The OpenSSL 1.1 Audit

Kenneth White
@kennwhite

International Cryptographic Module Conference
May 19, 2016

Agenda

- Background
- OpenSSL audit update
- Remaining roadmap
- Questions

Open Crypto Audit Project

Open Crypto Audit Project

- Originally formed to manage community-funded TrueCrypt audit
- Independent technical research public interest organization
- Technical Advisory Board: academic, industry, and legal experts in security and privacy

Open Crypto Audit Project

- Originally formed to manage community-funded TrueCrypt audit
- Independent technical research public interest organization
- Technical Advisory Board: academic, industry, and legal experts in security and privacy
- Mission: Research, analysis & education around technical security in open source software
- Focus: software security, cryptography engineering, public awareness

Open Crypto Audit Project

- Originally formed to manage community-funded TrueCrypt audit
- Independent technical research public interest organization
- Technical Advisory Board: academic, industry, and legal experts in security and privacy
- Mission: Research, analysis & education around technical security in open source software
- Focus: software security, cryptography engineering, public awareness
- Current project: CII OpenSSL audit

Why OpenSSL?

Why OpenSSL?

Because it's everywhere.

Why OpenSSL?

server

desktop

mobile

Why OpenSSL?

DBs, middleware, Web Services

operating system updates

package managers

mail

libcurl

Why OpenSSL?

It's everywhere.

Why OpenSSL?

OpenSSL 1.0.2-FIPS is validated on over 100 platforms

Why OpenSSL?

Especially in the enterprise

About

Cisco AnyConnect™
Secure Mobility Client

OpenSSL

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE

Cisco AnyConnect™
Secure Mobility Client

libcurl

This product incorporates the libcurl HTTP library:
Copyright (c) 1996 - 2006, Daniel Stenberg,
(daniel@haxx.se).

The OpenSSL Audit

The OpenSSL Audit

- Commissioned by Linux Foundation's Core Infrastructure Initiative (CII)
- Ambitious Scope
  - Independent review
  - Coordinating closely with OpenSSL core team
  - Delayed for v. 1.1 maturity (*significant* refactor)
  - Diverse, complex codebase
    - Linux, BSDs, Windows, OSX, SRV5 (AIX, HP-UX, Solaris)
    - Intel x86 (incl. AES-NI), ARMv7, MIPS, PowerPC, Alpha…
    - FIPS module

OpenSSL Audit

- Goals
  - Thorough public security analysis of the core code in the next major release of OpenSSL
  - Demonstrate viability of a reusable open source test harness framework
  - Foster web-scale peer-reviewed public tools & data sets for protocol & negotiation analysis

OpenSSL Audit

Rough metrics: 412-494K total SLOC
OpenSSL v. 1.1 Master (*2015-03-14*)

C	71%	352,061
Asm*	25%	123,192
Perl*	3%	14,032
Shell	0.7%	3,249
C++	0.3%	1,370
	Total	493,904

OpenSSL Audit

- Phase 1 Goals
  - BigNum: multiprecision ints, constant time, blinding
  - BIO (focus on composition & file functions)
  - ASN.1 & x509 (cert & key parsing, DER/PEM decoding, structs, subordinate chains)
  - 93M cert corpus, "Frankencert" fuzzing
- Phase 2 Goals
  - TLS state machine
  - EVP (PKI constructions, H/MACs, envelopes)
  - Protocol flows, core engine implementation
  - Memory management
  - Crypto core (RSA, SHA-2, DH/ECDH, CBC, GGM…)

OpenSSL Audit

Caveats

- Schedule, funding, or quality: Pick 2
- High Priority
  - Major architectures
  - Modern (TLS 1.1+) protocols & primitives
  - DH, ECC, signatures, ASN.1 & x509
  - Non-crypto constructions (data structures, memory management, core API/ABI hooks)
- Lower Priority
  - AES implementation (finite field tables, matrix transformations, etc. TBD, possibly in Phase 3 formal academic cryptanalysis)
  - DTLS
  - S/MIME
  - OpenSSL s_server (smtp-aware web server!)

OpenSSL Audit

Major Software Components

- BIGNUM (code review & minor tooling)
- BIOs (code review & minor tooling)
- PEM/x509 Parsing (code review & tooling)
- ASN.1 (primarily tooling)
- Side channels in cryptographic primitives
- TLS Stack

OpenSSL Audit

Key Phase I Findings

OpenSSL Audit

Key Phase I Findings

- Complexity: led to some potential bugs invalidated due to pre- or post- target parsing
- PEM parsing contained unexpected formats including access to ASN.1 decoding facilities HMAC and CMAC algorithms
- Tooling used to provide most coverage for ASN.1 complex parsing
- Memory leak and integer overflow identified but very unlikely invalid or low severity issues
- RSA uses blinding and constant time operations by default
- RSA_padding_check_SSLv23 does not appear to be constant time, but is deprecated
- ECDSA also constant time, although implemented at the encryption layer rather than the BIGNUM layer
- Some overreads identified in the TLS stack handshake, but unlikely to result in security issues

OpenSSL Audit

Key Phase I Findings

- x509 & ASN.1fuzzing done on ~20M certs using afl-cmin
  - Corpus of 277 certificates that result in diverse paths being taken through the certificate parsing code.
  - Fuzzed the PEM_read_X509 function for 228 hours covering 28,552,385 executions, and 803 paths
  - Fuzzed the d2i_X509_fp function for 228 hours also, covering 28,647,659 executions and 959 paths.
  - x509 fuzzing resulted in no crashes or interesting results
  - DER fuzzing resulted in four instances of particularly slow execution
  - Tool developed to exercise several types of ASN.1 structures

OpenSSL Audit

Key Phase I Findings

TLS Handshake

- Some data structures in init_buf used wen parsing network input masked buffer overreads
- selftls did generate some crashes, but unlikely to lead to directly exploitable conditions (due to the oversized backing buffer)

Crashes identified by small stub developed for fuzzing the BIO_print function when the attacker can control a format string

No crashes identified by a small fuzzer developed for BIGNUM operations

OpenSSL Audit

Key Phase II Findings

OpenSSL Audit

Key Phase II Findings

- Potential code execution via a stack buffer overflow when processing SSLv3 records using certain digest functions during PSK authentication (deprecated)
- Potential code execution via heap buffer overflow during server key exchange messages
- Possible Denial of Service caused by an uncontrolled out of bound read while processing client key exchange messages
- Denial of Service caused by replay protections in DTLS
- A few cases of potentially unwiped secrets in memory, likely difficult to exploit

OpenSSL Audit

Future work

ChaCha20 and Poly1305 implementation

https://www.openssl.org/blog/blog/2016/02/15/poly1305-revised/

Documentation of EVP_* opaque structures (function calls to initialize and process, rather than direct access)

FIPS v 2.0 module

- Implemented on over 100 platforms
- Not in the initial release of v 1.1
- CMVP validation: $350K+ (est.)

Post-Logjam

The Real World

Real-world Apache/Nginx TLS

PROTOCOL	CIPHER	KEYEX	HMAC	MODE	AUTH
SSL v1	NULL	RSA	MD5	ECB	ECDSA
SSL v2	DES	DH	SHA-1	CBC	RSA
SSL v3	3DES	DHE	SHA-256	GCM	
TLS 1.0	RC4	ECDH	SHA-384		
TLS 1.1	Twofish		SHA-512		
TLS 1.2	Blowfish		Poly1305		
TLS 1.3	AES				
	ChaCha20				

Real-world Apache/Nginx TLS

PROTOCOL	CIPHER	KEYEX	HMAC	MODE	AUTH
SSL v1	NULL	RSA	MD5	ECB	ECDSA
SSL v2	DES	DH	SHA-1	CBC	RSA
SSL v3	3DES	DHE	SHA-256	GCM	
TLS 1.0	RC4	ECDH	SHA-384		
TLS 1.1	Twofish		SHA-512		
TLS 1.2	Blowfish		Poly1305		
TLS 1.3	AES				
	ChaCha20				

***Also*:**

HSTS (strict secure transport), HPKP (pinning), CT (cert transparency), SNI (virtual hosts)

Questions?