CS356: Passwords and More TLS

W. Michael Petullo

University of Wisconsin–La Crosse

As of November 3, 2021

Passwords and More TLS: Brute Force

- Cryptographic hash of passwords typically stored in password files.
- One-way hash prevents deriving passwords directly from password file; that is, if p is a password, s is a random salt, and H is a hash function then the password file is comprised of records containing
$$s, H(s, p).$$
- Given access to a password file, we can try to break it by extracting the salt s and, for each combination of characters c, calculating
$$s, H(s, c).$$
- If $H(s, p) = H(s, c)$, then $p = c$.
- Given a c-character password, this will take on average un^c units of time, where n is the number of legal password characters and u represents how long it takes to perform a single hash. (Exponential growth with length of password.)

Passwords and More TLS: Dictionary Attacks

- Cryptographic hash of passwords typically stored in password files.
- One-way hash prevents deriving passwords directly from password file; that is, if p is a password, s is a random salt, and H is a hash function then the password file is comprised of records containing
$$s, H(s, p).$$
- Given access to a password file, we can try to break it by extracting the salt s and, for each word w in the dictionary, calculating
$$s, H(s, w).$$
- If $H(s, p) = H(s, w)$, then $p = w$.
- On average we will find each password found in the dictionary in the time it takes to hash ½ of the words in the dictionary.

Passwords and More TLS: Hash Lookup Table

- Cryptographic hash of passwords typically stored in password files.
- One-way hash prevents deriving passwords directly from password file; that is, if p is a password, s is a random salt, and H is a hash function then the password file is comprised of records containing

  $$s, H(s, p).$$

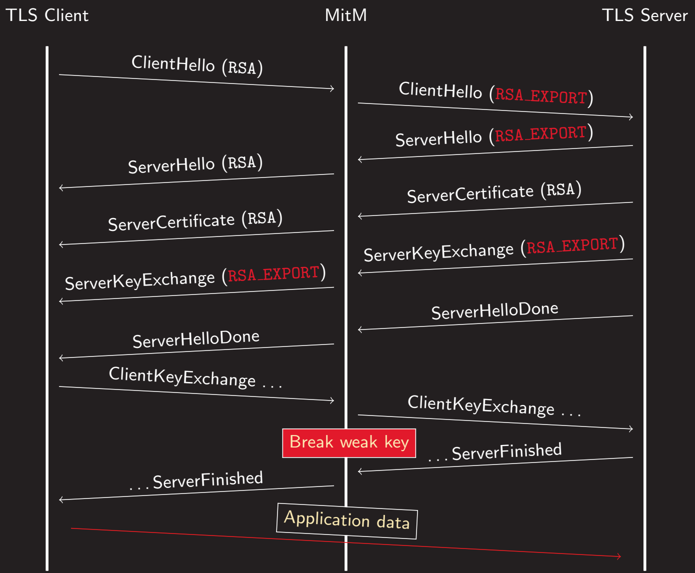
- What if we precomputed hashes?
- Very fast to check passwords, but requires an impossible amount of storage; storing only eight-character password/hash pairs requires

  $$sn^8$$

  bytes, where

  s is the number of salts, and
  n is the number legal password characters.

Passwords and More TLS: FREAK (2015)



TLS Client

MitM

TLS Server

ClientHello (RSA)

ClientHello (RSA_EXPORT)

ServerHello (RSA_EXPORT)

ServerHello (RSA)

ServerCertificate (RSA)

ServerCertificate (RSA)

ServerKeyExchange (RSA_EXPORT)

ServerKeyExchange (RSA_EXPORT)

ServerHelloDone

ServerHelloDone

ClientKeyExchange . . .

ClientKeyExchange . . .

Break weak key

. . . ServerFinished

. . . ServerFinished

Application data

2015 IEEE Symposium on Security and Privacy

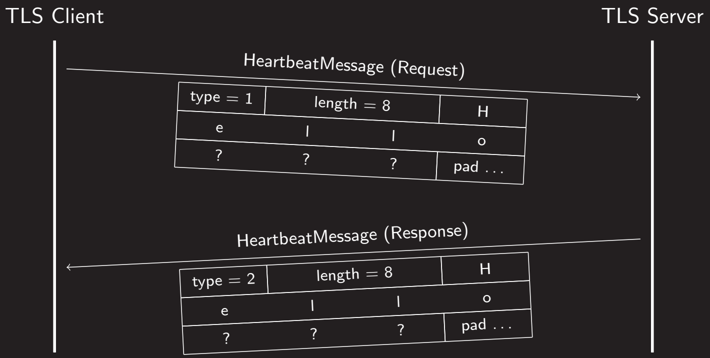
A Messy State of the Union:
Taming the Composite State Machines of TLS

Benjamin Beurdouche*, Karthikeyan Bhargavan*, Antoine Delignat-Lavaud*,
Cédric Fournet†, Markulf Kohlweiss†, Alfredo Pironti*,
Pierre-Yves Strub‡, Jean Karim Zinzindohoue§*

*INRIA Paris-Rocquencourt, †Microsoft Research, ‡IMDEA Software Institute, §Ecole des Ponts ParisTech

Error in the state machines of OpenSSL, SecureTransport (Apple), and Mono (open-source .NET) allowed MitM to downgrade the strength of TLS connection.

Passwords and More TLS: Heartbeat



Heartbeat is an extension to TLS. It allows for “keep alive” messages. A host sends a message, and its peer echos the message back.

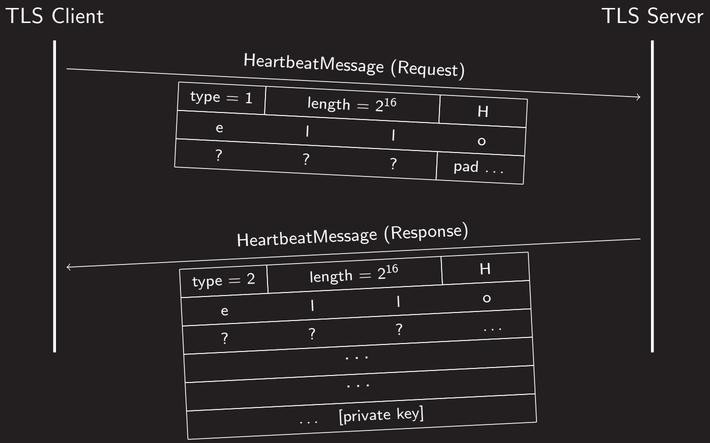
Memory Errors: Buffer Overflow—Heartbleed

CVE-2014-0160: Error in lines 5 and 14 trusts untrustworthy input!

```c
 1  unsigned char *p = &s->s3->rrec.data[0], *pl;
 2  unsigned short hbtype;
 3  unsigned int payload, padding = 16;
 4  hbtype = *p++;
 5  n2s(p, payload); /* Read payload length from packet. */
 6  pl = p;
 7  if (hbtype == TLS1_HB_REQUEST) {
 8      unsigned char *buffer, *bp;
 9      int r;
10      buffer = OPENSSL_malloc(1 + 2 + payload + padding);
11      bp = buffer;
12      *bp++ = TLS1_HB_RESPONSE; /* Message type to buffer. */
13      s2n(payload, bp);         /* Message size to buffer. */
14      memcpy(bp, pl, payload); /* Copy message to buffer. */
15  }
```

Summary of Heartbleed bug in OpenSSL

Passwords and More TLS: Heartbeat



TLS Client

TLS Server

HeartbeatMessage (Request)

type = 1	length = 2^{16}		H
e	l	l	o
?	?	?	pad . . .

HeartbeatMessage (Response)

type = 2	length = 2^{16}		H
e	l	l	o
?	?	?	. . .
	. . .		
	. . .		
	. . . [private key]		

Buffer overflow: Heartbleed exploits OpenSSL's trust of the attacker's HeartbeatMessage and the length therein. OpenSSL should have checked than the length does not exceed the size of the message, but it does not. Thus `memcpy` copies from beyond the buffer that contains the message, possibly copying secrets into the server's response.

TLS: Exercise

① Compile sample BSD socket service from tlsproxy project; interact with the service using `nc`.

② Compile the mbed TLS sample in your tlsproxy repository; use it to connect to a real website.

③ Run BSD socket service in strace; can you catch the `listen` and `accept` calls?

④ Use `ps aux` to identify the process ID of the BSD socket service; run `ls -l /proc/PID/fd`.

⑤ With the BSD socket service running, execute `netstat -ltnp`; do you see evidence of the program's socket?

⑥ Connect to the BSD socket service, and execute `netstat -tnp`; do you see a socket in a different state? How has `/proc/PID/fd` changed?

⑦ Observe interaction with BSD socket service using Wireshark.

⑧ Observe tlsproxy template's TLS handshake using Wireshark.

Passwords and More TLS: Assignments

Graded Homework	Aquinas: tlsproxy
Reading	Read 0x700–0x753

`https://www.flyn.org/courses/cs356/schedule`