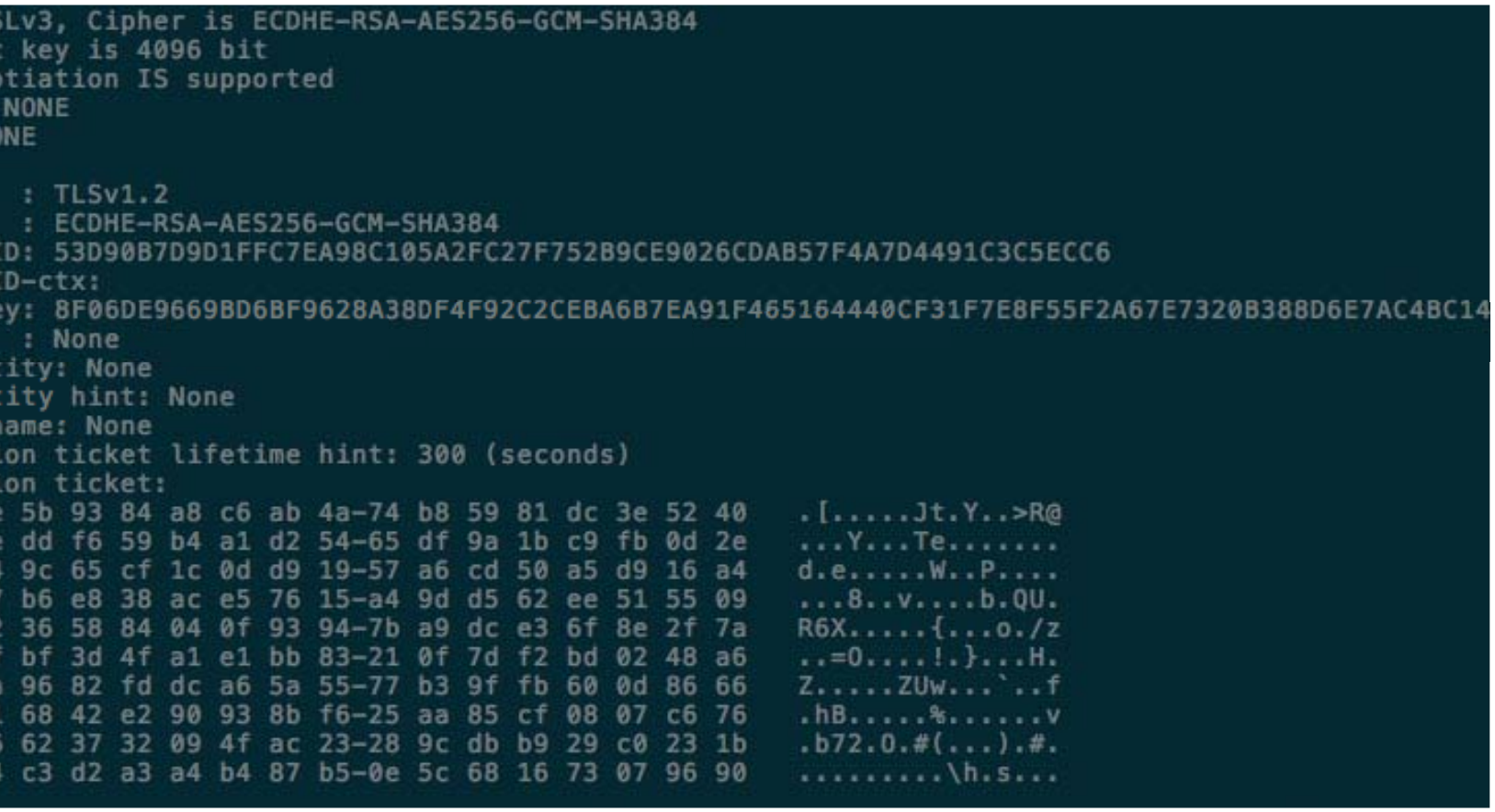

TESTING
I FIND YOUR LACK OF TESTS DISTURBING.
DIY.DESPAIR.COM

Testing

How to test?- Tools

- openssl s_client (or gnutls-cli)
- ssllabs.com: checks for servers as well as clients
- xmpp.net
- sslscan
- SSLyze

Tools: openssl s_client



Tools: sslscan

Tools: ssllabs

QUALYS® SSL LABS

Home Qualys.com Projects Contact

You are here: Home > Projects > SSL Server Test > git.bettercrypto.org

SSL Report: git.bettercrypto.org (213.129.229.244)

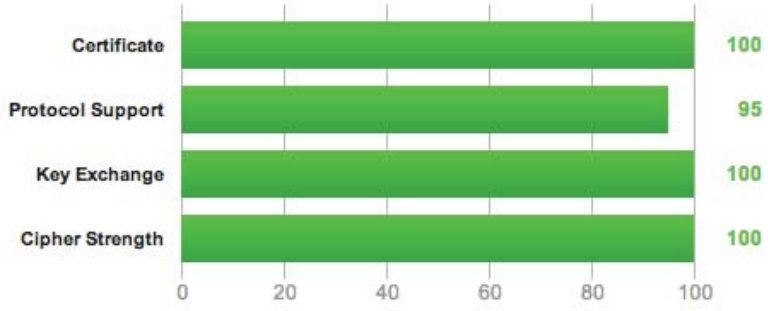
Assessed on: Fri Nov 22 07:41:58 UTC 2013 | Clear cache

Scan Another »

Summary

Overall Rating

A



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)

onfiguration

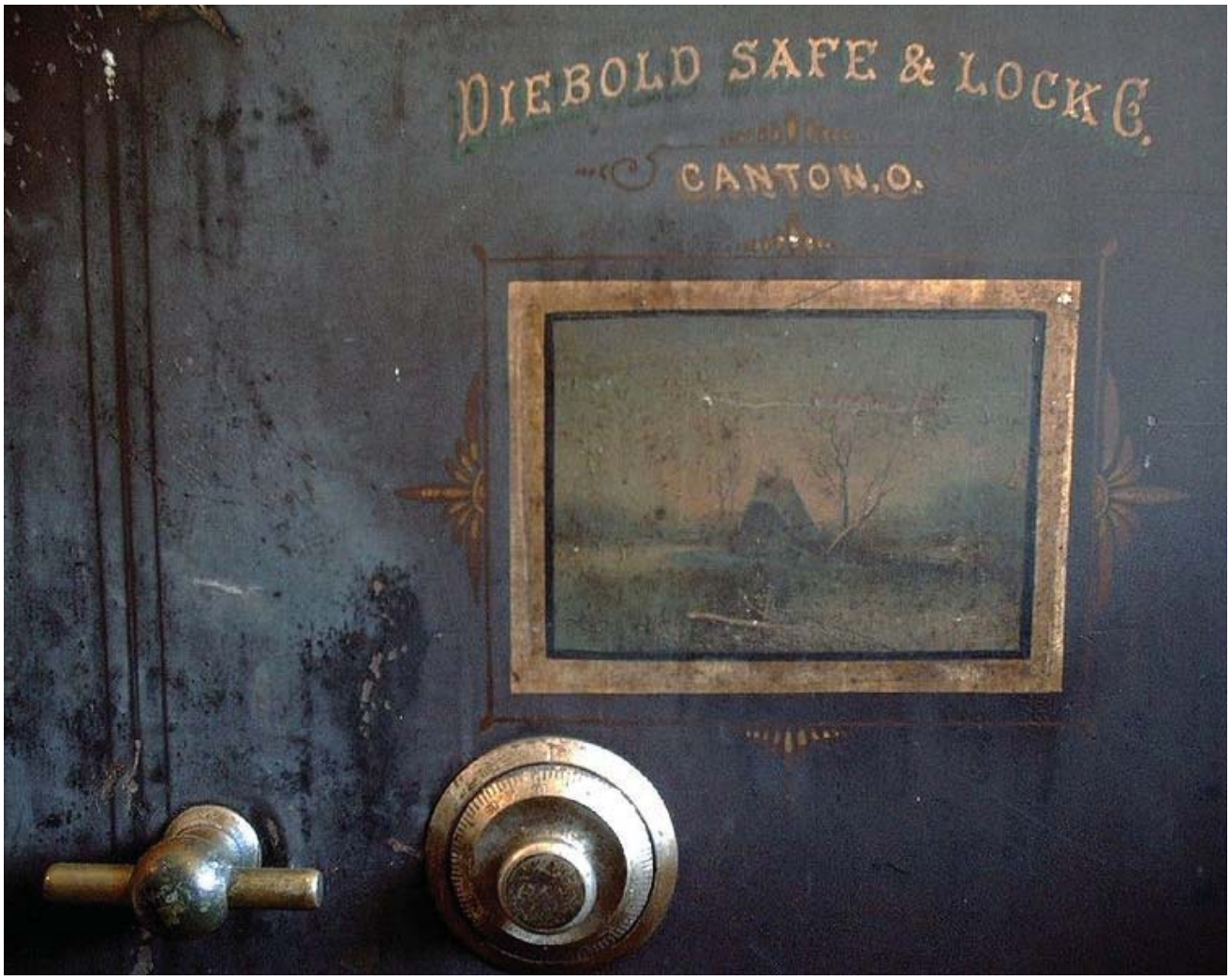
Protocols

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

Chrome 31 / Win 7	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	25
Firefox 10.0.12 ESR / Win 7	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	25
Firefox 17.0.7 ESR / Win 7	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	25
Firefox 21 / Fedora 19	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	25
Firefox 24 / Win 7	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88) FS	25
Googlebot Oct 2013	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	25
IE 6 / XP No FS [1] No SNI [2]			Fai
IE 7 / Vista	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	25
IE 8 / XP No FS [1] No SNI [2]			Fai
IE 8-10 / Win 7	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	25
IE 11 / Win 7	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	25
IE 11 / Win 8.1	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	25
Java 6u45 No SNI [2]			Fai
Java 7u25			Fai
OpenSSL 0.9.8y	TLS 1.0	TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x39) FS	25
OpenSSL 1.0.1e	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030) FS	25
Opera 17 / Win 7	TLS 1.2	TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 (0x6b) FS	25
Safari 5.1.9 / OS X 10.6.8	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	25
Safari 6 / iOS 6.0.1	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028) FS	25
Safari 6.0.4 / OS X 10.8.4	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) FS	25



Demo

GPG - Encryption

```
o "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//bNnOZfw8Ci+AmzPa0MjQDLxNIPTSaVa30/9fm692XpYu
ynp/iRUFhcq2DPRoFWq71V9BABehzBzQAOrGZ6bZGTziVFkAfm4+79NvHGkEOaji
pf4CmlpyRE7yJWgIUd7TbIrKiC18yzA229YsC5QuTxqq+TFRiUpji/H3XVp/bTFu
tgHfAEagv5vdYCwlBmPuLc4oUSQ+zDoLEFw8AFaHaGMsCF/Pmq23kO8uj6Ku7PJQ
FYd/HHmvQGlXrBuQXcZMml3q1W4sz4T2ZZW8HTFO0J2s+OXwIK2rptHgU+f82Cw6
TFZmG9dLJoJeYr2OYK3YvOsIxjOhueqiNdTmVMTL5PiL96xXvIMmlRCDpQFOXOEL
iToSbl/M340jhODN8t2NgI2XXqsN5lBlNZPyeTb84Uf8ZpBrkKC3gf8L8V+8BOC0
hDT/aMKhe2BoeaPUeA/+rxb4IUo7oYgPl8PLT8bJStAG8EwsIKI6yUXl6vNSYBgF
eLofD3YHWg03pk4wgaQ2u5xjPz7216MhSr/33Vel/DDwTd+yfUdT4SGXVmPqvEbC
hxW+Dq7W0WGwGr8Ed3yMIfRJcq7NEFwnz7DXyZR3LPdi1TYvvr+S8hbKomppo9S6
8uM5oBBD2LFnHPXgU7aWGGksC+jZFvrbNmeDGJWegaITEAe7rKvKM1joiRHLNWYk
jnTySr+PKFHLwMMHCr8tfe2BTuRkdC4MQB0f3SL3ic0tLI1zUFpKXozN+H46EMbl
5o+rlgk+Kpb7UM5tbsBo+5E9e812TPp1Ofkc63IMpUlxUCD/IvQ4G4za3b60A/ev
DyRMs2wFaKKLV6g+cRUw8vmIbSFdBJlbodMcOwje8masAkk7kR/lM4IFZgu5v/xy
```

GPG - Decryption

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

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

```
need a passphrase to unlock the secret key for
: "David Durvaux <david.durvaux@belnet.be>"
-bit RSA key, ID DA5686BB, created 2010-12-04 (main key ID E84A

 encrypted with 8192-bit RSA key, ID DA5686BB, created 2010-12-
  "David Durvaux <david.durvaux@belnet.be>"
 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+5ANAwACAcIpLB1XqzNYAcsgYgBThBnfVGhpcyBpcyBhIHJlYWxseSBz
ZWNyZXQgIAqJBBwEAAECAAYFAlOEGd8ACgkQwiksHVerM1jHGiAA2LHHYXTKvSDJ
iKKAOCc5P7YMRuodTYkHGPq3FGkR0k0yrXsFd3VwxyUoqWFodEOT675DhT/fJ2+t
eksFCrsY9jtq44OQJdxTSahA2FwlwloE+1x27oAPtKdzKF0TMzGxwVa+wPnly51s
5Vj51n16oMMhigSLZNU/aPe1B8GLI9RXiKeLAy6SW65cq5CU2YhfMhnwKsLk3flU
M/wXb7403NAqvCsRrdL8DWDxq7dvTmh1xFRJE53d5NxYI8l7K4SGwbIVs17e71a/
ZtPhrhacOWlYmwRXcS9qVewKt7Ri3DPdrsZ8CNVj1tJ9UcAE8fbe8RPlfJgVUF3i
AiOIJJZZgWVoVdknAoFjXpH3rHvQWXZicSRLmU33K/yoE5WAObSRQLMG18+EM1dc
MU4yFlUcRj0ZEkqAuwxgmFUeLiBF9fupqtBaJ+MiqctbFEZgbPamVcWfDS3ibt7G
3MGBDR/RLUvhDwqlg7nZxRVlEBPBNNB+rxT7sxDJTRNpqu2X67hITi9c4kIQn/jz
NFubH7NHvv6cJtAf2nDWT7i8NnMf5znGia3nJJ7zvC7RDcwFbXswHSTX7XZJYTit
```

GPG – Check Signature

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

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

ifferents way to sign / verify:

ttps://www.gnupg.org/gph/en/manual/x135.html

ther techniques

- Clearsigned Documents

```
Elgamal
n only)
n only)
 1
 between 1024 and 8192 bits long.
 you want? (2048) 4096
ze is 4096 bits
how long the key should be valid.
y does not expire
y expires in n days
y expires in n weeks
y expires in n months
y expires in n years
r? (0) 90
Mon Aug 25 18:02:41 2014 CEST
? (y/N) y

construct a user ID to identify your key.

 Key
demo@localhost

is USER-ID:
Demo) <demo@localhost>"

(C)omment, (E)mail or (O)kay/(Q)uit? O
phrase to protect your secret key.

rate a lot of random bytes. It is a good idea to perform
on (type on the keyboard, move the mouse, utilize the
he prime generation; this gives the random number
ter chance to gain enough entropy.
rate a lot of random bytes. It is a good idea to perform
on (type on the keyboard, move the mouse, utilize the
he prime generation; this gives the random number
ter chance to gain enough entropy.
3E marked as ultimately trusted
et key created and signed.

he trustdb
(s) needed, 1 complete(s) needed, PGP trust model
valid:   7  signed:  21  trust: 0-, 0q, 0n, 0m, 0f, 7u
valid:  21  signed:  30  trust: 21-, 0q, 0n, 0m, 0f, 0u
```

Key generatio

`gpg --gen-key`

- Kind of Key
- Keylength
- Expiration Perio

GPG – Key signing

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

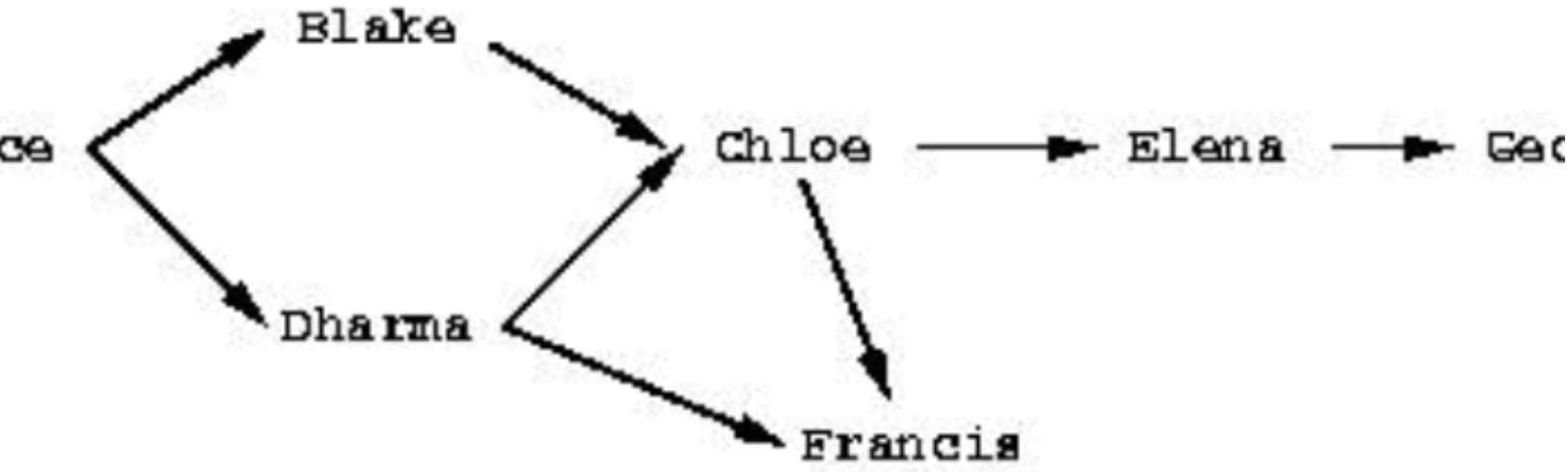
```
pub  4096R/77659F3E  created: 2014-05-27  expires: 2014-08-25  usage: SC
                     trust: ultimate      validity: ultimate
sub  4096R/A77F0BAA  created: 2014-05-27  expires: 2014-08-25  usage: E
[ultimate] (1). Demo Key (Demo) <demo@localhost>


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 9F3

     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
```

GPG – Let's do it!

Let's do a key party!

GPG – Sending key

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

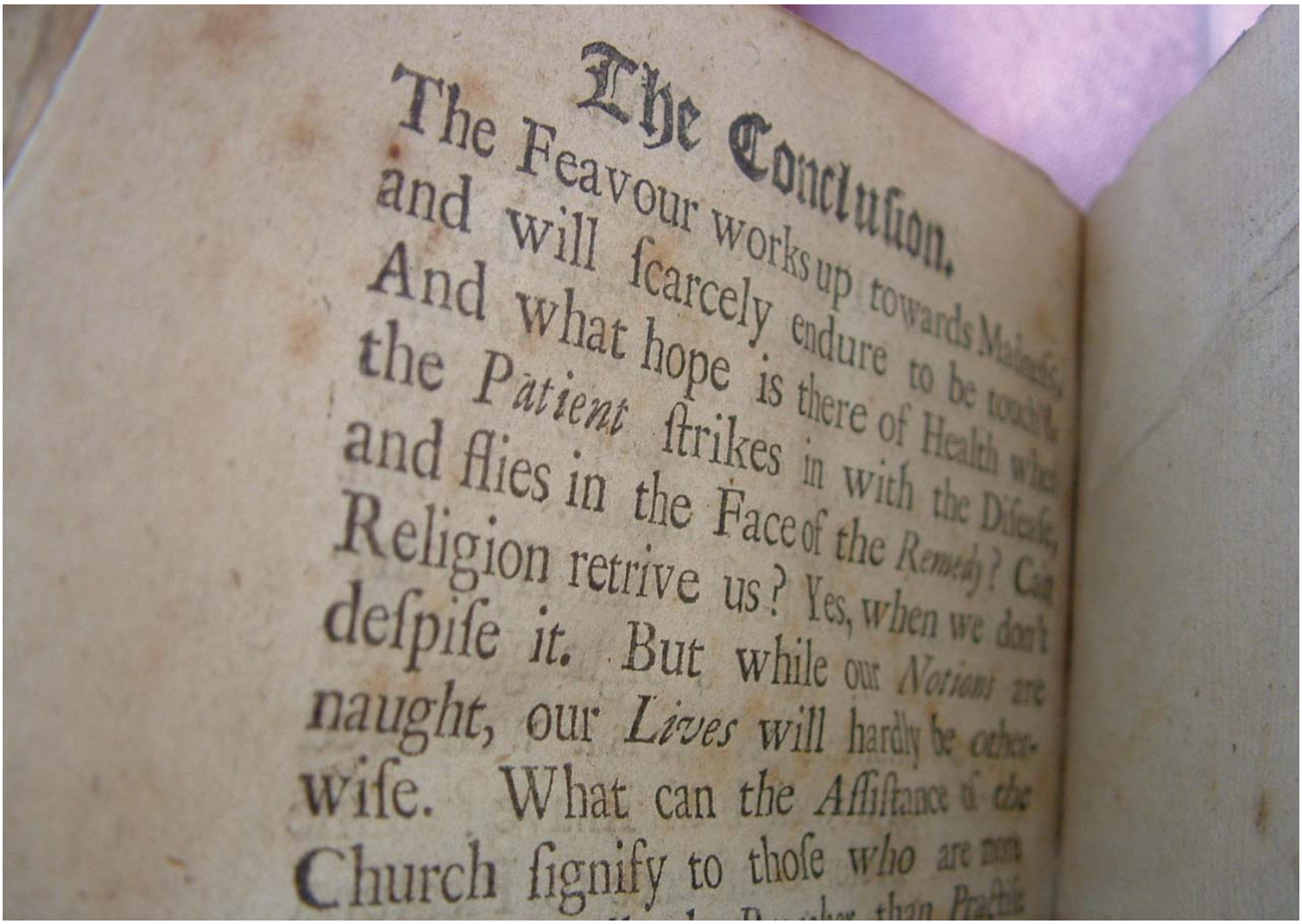


GPG - Integration

- 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.

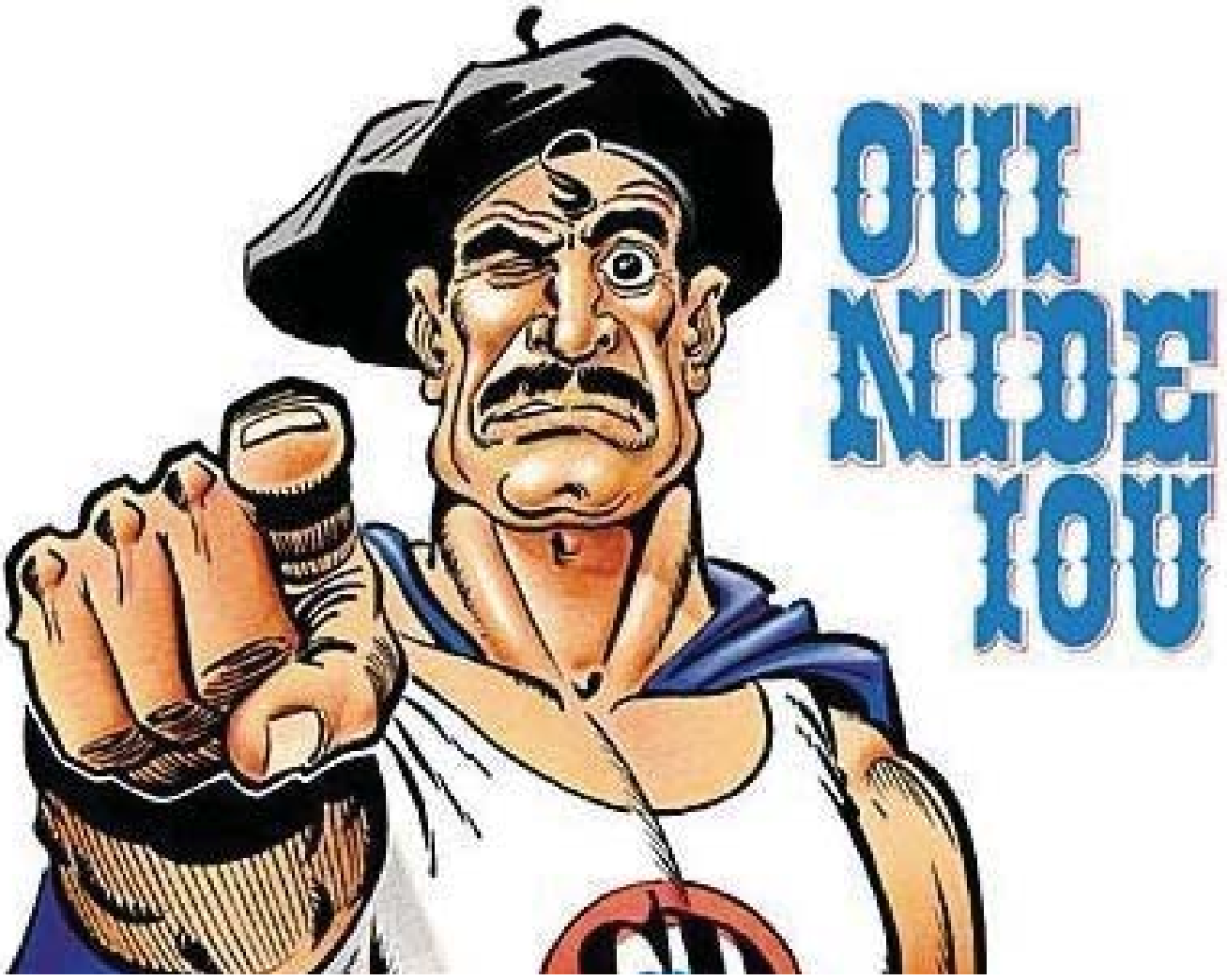
The Feavour works up towards Madness,
and will ſcarcely endure to be touch'd.
And what hope is there of Health when
the *Patient* ſtrikes in with the Diſeaſe,
and flies in the Face of the *Remedy*? Can
Religion retrive us? Yes, when we don't
deſpiſe it. But while our *Notions* are
naught, our *Lives*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

Conclusion

Future ideas

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

But . . .



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

Thank you!

BetterCrypto.org

https://git.bettercrypto.org/ach-master.git

http://lists.cert.at/cgi-bin/mailman/listinfo/ach

Contact

- david@autopsit.org — @ddurvaux
- aaron@lo-res.org — @KaplanAaron

More?

The asymmetric magic

- RSA "formula" : $c = m^e \mod(n)$
  - with
    - c which is the ciphertext
    - m is the cleartext message
    - e and n are the public key
- Decipher with $m = c^d \mod(n)$
  - d being the private key

Heartbleed

```
/* Enter response type, length and copy
*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…

ECC

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

Some thoughts on ECC

Currently this is under heavy debate

Trust the Math

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