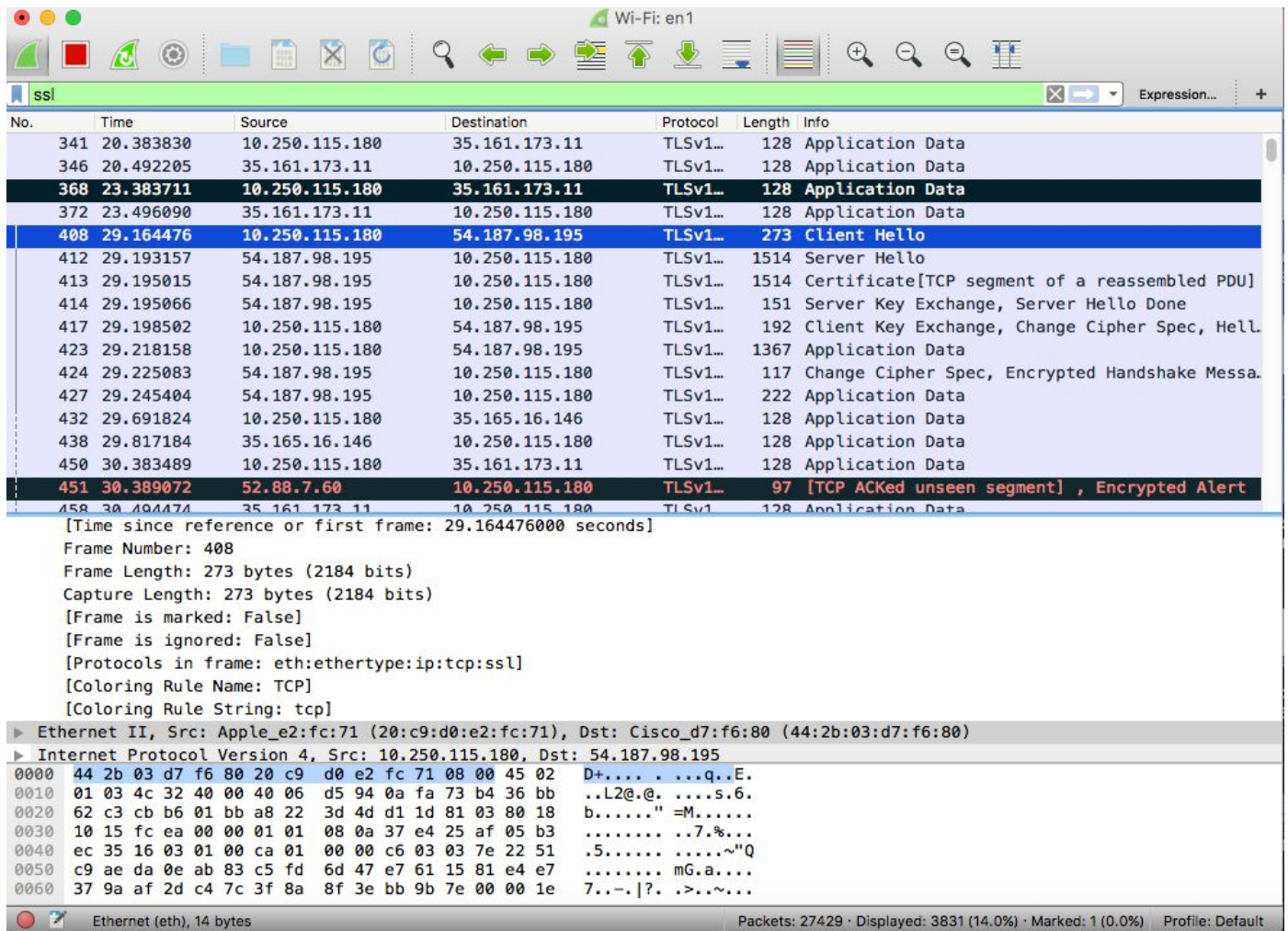


Paul Diaz  
HW9  
CS 166

1. Use Wireshark to capture SSL authentication packets.



- a. The communication starts off by Alice saying “Can we talk” which is “Client Hello” highlighted in blue — source 10.250.115.180. Then server says hello back which is “Server Hello” in wireshark — source 35.161.173.11 then sends Certificate[TCP segment of reassembled PDU] and Server Key Exchange, Hello Done. Then client responds back by sending Client Key Exchange, Change Cipher spec, Hello. Then server sends Change Cipher Spec, Encrypted Handshake Message, then Encrypted Alert.

- b. SSL packets contain the following — One packet contains Handshake protocol that contains handshake type, length, and version. It also contains random session ID length, cipher suites length, compression method and its length, extension server name, and others (please see pic attached).

The image shows a Wireshark packet capture window. The title bar reads "Wireshark · Packet 408 · wireshark\_en1\_20170508190119\_uYX9hs". The packet list on the left shows:

- Internet Protocol Version 4, Src: 10.250.115.180, Dst: 54.187.98.195
- Transmission Control Protocol, Src Port: 52150, Dst Port: 443, Seq: 1, Ack: 1, Len: 207
- Secure Sockets Layer
  - TLSv1.2 Record Layer: Handshake Protocol: Client Hello
    - Content Type: Handshake (22)
    - Version: TLS 1.0 (0x0301)
    - Length: 202

The packet details pane on the right shows the structure of the "Handshake Protocol: Client Hello" packet:

- Handshake Type: Client Hello (1)
- Length: 198
- Version: TLS 1.2 (0x0303)
- Random
- Session ID Length: 0
- Cipher Suites Length: 30
- Cipher Suites (15 suites)
- Compression Methods Length: 1
- Compression Methods (1 method)
- Extensions Length: 127
- Extension: server\_name
- Extension: Extended Master Secret
- Extension: renegotiation\_info
- Extension: elliptic\_curves
- Extension: ec\_point\_formats
- Extension: SessionTicket TLS
- Extension: Application Layer Protocol Negotiation
- Extension: status request

The packet bytes pane at the bottom shows the raw data in hexadecimal and ASCII:

Offset	Hex	ASCII
0000	44 2b 03 d7 f6 80 20 c9 d0 e2 fc 71 08 00 45 02	D+... . ...q..E.
0010	01 03 4c 32 40 00 40 06 d5 94 0a fa 73 b4 36 bb	..L2@.@. ....s.6.
0020	62 c3 cb b6 01 bb a8 22 3d 4d d1 1d 81 03 80 18	b....." =M.....
0030	10 15 fc ea 00 00 01 01 08 0a 37 e4 25 af 05 b3	..... ..7.%...
0040	ec 35 16 03 01 00 ca 01 00 00 c6 03 03 7e 22 51	.5..... ~"Q
0050	c9 ae da 0e ab 83 c5 fd 6d 47 e7 61 15 81 e4 e7	..... mG.a....
0060	37 9a af 2d c4 7c 3f 8a 8f 3e bb 9b 7e 00 00 1e	7..-. ?. .>..~...
0070	c0 2b c0 2f cc a9 cc a8 c0 2c c0 30 c0 0a c0 09	..+./..... ,.0....
0080	c0 13 c0 14 00 33 00 39 00 2f 00 35 00 0a 01 00	.....3.9 ./5....

At the bottom of the window, the status bar reads: "No.: 408 · Time: 29.164476 · Source: 10.250.115.180 · Destination: 54.187.98.195 · Protocol: TLSv1.2 · Length: 273 · Info: Client Hello". There are "Help" and "Close" buttons at the bottom left and right respectively.

2. SSL and IPsec are both designed to provide security over the network.

- a. **The purpose of SSL is similar to IPsec, namely, security over the network and accomplish the same thing: authentication, session key, and more. They both provide encryption, integrity protection, and authentication.**
  - b. **In terms of implementation, SSL is relatively simple while IPsec is complex. SSL lives at the socket layer, as a result, SSL resides in user space, while IPsec lives at the network layer — not directly accessible from user space.**
3. Suppose we use symmetric keys for authentication and each of N users must be able to authenticate any of the other N -1 users. Evidently, such a system requires one symmetric key for each pair of users, or on the order of  $N^2$  keys. On the other hand, if we use public keys, only N key are required , but we must then deal with PKI issues.
  - a. **Kerberos requires N symmetric keys for N users. Users do not share keys with each other. Instead each user shares one key with the KDC, that is, Alice and the KDC share  $K_A$ , Bob and the KDC share  $K_B$ , Carol and the KDC share  $K_C$ , and so on. Then the KDC acts as a go-between that enables any pair of users to communicate securely with each other. The bottom line is that Kerberos uses symmetric keys in a way that doe scale.**
  - b. **In Kerberos, no PKI is required, but TTP plays a similar role as a certificated authority in a public key system. Kerberos TTP is known as the key distribution center, or KDC. Since the KDC acts as a TTP, if it's compromised, the security of the entire system is compromised.**
4. IKE has two phases, Phase 1 and Phase 2. In IKE Phase 1, there are four key options and for each of these, there i s a main mode and aggressive model
  - a. **As with the digital signature variant, the main difference from main mode is that aggressive mode does not attempt to hide identities while main mode does. Main mode MUST be implemented, while aggressive mode SHOULD be implemented. Aggressive mode should be implemented.**
  - b. **The primary advantage of the Phase 1 digital signature key option is that, it can start the protocol without waiting to obtain the other side's public key. This is true since it doesn't need to know anyone's public key — since everyone knows their private keys. However along the process of this protocol, it needs to obtain the other party's public key, in order to verify him/her as a proof, but this can be done in parallel during the process of the this protocol.**