Project „ACH“ (Applied Crypto Hardening)

www.bettercrypto.org

Motivation

NATIONAL SECURITY AGENCY
PEEPING WHILE YOU'RE SLEEPING
The NSA
The only part of government
that actually listens.

Don‘t give them anything for free

It‘s your home, you fight

TL;DR - Quickinfos

- Website: www.bettercrypto.org
- Git repo: https://git.bettercrypto.org
- Mailing list: http://lists.cert.at/cgi-bin/mailman/listinfo/ach
- Jabber chat: `ach@conference.jabber.metalab.at`

Why is this relevant for you?

- You run networks and services. These are targets. It seems that even sysadmins are targets (source: ZDNet/the leaks)

- However good crypto is hard to achieve
- Crypto does not solve all problems, **but it helps**

"The Bottom Line Is That Encryption Does Work“, Edward Snowden

Who?

Wolfgang Breyha (uni VIE), David Durvaux (CERT.be), Tobias Dussa (KIT-CERT), L. Aaron Kaplan (CERT.at), 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), Tobias Pape

New contributors: IAIK, A-Sit

Idea

- Do at least something against the **Cryptocalypse**
- Check SSL, SSH, PGP crypto Settings in the most common services and certificates:
  - Apache, Nginx, lighthttp
  - IMAP/POP servers (dovecot, cyrus, ...)
  - openssl.conf
  - Etc.
- Create **easy, copy & paste-able settings** which are „OK“ (as far as we know) for **sysadmins.**
- Keep it short. There are many good recommendations out there written by cryptographers for cryptographers
- **Many eyes must check this!**

Contents so far

- Disclaimer
- Methods
- Elliptic Curve Cryptography
- Keylengths
- Random Number Generators
- Cipher suites – general overview & how to choose one
- Recommendations on practical settings
- Tools
- Links

Methods

- How we develop this whitepaper
- Public review
- We need your review!

GENERAL REMARKS ON CRYPTO

Some thoughts on ECC

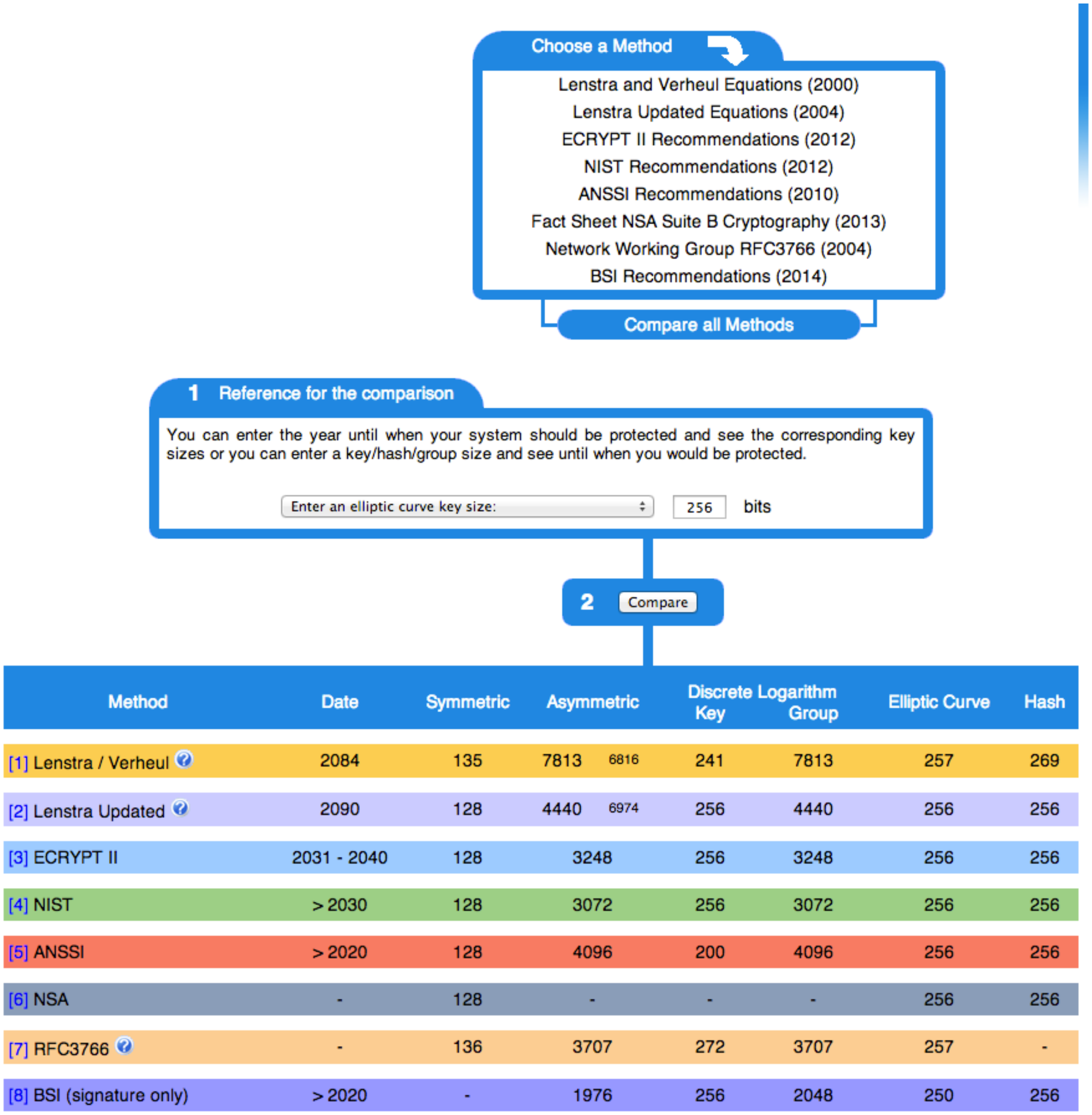
- Currently this is under heavy debate
- Trust the Math
- “Nothing Up My Sleeve Numbers”
  - eg. NIST P-256 (http://safecurves.cr.yp.to/rigid.html)
  - `Coefficients generated by hashing the unexplained seed c49d3608 86e70493 6a6678e1 139d26b7 819f7e90.`
- Might have to change settings tomorrow
- Most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

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

AES 128? Isn‘t that enough?

- “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
- Some theoretical attacks on AES-256



Choose a Method

Lenstra and Verheul Equations (2000)
Lenstra Updated Equations (2004)
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.

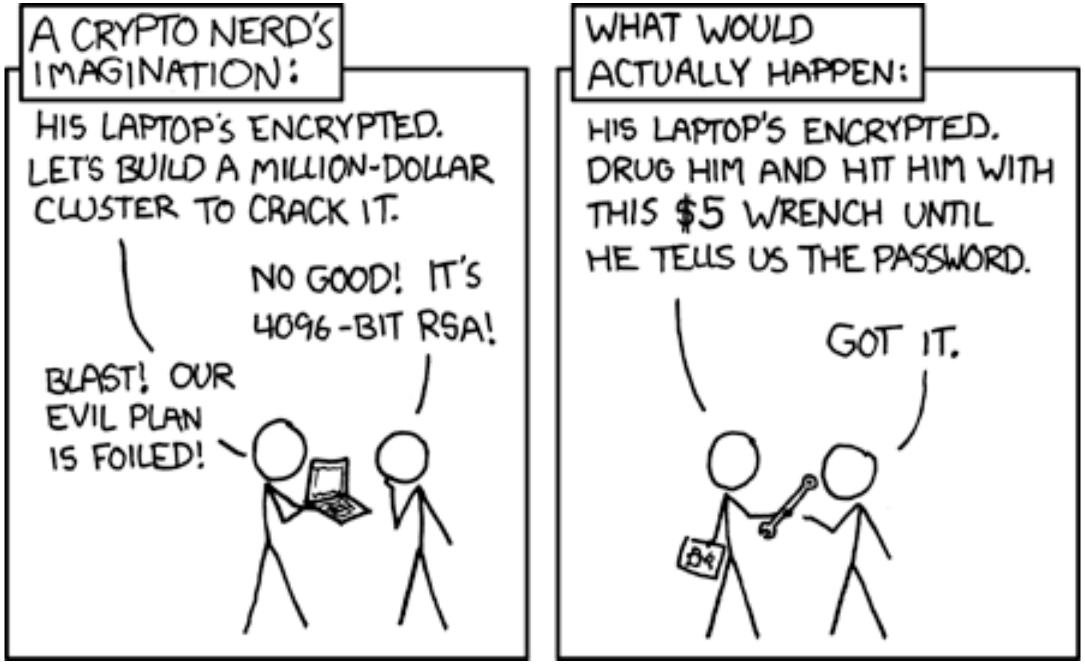
Enter an elliptic curve key size: 256 bits

2 Compare

Method	Date	Symmetric	Asymmetric		Discrete Logarithm		Elliptic Curve	Hash
					Key	Group		
[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

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

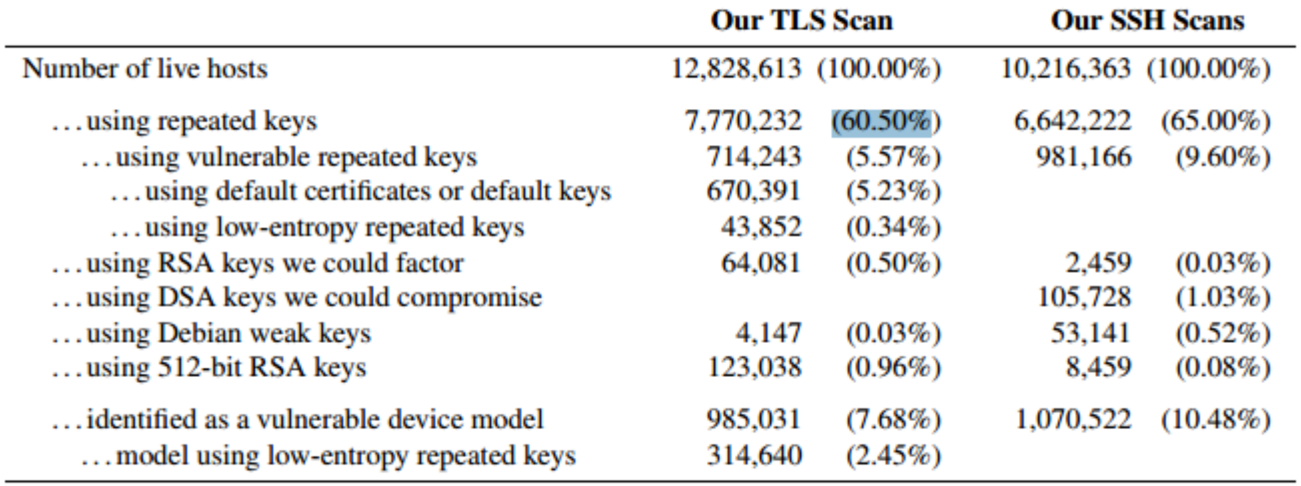

A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.

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

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

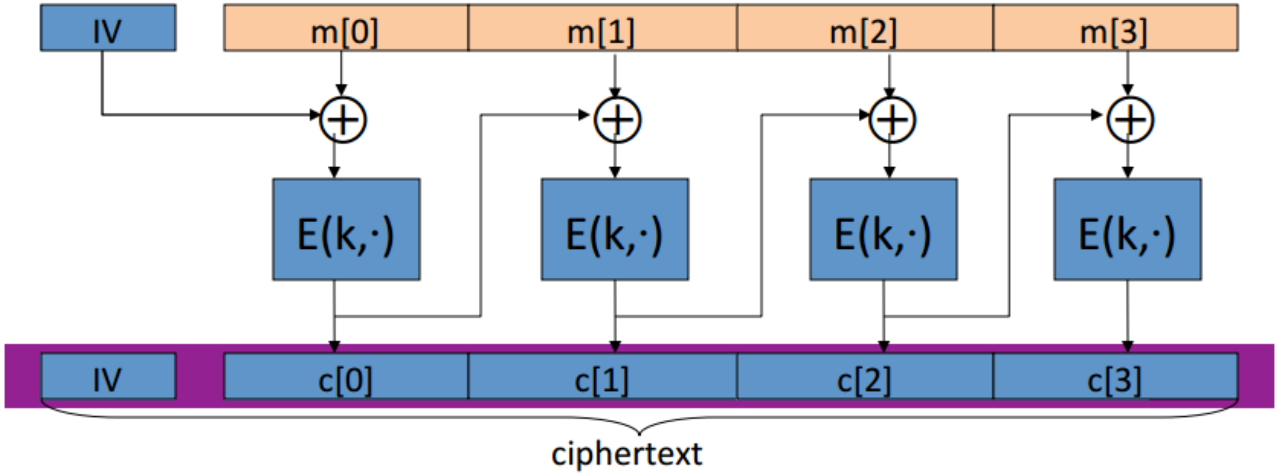
RNGs

- Weak RNG
  - Dual EC_DRBG is weak (slow, 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

ATTACKS

Attacks - BEAST

- Browser Exploit Against SSL/TLS (***BEAST***) attack
  - Predict IV of CBC

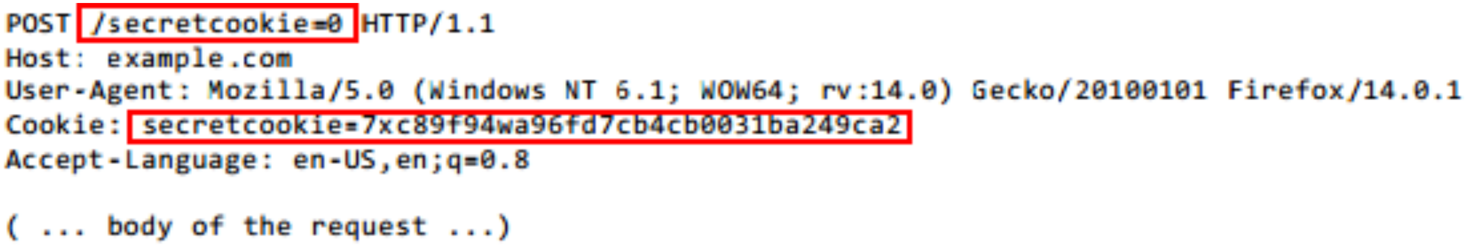


– Subsequent packet use IV that is the last cyphertext block of the previous packet

– Chosen Plaintext Attack (eg. Cookie-name)

Attacks - CRIME

- Compression Ratio Info-leak Made Easy (CRIME) attack
  - Sidechannel attack
  - Information based on compressed size of http requests
  - MITM, Bruteforce: Client Javascript to Browse to …



```
POST /secretcookie=0 HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:14.0) Gecko/20100101 Firefox/14.0.1
Cookie: secretcookie=7xc89f94wa96fd7cb4cb0031ba249ca2
Accept-Language: en-US,en;q=0.8

( ... body of the request ...)
```

  - Compressed size smaller when secretcookie correct.

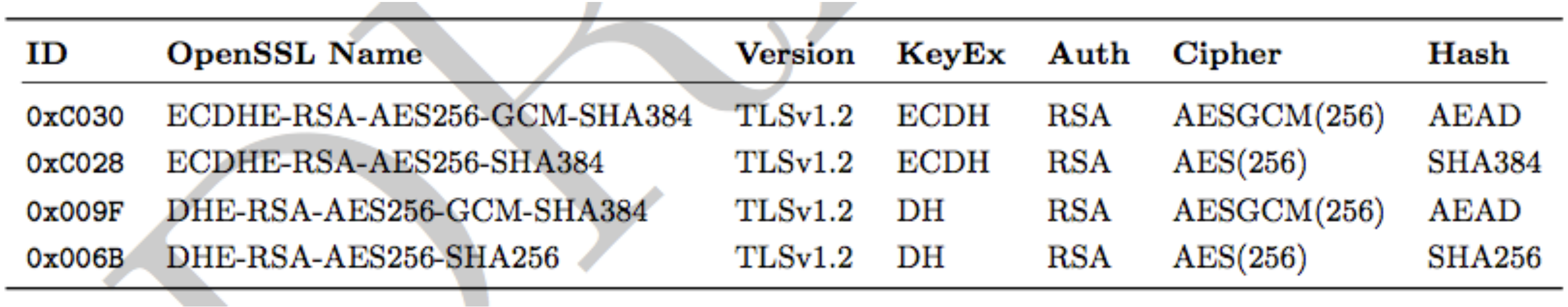
CIPHER SUITES

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 better
  - Disable TLS-Compression (SSL-CRIME Attack)
  - Implement HSTS (HTTP Strict Transport Security)
- Variant A: fewer supported clients
- Variant B: more clients, weaker settings

Variant A

’EECDH+aRSA+AES256:EDH+aRSA+AES256:!SSLv3’



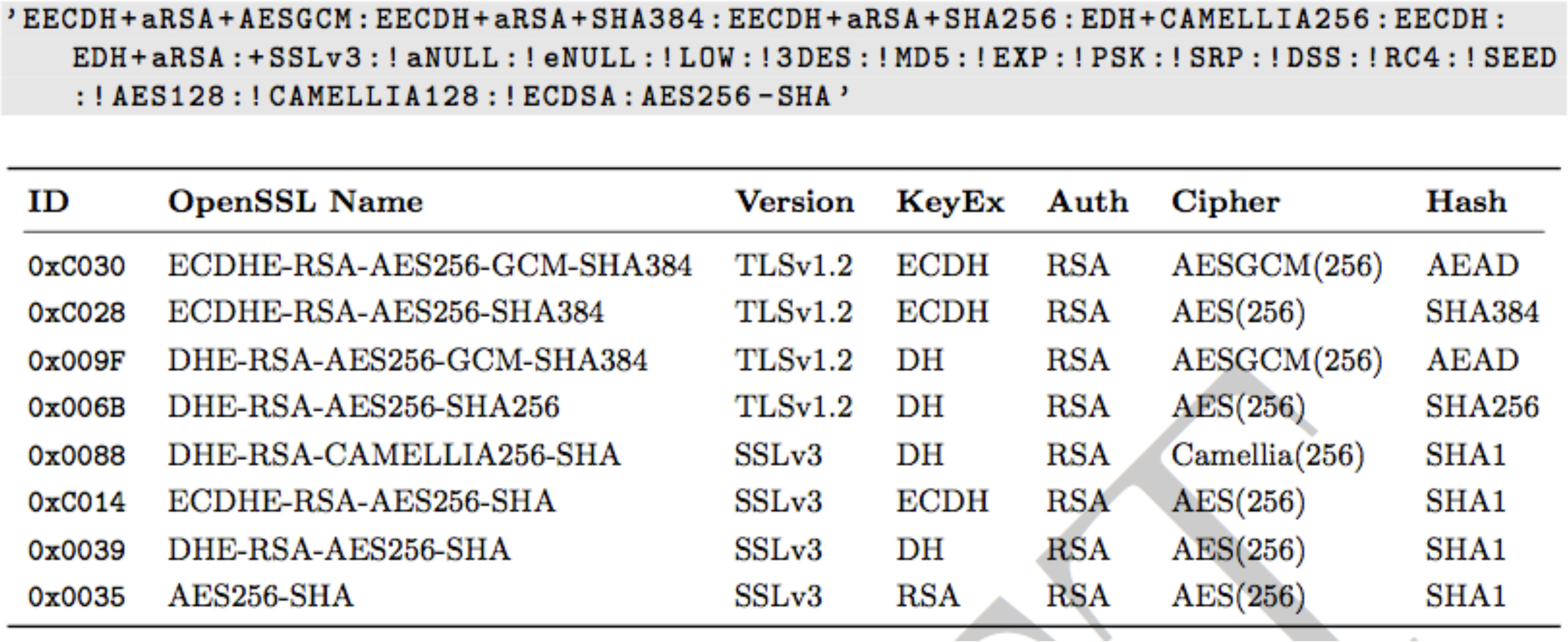
ID	OpenSSL Name	Version	KeyEx	Auth	Cipher	Hash
0xC030	ECDHE-RSA-AES256-GCM-SHA384	TLSv1.2	ECDH	RSA	AESGCM(256)	AEAD
0xC028	ECDHE-RSA-AES256-SHA384	TLSv1.2	ECDH	RSA	AES(256)	SHA384
0x009F	DHE-RSA-AES256-GCM-SHA384	TLSv1.2	DH	RSA	AESGCM(256)	AEAD
0x006B	DHE-RSA-AES256-SHA256	TLSv1.2	DH	RSA	AES(256)	SHA256

Compatibility:

Only clients which support TLS1.2 are covered by these cipher suites (Chrome 30, Win 7 and Win 8.1, Opera 17, OpenSSL ≥ 1.0.1e, Safari 6 / iOS 6.0.1, Safari 7 / OS X 10.9)

Variant B

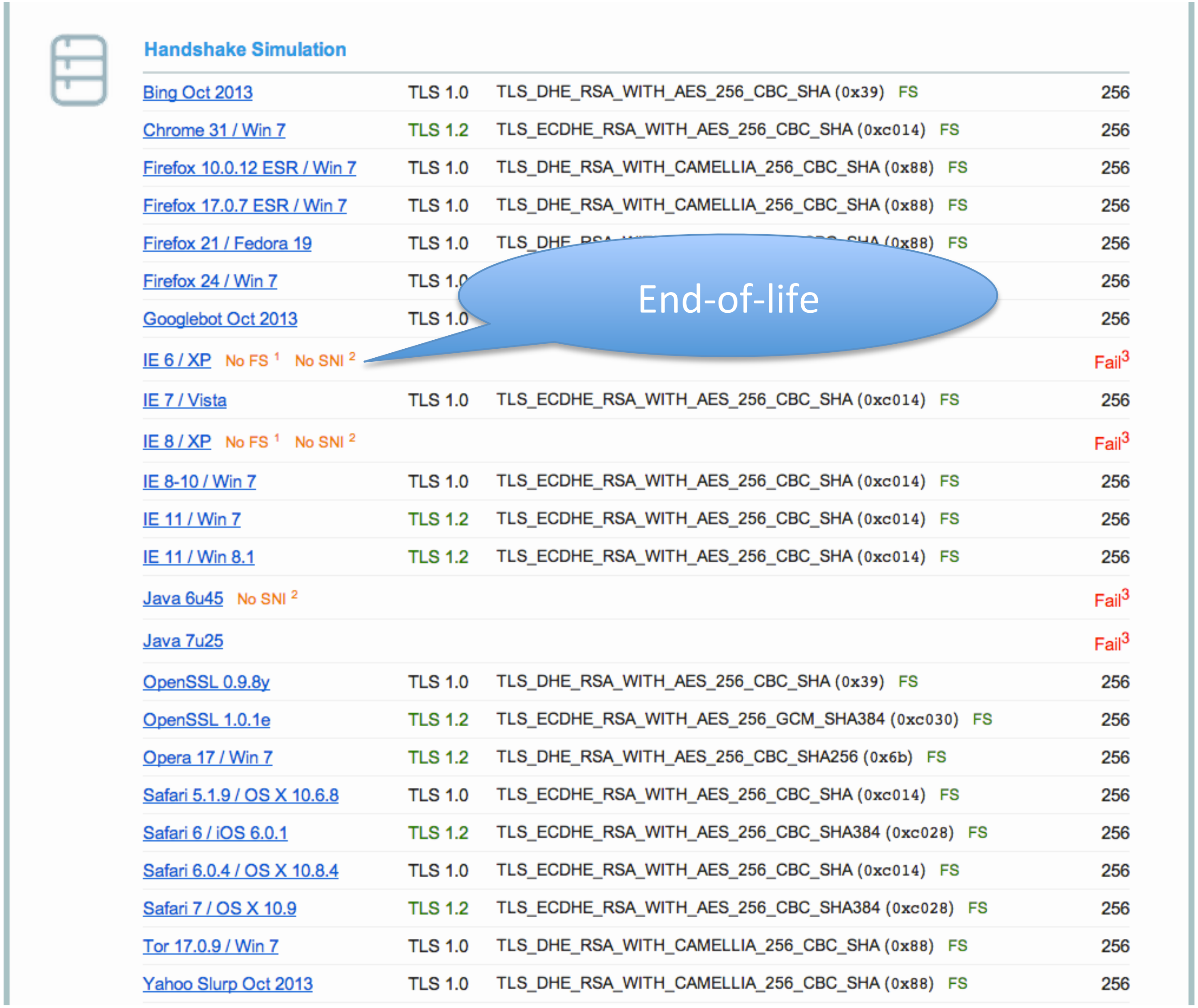
weaker ciphers, many clients



```
'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:!ECDSA:AES256-SHA'
```

ID	OpenSSL Name	Version	KeyEx	Auth	Cipher	Hash
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
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
0x0088	DHE-RSA-CAMELLIA256-SHA	SSLv3	DH	RSA	Camellia(256)	SHA1
0xC014	ECDHE-RSA-AES256-SHA	SSLv3	ECDH	RSA	AES(256)	SHA1
0x0039	DHE-RSA-AES256-SHA	SSLv3	DH	RSA	AES(256)	SHA1
0x0035	AES256-SHA	SSLv3	RSA	RSA	AES(256)	SHA1

Variant B: Compatibility



Handshake Simulation

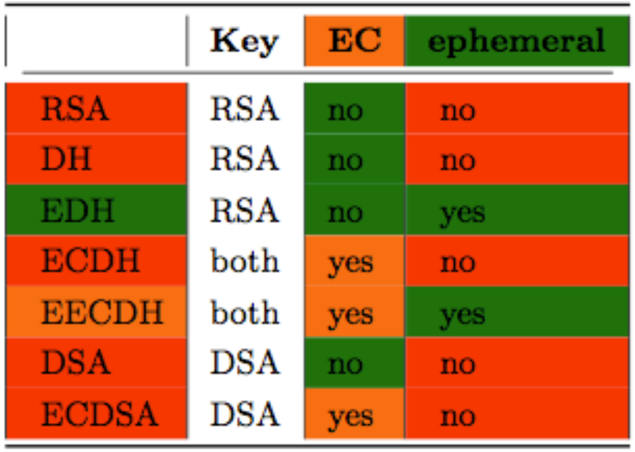
Bing Oct 2013	TLS 1.0	TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x39)	FS	256
Chrome 31 / Win 7	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)	FS	256
Firefox 10.0.12 ESR / Win 7	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88)	FS	256
Firefox 17.0.7 ESR / Win 7	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88)	FS	256
Firefox 21 / Fedora 19	TLS 1.0	TLS_DHE_RSA SHA (0x88)	FS	256
Firefox 24 / Win 7	TLS 1.0			256
Googlebot Oct 2013	TLS 1.0			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
Safari 7 / OS X 10.9	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028)	FS	256
Tor 17.0.9 / Win 7	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88)	FS	256
Yahoo Slurp Oct 2013	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88)	FS	256

Choosing your own cipher string (1)

- Rolling your own cipher suite string involves a trade-off between:
  - Compatibility (server <-> client), vs.
  - Known weak ciphers/hashes/MACs
  - The choice ECC or not, vs.
  - Support by different ssl libs (gnutls, openssl,...) vs.
  - Different versions of ssl libs
- In case of ssl lib version issues: do you want to re-compile the whole server for a newer version?
- Be aware of these issues before choosing your own cipher suite

Choosing your own cipher string (2)

- Complexity
- Multi-dimensional optimisation



	Key	EC	ephemeral
RSA	RSA	no	no
DH	RSA	no	no
EDH	RSA	no	yes
ECDH	both	yes	no
EECDH	both	yes	yes
DSA	DSA	no	no
ECDSA	DSA	yes	no

- Consider strong alternatives to de-facto standards
- Potential future solution: generator for settings?

PRACTICAL SETTINGS

What we have so far

- Web server: Apache, nginx, MS IIS, lighttpd
- Mail: Dovecot, cyrus, Postfix, Exim
- DBs: Mysql, Oracle, Postgresql, DB2
- VPN: OpenVPN, IPSec, Checkpoint, ...
- Proxies: Squid, Pound
- GnuPG
- SSH
- IM servers (jabber, irc)

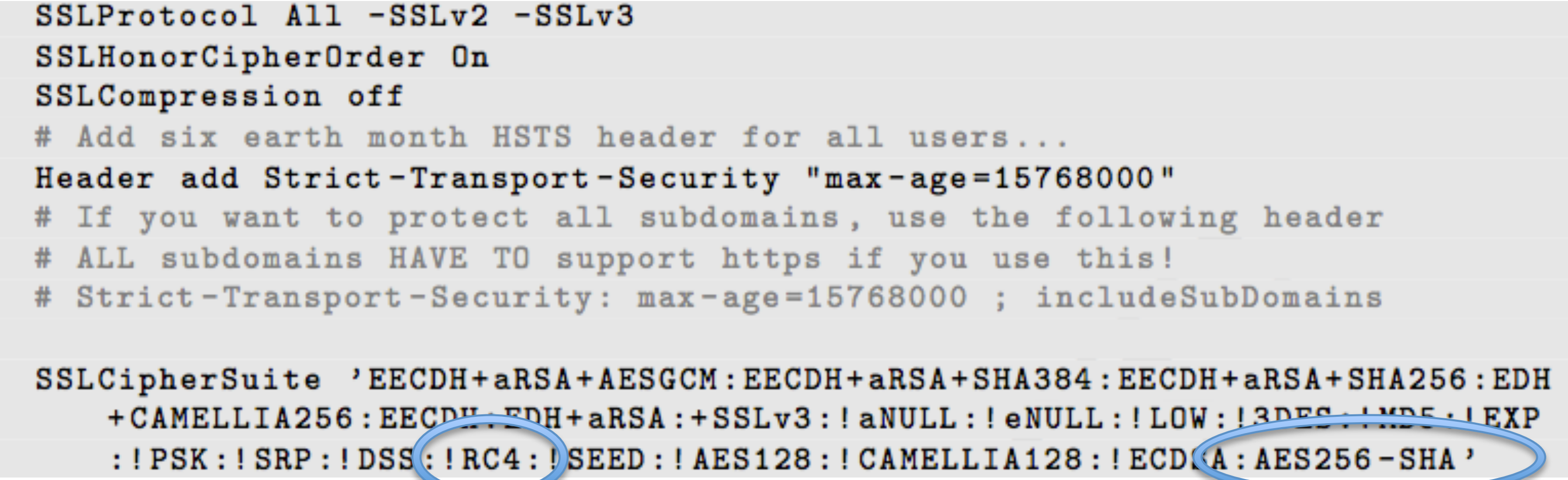
What we would like to see

- Mail: Exchange
- SIP
- RDP

- Everything as HTML (easier to copy & paste)
- Config generator on the website

Example: Apache

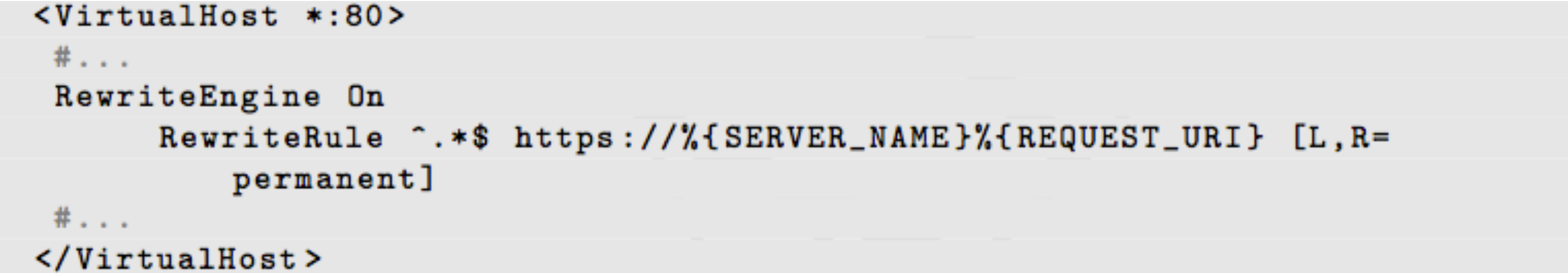
Selecting cipher suites:



```
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:!ECDSA:AES256-SHA'
```

Additionally:



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

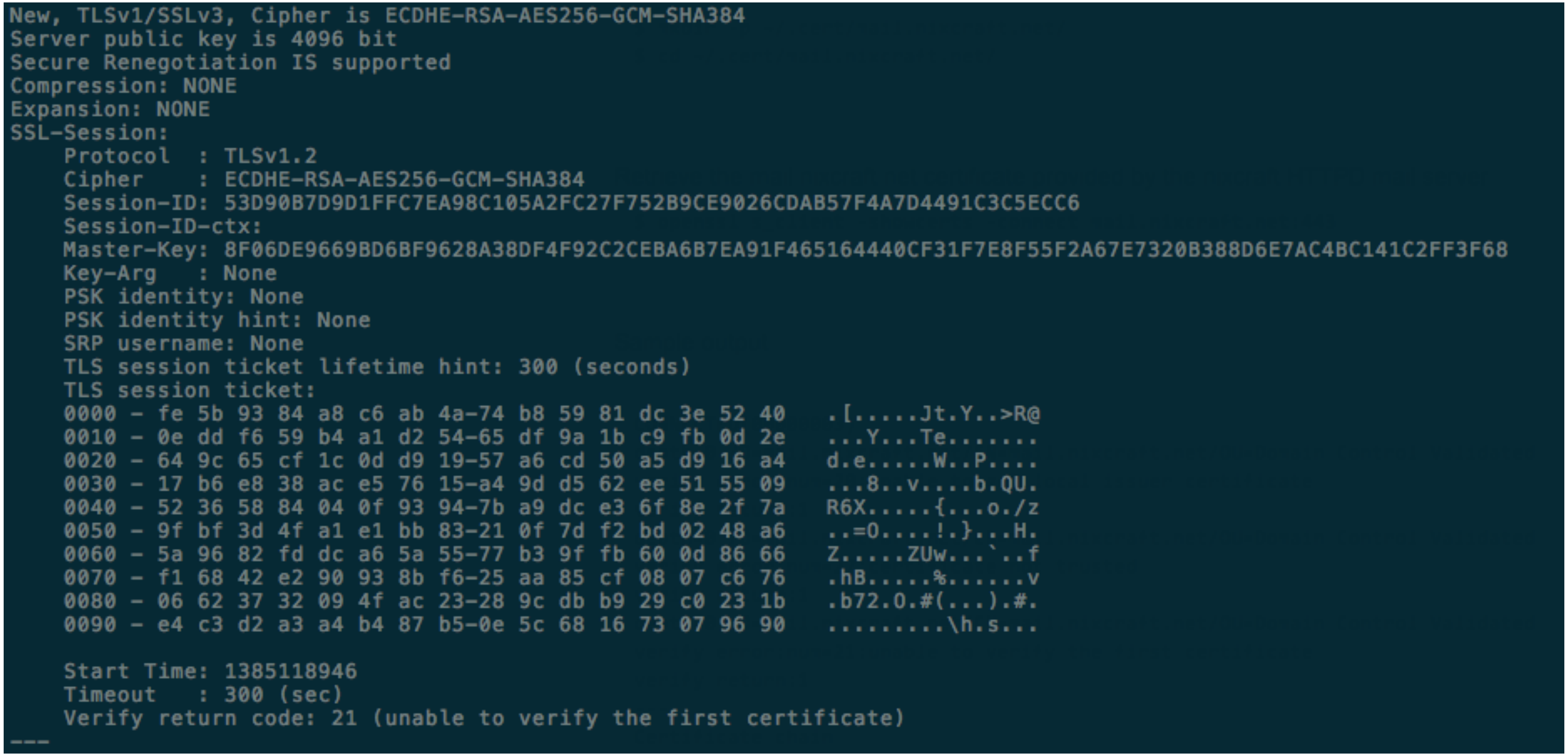
TESTING

How to test? - Tools

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

Tools: openss s_client

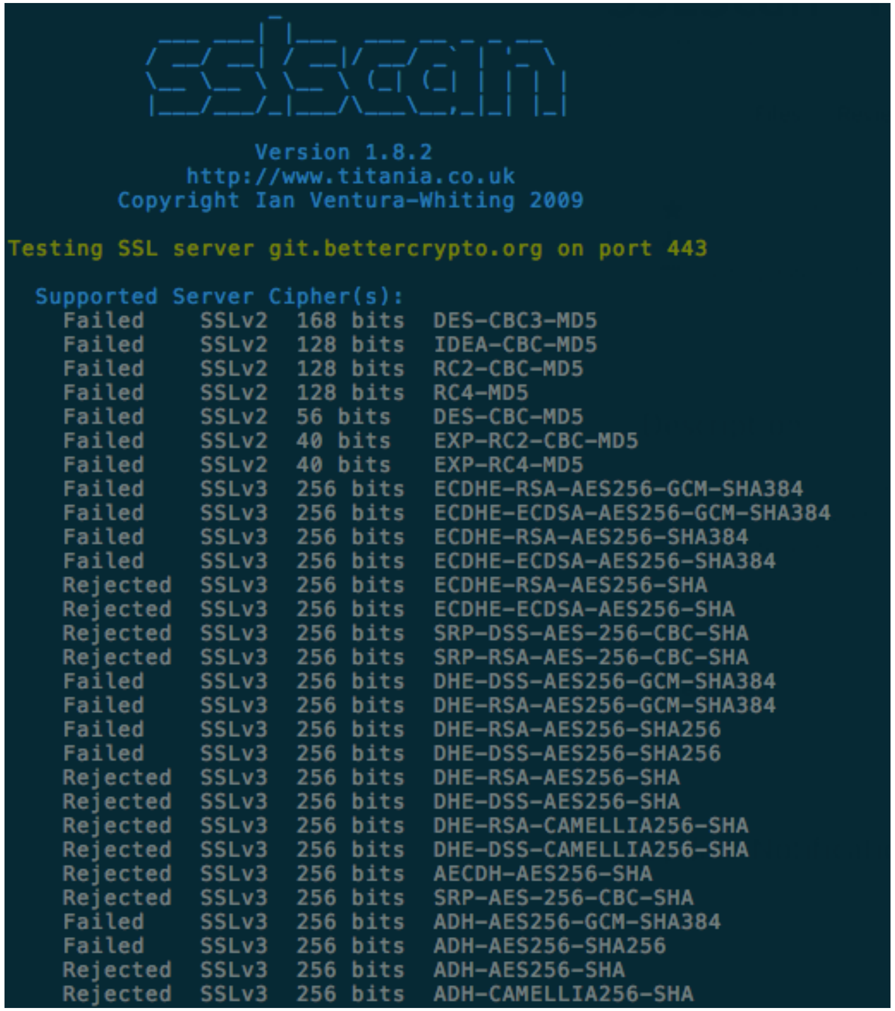
openssl s_client -showcerts –connect git.bettercrypto.org:443



```
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-Arg   : 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   ..=O....!.}...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.O.#(...).#.
    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: sslscan



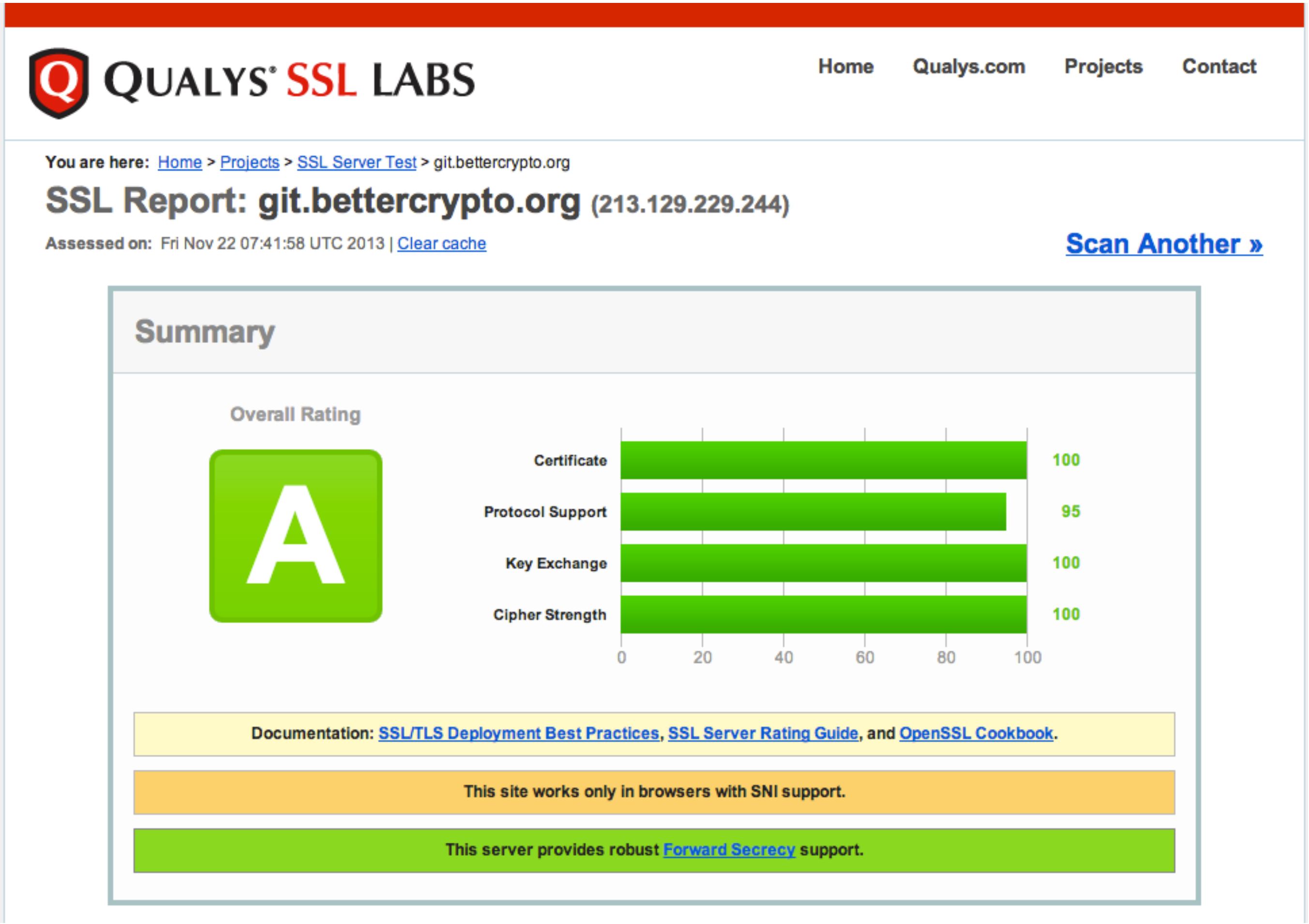
```
                   _
           ___ ___| |___  ___ __ _ _ __
          / __/ __| / __|/ __/ _` | '_ \
          \__ \__ \ \__ \ (_| (_| | | | |
          |___/___/_|___/\___\__,_|_| |_|

                  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

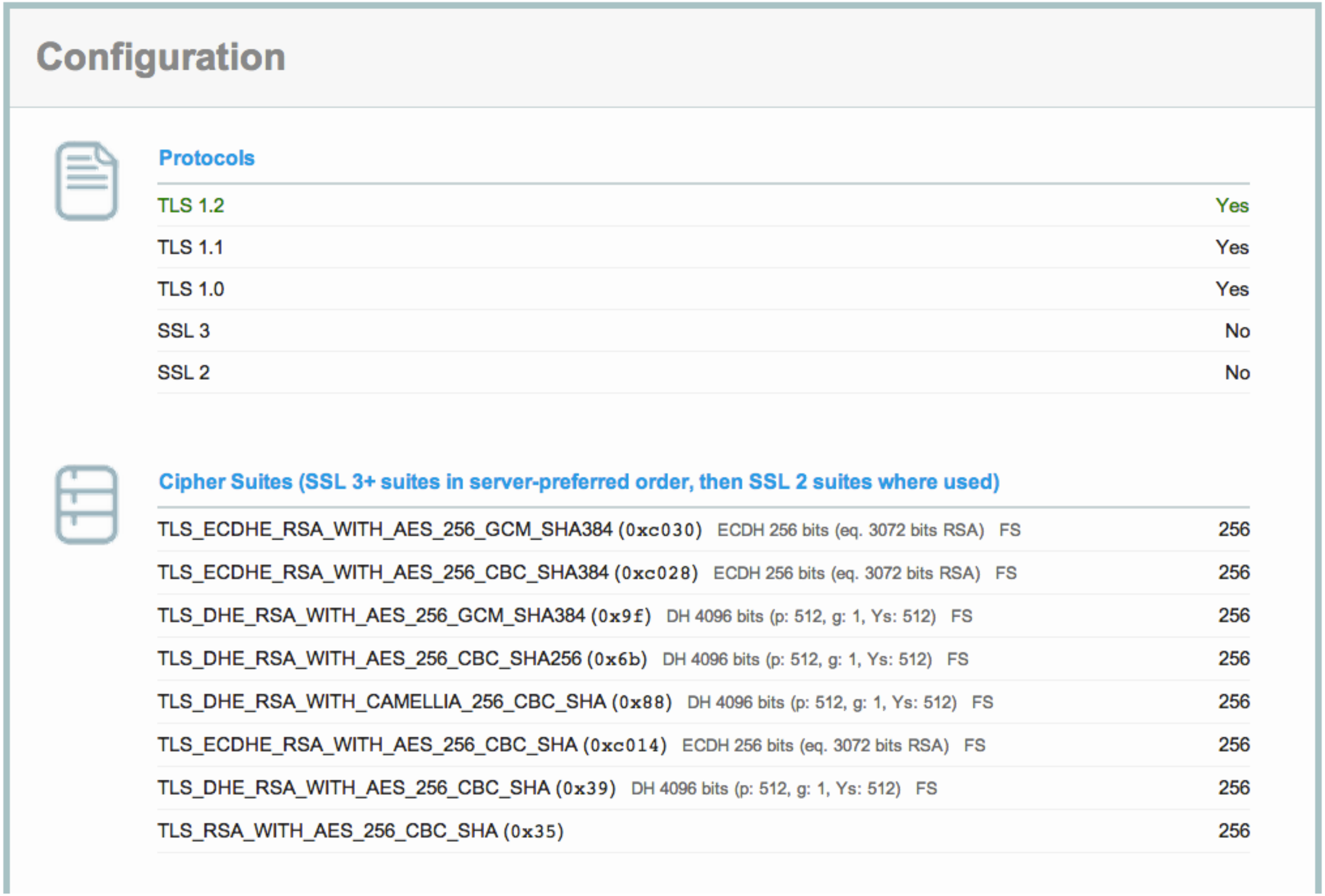
Summary

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)



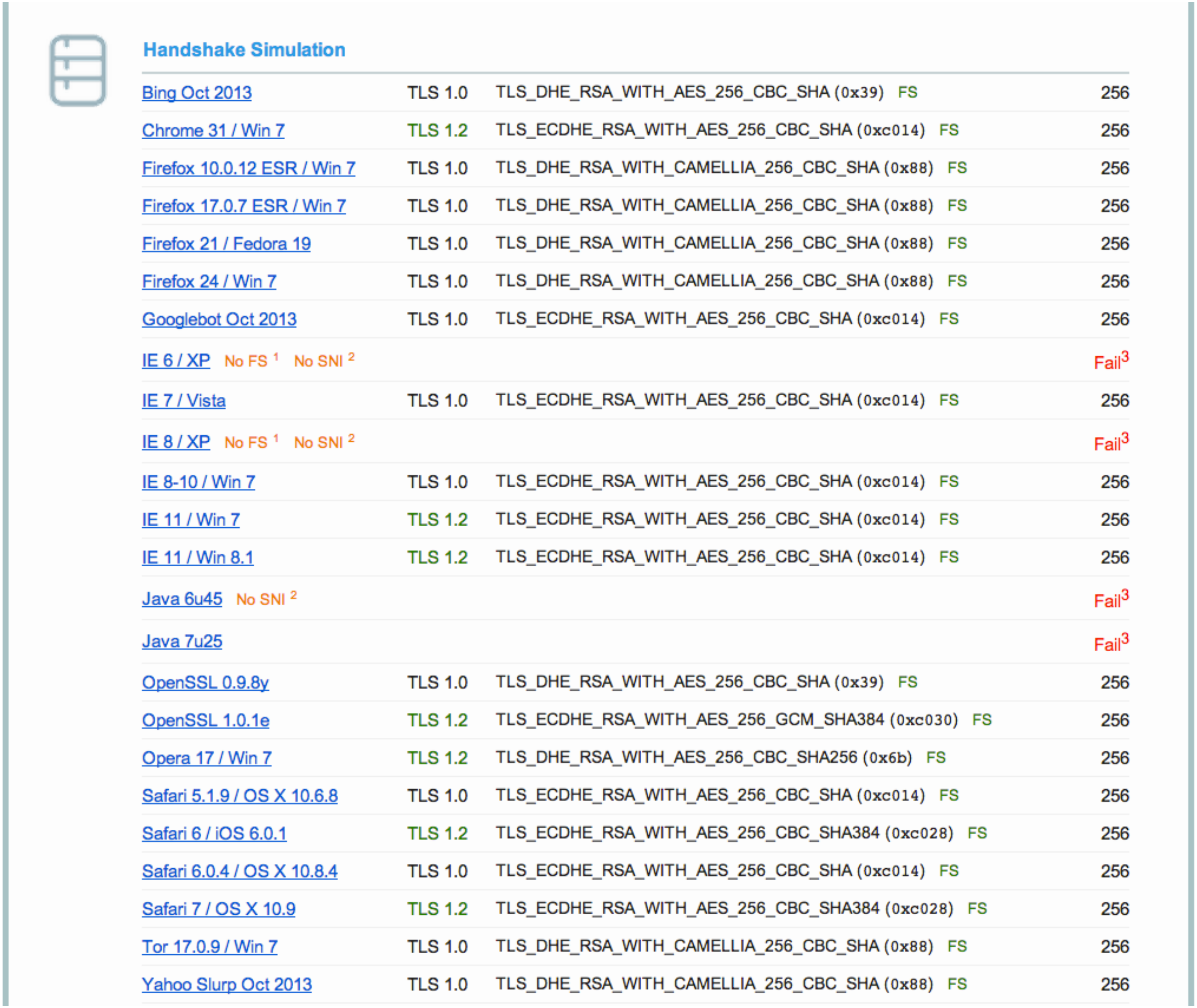
Configuration

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

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
Safari 7 / OS X 10.9	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028)	FS	256
Tor 17.0.9 / Win 7	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88)	FS	256
Yahoo Slurp Oct 2013	TLS 1.0	TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x88)	FS	256

WRAP-UP

Current state as of 2014/04

- ✓ Solid basis with Variant (A) and (B)
- ✓ Public draft was presented at the CCC
- Section „cipher suites“ still a bit messy, needs more work
- Need to convert to HTML

How to participate

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

Links

- Website: www.bettercrypto.org
- Git repo: https://git.bettercrypto.org
- Mailing list: http://lists.cert.at/cgi-bin/mailman/listinfo/ach

Thank you!