BetterCrypto - three years in..

TROOPERS16, Heidelberg, DE | 2016-03-17

Aaron Zauner | @a_z_e_t | azet@azet.org

Timeline

We start at the beginning. The year is 2013.

- June: Snowden revelations
- Summer: More leaks start apprearing
- . . . People start talking about a Crypto-Apocalypse (OMG!)
- August: Aaron Kaplan and Adi Kriegisch start discussing this topic/guide
- September/October: Project goes public, a lot of contributions and ML discussion

Motivation

- Lack of available guides for sysadmins/mgmt for ‘crypto hardening’
- No up-to-date blog posts we could make use of
- Crypto-guides (ENISA, eCrypto II, NIST etc.) for experts, not end-users/admins

BetterCrypto

BetterCrypto(.org) - Applied Crypto Hardening is born

- Clear audience: sysadmins without expert knowledge (e.g. crypto),
  management, decision makers,..
- Clear target: explain all decisions, have open-mailing list discussion, everything FOSS, public and auditable

BetterCrypto (cont.)

- Do at least something against the **Cryptocalypse**
- Check SSL, SSH, PGP crypto settings in the most common services and certificates:
  - Apache, Nginx, lighthttpd
  - IMAP/POP servers (dovecot, cyrus, . . .) – openssl.conf
  - Etc.
- Write down our experiences as guide
- Create easy, copy & paste-able settings which are "OK" (as far as we know) for sysadmins.
- Many eyes must check this!
- FOSS

Why is this relevant for you?

- You run networks and services. These are targets. If you believe it or not.
- You produce code. Make sure it uses good crypto coding practices
- However good crypto is hard to achieve
- Crypto does not solve all problems, but it helps

Who?

Wolfgang Breyha (uni VIE), David Durvaux, 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), IAIK, A-Sit, ...

- Sysadmins
- Engineers
- Devs.
- Cryptographers
- Security Engineers
- ...

Contents.

About 100 pages. Rough Overview:

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

Methods and Principles

Methods:

- Public review
- commits get **discussed**
- recommendations **need** references (like wikipedia)
- Every commit gets logged & we need your review!

How to contribute?

- `https://git.bettercrypto.org` (master, read-only)
- `https://github.com/BetterCrypto/` (please clone this one & send PRs)

1. discuss the changes first on the mailinglist
2. clone
3. follow the templates
4. send pull requests
5. **split the commit into many smaller commits**
6. don't be cross if something does not get accepted.
7. be ready for discussion

What do we provide?

- A common 'CipherString'
- Template configurations for *a lot* of different open source projects (also as textfiles)
- References, Crypto Background, Testing, Tools, etc,..

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)
- *DANE* (this section is still WIP)
- *Configuration code snippets*

CipherString and Suite

In SSL/TLS terminology; a *ciphersuite* combines the previously mentioned cryptographic techniques to work together and forms part of a secure (online) communication protocol

- Elliptic Curve Diffie-Hellman (Ephemeral - PFS)
- RSA
- AES128
- Galois Counter Mode (GCM)
- SHA256
- IANA standardized TLS parameters TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

differs between implementations (openssl, gnutls, cryptoapi etc.) and versions!

(Perfect) Forward Secrecy

Problem:

- Three letter agency (TLA) records all encrypted traffic
- Someday TLA gains access to private-key (Brute Force, Physical Force)
- TLA can decrypt all recorded traffic

Solution:

- **Ephemeral** session keys via Diffie Hellman (**ECDHE** and **DHE**)

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? Is that enough?

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

— Vincent Rijmen in a personal mail exchange Dec 2013

- Some theoretical attacks on AES-256

CipherString and Suite

- What is a SSLCipherSuite?
- vs. SSLProtocol
- Example:

```
SSLProtocol All -SSLv2 -SSLv3
SSLCipherSuite
```

‘EDH+CAMELLIA:EDH+aRSA:EECDH+aRSA+AESGCM:EECDH+a
56:EECDH:+CAMELLIA256:+AES256:+CAMELLIA128:+AES128:+S
:!DSS:!RC4:!SEED:!ECDSA:CAMELLIA256-SHA:AES256-
SHA:CAMELLIA128-SHA:AES128-SHA’

CipherString and Suite

- General:
  - Disable SSL 2.0 (weak protocol and algorithms)
  - Disable SSL 3.0 (BEAST, POODLE)
  - Disable RC4 cipher (RFC7465)
  - Disable EXPORT suites (FREAK Attack)
  - Enable TLS 1.0 or better
  - Disable TLS-Compression (SSL-CRIME Attack)
  - Implement HSTS (HTTP Strict Transport Security)
  - Implement OCSP stapling (Security and performance improvement)
- 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
5, Safari 7/OS X 10.9)
Excellent for controlled environments, like intranet.

Variant B

- weaker ciphers, broad client support

```
'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

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:!ECDHA:AES256-SHA'
```

- Additionally mod_rewrite:

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

Testing

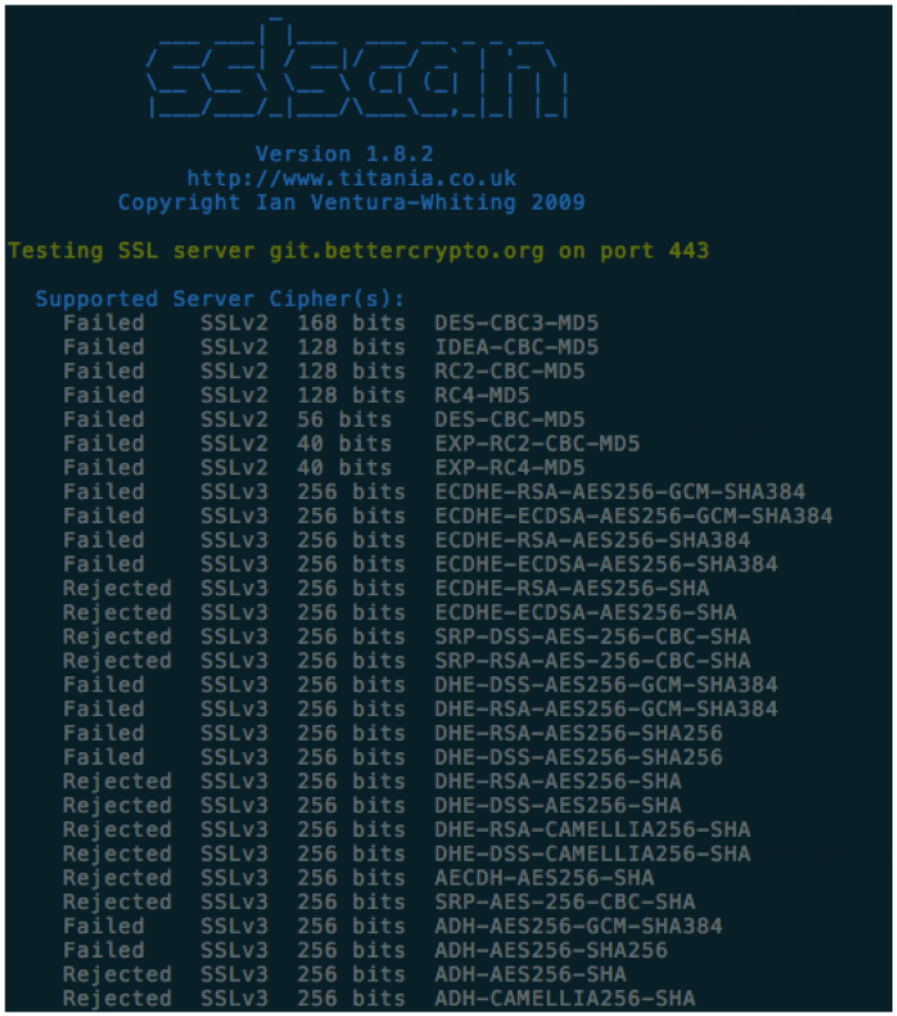
Tools: openssl s_client

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

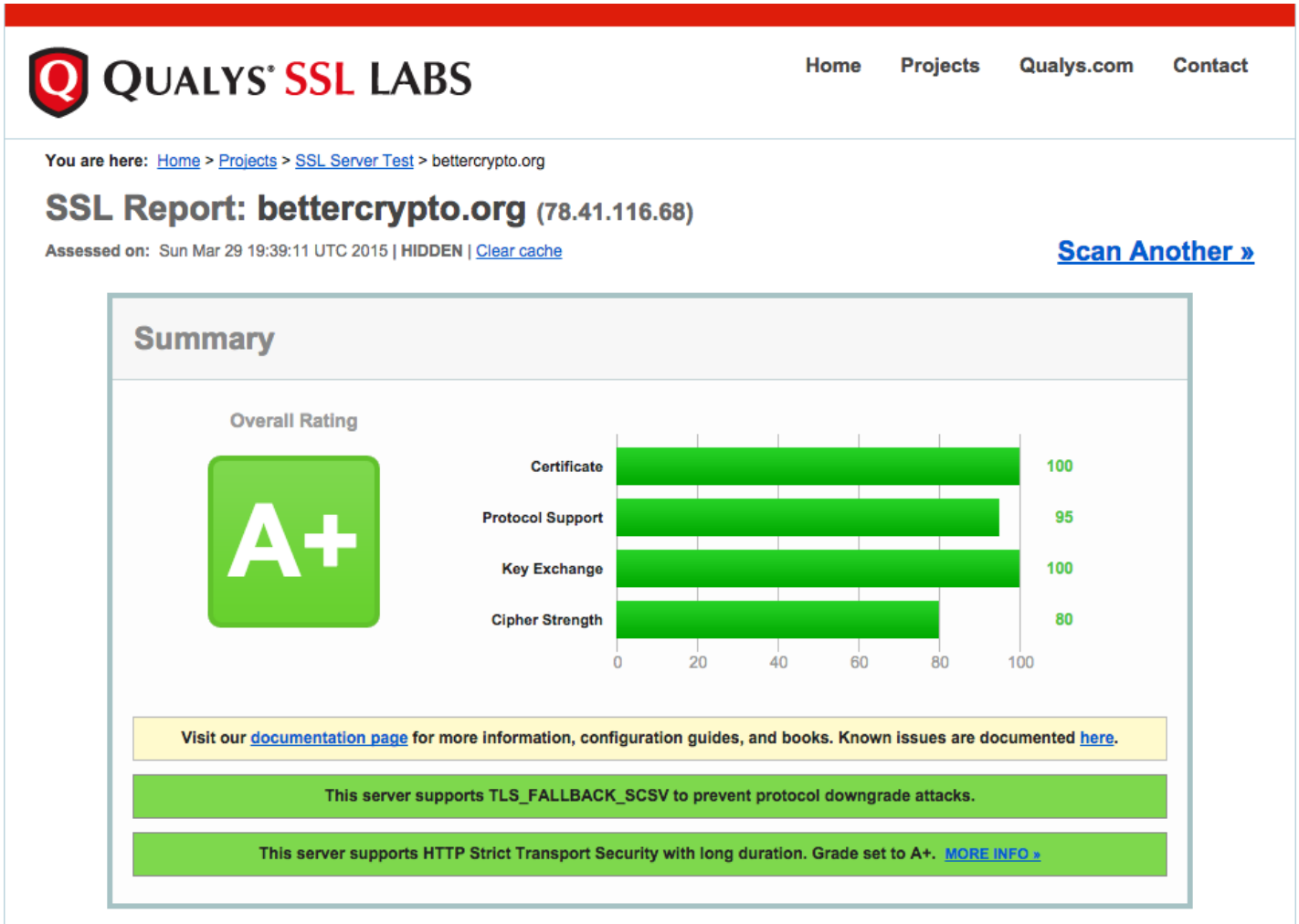
```
New, TLSv1/SSLv3, Cipher is ECDHE-RSA-AES256-GCM-SHA384
Server public key is 4096 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol  : TLSv1.2
    Cipher    : ECDHE-RSA-AES256-GCM-SHA384
    Session-ID: 53D90B7D9D1FFC7EA98C105A2FC27F752B9CE9026CDAB57F4A7D4491C3C5ECC6
    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



Tools: ssllabs.com



Tools: ssllabs.com (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

Tools: ssllabs.com (3)

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

Tools: SSLyze

SSLyze is a “Fast and full-featured SSL scanner”

A tool to test internally which cipher strings are supported.
The tool offers these features (amongst others):

- get a list of targets (ip:port) from a file
- XML output
- heartbleed test
- OCSP stapling test
- SSLv2-TLS1.2 testing
- finding preferred and supported cipher strings
- STARTTLS testing (IMAP, pop, . . .)
- XMPP testing
- SNI support
- HSTS testing

Tools: SSLyze (1)

```
aaron@homeostasis-old-2:~/Desktop/work/nic.at/projects/ACH/ach-master/tools/sslyze % python sslyze.py test.bettercrypto.org --regular --hsts


REGISTERING AVAILABLE PLUGINS
-----------------------------

 PluginOpenSSLCipherSuites
 PluginHSTS
 PluginCertInfo
 PluginSessionRenegotiation
 PluginCompression
 PluginSessionResumption
 PluginChromeSha1Deprecation
 PluginHeartbleed


CHECKING HOST(S) AVAILABILITY
-----------------------------

  test.bettercrypto.org:443          => 78.41.116.67:443


SCAN RESULTS FOR TEST.BETTERCRYPTO.ORG:443 - 78.41.116.67:443
---------------------------------------------------------

 * Deflate Compression:
     OK - Compression disabled

 * Session Renegotiation:
     Client-initiated Renegotiations:   OK - Rejected
     Secure Renegotiation:              OK - Supported

 * Certificate - Content:
     SHA1 Fingerprint:                  f706f915c6bd0ce2fcee9a27cc1b4d6f4152fc8b
     Common Name:                       test.bettercrypto.org
     Issuer:                            StartCom Class 1 Primary Intermediate Server CA
     Serial Number:                     057241752D85F3
     Not Before:                        Mar 28 17:36:44 2015 GMT
     Not After:                         Mar 29 08:47:53 2016 GMT
     Signature Algorithm:               sha256WithRSAEncryption
     Key Size:                          4096 bit
```

Tools: SSLyze (2)

```
    Signature Algorithm:                sha256WithRSAEncryption
    Key Size:                           4096 bit
    Exponent:                           65537 (0x10001)
    X509v3 Subject Alternative Name:    {'DNS': ['test.bettercrypto.org', 'bettercrypto.org']}

* Certificate - Trust:
    Hostname Validation:                OK - Subject Alternative Name matches
    "Mozilla NSS - 08/2014" CA Store:   FAILED - Certificate is NOT Trusted: unable to get local issuer certificate
    "Microsoft - 08/2014" CA Store:     FAILED - Certificate is NOT Trusted: unable to get local issuer certificate
    "Apple - OS X 10.9.4" CA Store:     FAILED - Certificate is NOT Trusted: unable to get local issuer certificate
    "Java 6 - Update 65" CA Store:      FAILED - Certificate is NOT Trusted: unable to get local issuer certificate
    Certificate Chain Received:         ['test.bettercrypto.org']

* Certificate - OCSP Stapling:
    NOT SUPPORTED - Server did not send back an OCSP response.

* HTTP Strict Transport Security:
    NOT SUPPORTED - Server did not send an HSTS header.

* OpenSSL Heartbleed:
    OK - Not vulnerable to Heartbleed

* TLSV1_2 Cipher Suites:
    Preferred:
                ECDHE-RSA-AES256-GCM-SHA384   ECDH-256 bits  256 bits      HTTP 200 OK
    Accepted:
                ECDHE-RSA-AES256-SHA384       ECDH-256 bits  256 bits      HTTP 200 OK
                ECDHE-RSA-AES256-SHA          ECDH-256 bits  256 bits      HTTP 200 OK
                ECDHE-RSA-AES256-GCM-SHA384   ECDH-256 bits  256 bits      HTTP 200 OK
                DHE-RSA-CAMELLIA256-SHA       DH-1024 bits   256 bits      HTTP 200 OK
                DHE-RSA-AES256-SHA256         DH-1024 bits   256 bits      HTTP 200 OK
                DHE-RSA-AES256-SHA            DH-1024 bits   256 bits      HTTP 200 OK
                DHE-RSA-AES256-GCM-SHA384     DH-1024 bits   256 bits      HTTP 200 OK
                CAMELLIA256-SHA               -              256 bits      HTTP 200 OK
                AES256-SHA256                 -              256 bits      HTTP 200 OK
                AES256-SHA                    -              256 bits      HTTP 200 OK
                AES256-GCM-SHA384             -              256 bits      HTTP 200 OK
                ECDHE-RSA-RC4-SHA             ECDH-256 bits  128 bits      HTTP 200 OK
                ECDHE-RSA-AES128-SHA256       ECDH-256 bits  128 bits      HTTP 200 OK
                ECDHE-RSA-AES128-SHA          ECDH-256 bits  128 bits      HTTP 200 OK
                ECDHE-RSA-AES128-GCM-SHA256   ECDH-256 bits  128 bits      HTTP 200 OK
                DHE-RSA-SEED-SHA              DH-1024 bits   128 bits      HTTP 200 OK
                DHE-RSA-CAMELLIA128-SHA       DH-1024 bits   128 bits      HTTP 200 OK
                DHE-RSA-AES128-SHA256         DH-1024 bits   128 bits      HTTP 200 OK
                DHE-RSA-AES128-SHA            DH-1024 bits   128 bits      HTTP 200 OK
                DHE-RSA-AES128-GCM-SHA256     DH-1024 bits   128 bits      HTTP 200 OK
                SEED-SHA                      -              128 bits      HTTP 200 OK
                RC4-SHA                       -              128 bits      HTTP 200 OK
                CAMELLIA128-SHA               -              128 bits      HTTP 200 OK
                AES128-SHA256                 -              128 bits      HTTP 200 OK
                AES128-SHA                    -              128 bits      HTTP 200 OK
                AES128-GCM-SHA256             -              128 bits      HTTP 200 OK
                ECDHE-RSA-DES-CBC3-SHA        ECDH-256 bits  112 bits      HTTP 200 OK
```

Mitigated Attacks

We've mitigated some high-profile TLS/SSL vulernabilities in the past years if you've deployed our guide. So far users have been pleased.

Mitigated Attacks: CRIME

- Requires TLS compression to perform attack.
- From the very beginning we've always turned off TLS or application level compression (BREACH e.g. is a very similar attack on HTTP compression).

Mitigated Attacks: POODLE

- Required SSLv3 ("TLS-POODLE" is specific to a certain unfamous vendor).
- We explicitly forbid SSLv3 - this kills the POODLE ;)

Mitigated Attacks: Logjam

- MITM Attack which requires 512, 768bit Diffie-Hellman to decrypt
- We've always recommended and, if possible, tried to supply a guide how to use DH params with $>=$ 1024 bits
- This was a discussion we had **very** early on in the project and a lot of contributors did their research well
- Some opened tickets, commited etc. - upstream

Mitigated Attacks: FREAK

- Requires EXPORT (low-security, early-90ties US ammunition export law) ciphers.
- We explicitly exclude EXPORT ciphers, problem solved.

Mitigated Attacks: DROWN

- Cross-"protocol" (version) attack that requires SSLv2.
- We've **always** recommended against enabling (completely insecure) SSLv2 in all configurations!
- Mail server daemon distributors used to recommend v2 - that's now gone as well.

Implementation specific attacks

- We can't do a lot against implementation specific attacks
- there have been quite a lot in the past years (OpenSSL, Apple, Microsoft, $APPLIANCEVENDOR,..)
- we try to provide a config guide, we're not auditors (up to sec. ngineers like you! ;))

Project statistics

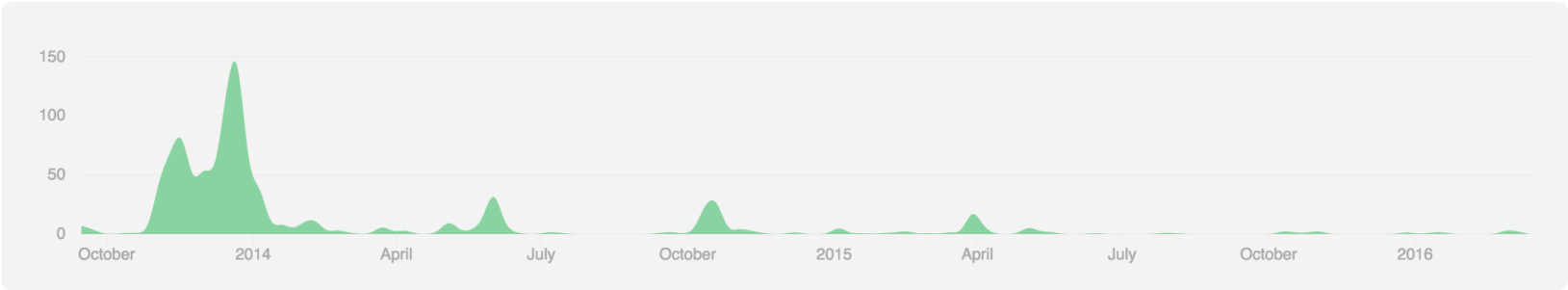
- 45 Contributors (git only) - ML as well (commited by others)
- 1501 commits
- Mostly LateX (a lot of overhead) - no line count stats.
- More than 1200 msgs to the mailing list

Project statistics (cont.)

Sep 15, 2013 – Mar 17, 2016

Contributions: **Commits**

Contributions to master, excluding merge commits



Project statistics (cont.)

- Always new posts to ML if new attacks appear
- Improvements to the document regularly on GitHub (e.g. Mail, Jabber section)

Discussion - PostQuantum

- some experts believe that a **real** quantum computer (not D-Wave) is only 10-50 years away
- new 'post-quantum crypto schemes' - no standard yet, we can't use them, almost no implementations
- we currently have no plans to add anything in this direction, but might in the future

Discussion - ECC

- djb et al: SafeCurves - are NIST/NSA curves trustable? parameters chosen by NSA, might contain backdoor?
- a lot of discussion in the project, we prefered DHE over ECDHE because of uncertainty in 2013
- Nowadays opinion by experts: common implementations are "hardened" and work well (see also: `https://eprint.iacr.org/2015/1018.pdf`)
- IETF will standardize new non-NIST curves in the near future, implementations will follow as will we.

Future

- We need continued input by (domain) experts.
- If you know a service, appliance or math well - talk to us, review,..
- This project is still active and needs a bit of upkeep by the community, if you like it, please help out

Future - testing?

What about automatically testing our configuration recommendations against different distributions, server daemon versions, TLS stacks et cetera? What about automatically deploying them as well?

- Jenkins
- Packer
- Vagrant
- Puppet/Chef/Ansible/CfEngine/..
- ...

Future - testing?

We'd like to provide a nice JavaScript-y webinterface for choosing your daemon version and getting a proper, up to date, secure configuration for it on the website. There has been interest in the past, but no one working on it currently.

Status Quo

- Distro and Package security w.r.t. Crypto settings has improved sigificantly over the past years
- Distribution security teams now work on similar issues, as do other secuity teams (e.g. browser vendors)
- There's a lot to do and **tons** of legacy systems with really bad configurations (see ongoing scanning research by various parties)
- IETF works on a lot of protocol and crypto security related improvements - time to market? ;)

Status Quo (cont.)

- Most sysadmins are still largely unfamiliar with the topic and follow stackexchange, (some-times wrong, unreviewed) blogposts etc.
- We do have quite a few sections and recommendations that need regular checking, love, testing and contribution
- The more people audit, discuss and review, the better the project becomes (if not spammed by crypto-tinfoils)

Questions?

- Website: `https://www.bettercrypto.org`
- Master (read-only) Git repo: `https://git.bettercrypto.org`
- Public github repo for PRs: `https://github.com/BetterCrypto/Applied-Crypto-Hardening`
- Mailing list: `http://lists.cert.at/cgi-bin/mailman/listinfo/ach`
- IRC: #bettercrypto on freenode
- Twitter: @bettercrypto