

TLS Parser

PA193 – Project
2016

Scope

- TLSv1.0 – TLSv1.2.
- Record protocol + Handshake protocol.
 - Client/ServerHello.
 - Certificate.
 - Client/ServerKeyExchange.
 - ServerHelloDone.
- Binary format (see RFC 5246).

Overall Design

- Parser processes input and builds a *HandshakeMessage* structure.
- Record layer protocol headers -> Handshake protocol headers -> Handshake message.
- Parsed values are printed on standard output.
- For example of usage see README.md.

Overall Design

```
// This is how the message looks like as  
// a whole (record layer + actual message)
```

```
typedef struct {  
    ContentType cType;  
    ProtocolVersion version;  
    uint16_t fLength;  
    HandshakeType hsType;  
    uint32_t mLength;  
    unsigned char *body; // Contains the rest of the  
    message  
} HandshakeMessage;
```

Parsing Client Hello

- Message Length should be at least 38 Bytes
- First 2 Bytes - Version – 0x03 + 0x01,0x02,0x03
- 4 Bytes time stamp.
- 28 Random Bytes.
- 1 Byte Session id length.
- Session id – Variable length
- 2 Bytes Cipher Suite length.
- Cipher Suite Variable length – Not Decoded.
- 1 Bytes – Compression method length
- Compressions methods one byte per method – Not decoded.
- Extensions – if any – checked.

Parsing Client Hello

```
typedef struct {  
    ProtocolVersion version;  
    Random random;  
    SessionID sessionId;  
    CipherSuiteCollection csCollection;  
    CompressionMethod compressionMethod;  
    uint8_t hasExtensions;  
    unsigned char *extensions;  
} ClientHello;
```

Parsing Server Hello

- Message Length should be at least 38 Bytes
- First 2 Bytes - Version - 0x03 + 0x01,0x02,0x03
- 4 Bytes time stamp.
- 28 Random Bytes.
- 1 Byte Session id length.
- Session id - Variable length
- 2 Bytes Cipher Suite Selected.
- 1 Bytes - Compression method Selected.
- Extensions - if any - checked.

Parsing Server Hello

```
typedef struct {  
    ProtocolVersion version;  
    Random random;  
    SessionID sessionId;  
    unsigned char cipherSuite[2];  
    uint8_t compresionMethod;  
    uint8_t hasExtensions;  
    unsigned char *extensions;  
} ServerHello;
```

Parsing Key Exchange

- Message length given in 2/3 Bytes.
- The details of the key not decode.
- Structure.

```
typedef struct {
    uint32_t mLength;
    unsigned char * ServerDHParams;
} ServerKeyExchange;

typedef struct {
    uint16_t pubKeyLength;
    unsigned char * pubKey;
} ClientKeyExchange;
```

Code Testing

- Valid messages for coding were created using Wireshark and ssldump.
- Messages for the following were created:
 - Client Hello.
 - Server Hello.
 - Client Key Exchange.
 - Certificate.
 - Server Key Exchange.
 - Server Hello Done.

Code Fuzzing

- In valid messages was created using RADAMSA, which is general purpose random fuzzer.
- RADAMSA created malformed outputs based on sample input given.
- Code was tested for 10000 sample/random test cases created by RADAMSA.
- The code did not crashed for any of the samples created by fuzzer.

Distribution of work

- Parser architecture / overall processing flow.
 - Martin Bajanik.
- Server & Client Hello messages parsing.
 - Mayank Samadhiya and Milan Patnaik.
- Key Exchanges messages parsing.
 - Mariami Gonashvili.
- Test cases & Code Testing.
 - Mayank Samadhiya and Milan Patnaik.
- Presentation, consultations and reviews.
 - Everyone.

Questions