Indian Institute of Technology Hyderabad

Department of Computer Science and Engineering

Course: CS6903 (Duration: 1 Hr) Feb 25, 2021
Marks: 68M
Roll No: Name:

1) What constitutes a cipher suite in TLS 1.2 and TLS 1.3? Explain the purposes of each of them. List out two cipher suites that you know of in TLS 1.2 and 1.3 and comment on which of the them is the strongest. 4 M

public-key algorithm: for Auth and key exchange
symmetric encryption algorithm: For Confidentiality
MAC algorithm: Integrity, by generating digest

TLSv1.2
Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA
Cipher Suite: TLS_DHE_DSS_WITH_AES_256_CBC_SHA

Note: Not limited to above two, you can compare any two version 1.2 cipher suits

TLSv1.3supports 5 ciphersuites

TLS_AES_128_GCM_SHA256
TLS_AES_256_GCM_SHA384
TLS_CHACHA20_POLY1305_SHA256
TLS_AES_128_CCM_SHA256
TLS_AES_128_CCM_8_SHA256

2) How CSR and X.509 DC differ from each other? What does CA do when CSR is received? 3M [1+1+1]

CSR is a kind of self-signed cert. Bob signs his public key with his private key. CA uses Bob's pub key to verify the integrity of the CSR and performs out-of-band background checks (refer how Let's Encrypt issues certs) before issuing DC by signing it with his private key.

3) What are the problems with 0-RTT Resumption in TLS 1.3? How those problems could be tackled? 4 M [2+2]

1) Lack of full forward secrecy

a. If session ticket keys are compromised, an attacker can decrypt 0-RTT data (encrypted by PSK) sent by the client on the first flight (but not the rest of the data in session)

b. Solution: Rotating session ticket keys regularly (weekly)

2) Replay attacks

a. Attacker replays 0-RTT data (e.g., HTTP GET/POST)

b. POST is not replay-safe

c. Solution: 1) Don't allow 0-RTT resumption on POST and allow resumptions only for some period on "safe" requests (GET) and 2) Each session tkt is valid only once

4) Given a user domain in which all the users share the following Diffie Hellman public parameters: a large prime number p and a generator g. Each user's DH public key is certified by a CA. Users communicate by performing a DH key exchange and then encrypting/decrypting messages with AES. Assume that Trudy gets hold of the CA's signature algorithm and the CA's private key, which were used to generate certificates. Can Trudy now decrypt old ciphertexts which were exchanged between two users before the CA's private key was compromised, and which Trudy had stored? Explain your answer. 4M

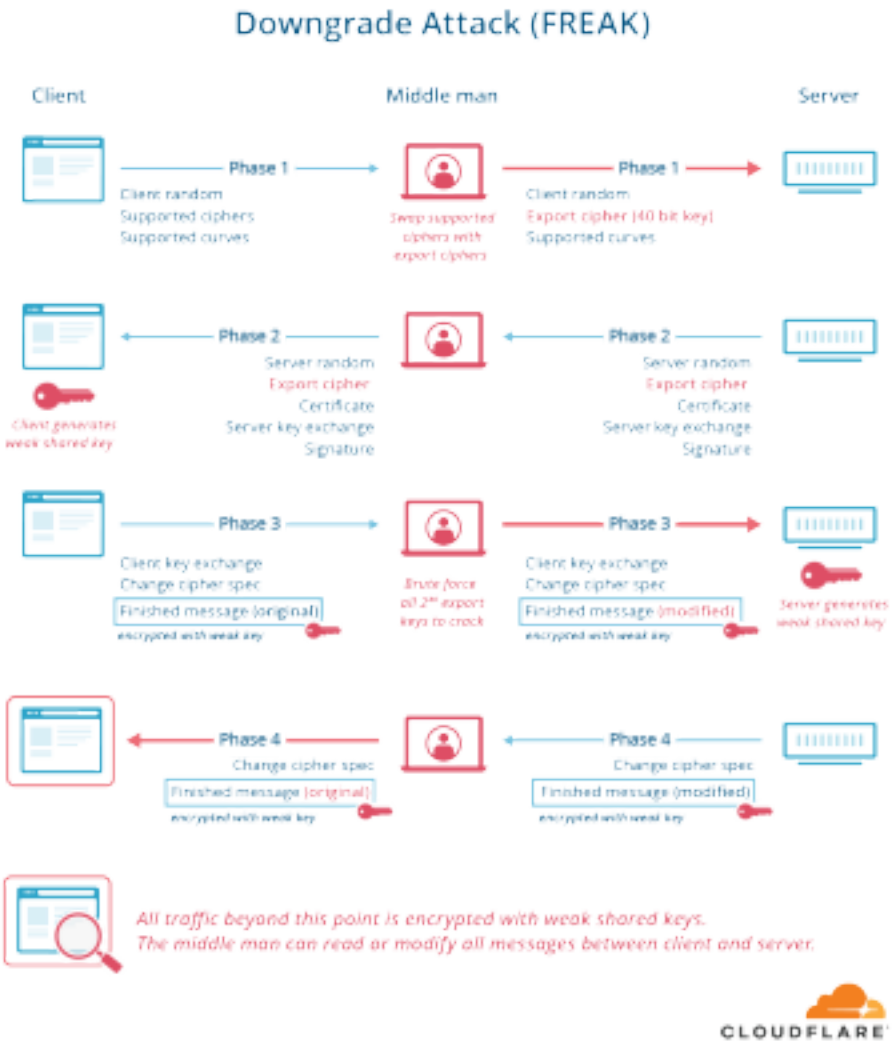
No, bcz CA's private key is used only to certify the DH public keys of users.

Trudy in fact needs DH private key of one the users to decrypt old ciphertexts and hence we say there is no perfect forward secrecy in DH/RSA based key exchange mechanisms.

5) What are the purposes of client key exchange message, client hello and server hello messages in TLS/SSL handshake? Explain one attack exploiting these messages by the man-in-the-middle (MitM) attacker with full message flow diagram? You need to clearly explain what the attacker does with the original messages of client/server. Further explain how such attacks are prevented in the latest TLS version(s)? (8 marks) (2+4+2)

Downgrade attack by forcing the SSL to use obsolete ciphers and protocols. TLS 1.3 addresses this issue by removing them from the list. Hash of all handshake msg is sent at the end of handshake to ensure there is no tampering by MITM.

POODLE (refer slide deck)



Export-strength ciphers are deprecated in tls13

a. FREAK, LogJam & CurveSwap attacks took advantage of two things: i. Support for weak ciphers in TLS 1.2
ii. Part of handshake which is used to negotiate which cipher to use is not signed in TLS 1.2

b. Difference wrt to DH in TLS 1.2: In 1.3, there is handshake key and session key while there was only one key (session) in TLS 1.2. App secret is derived from Handshake secret as the salt, DHE parameters as IKM to HKDF.

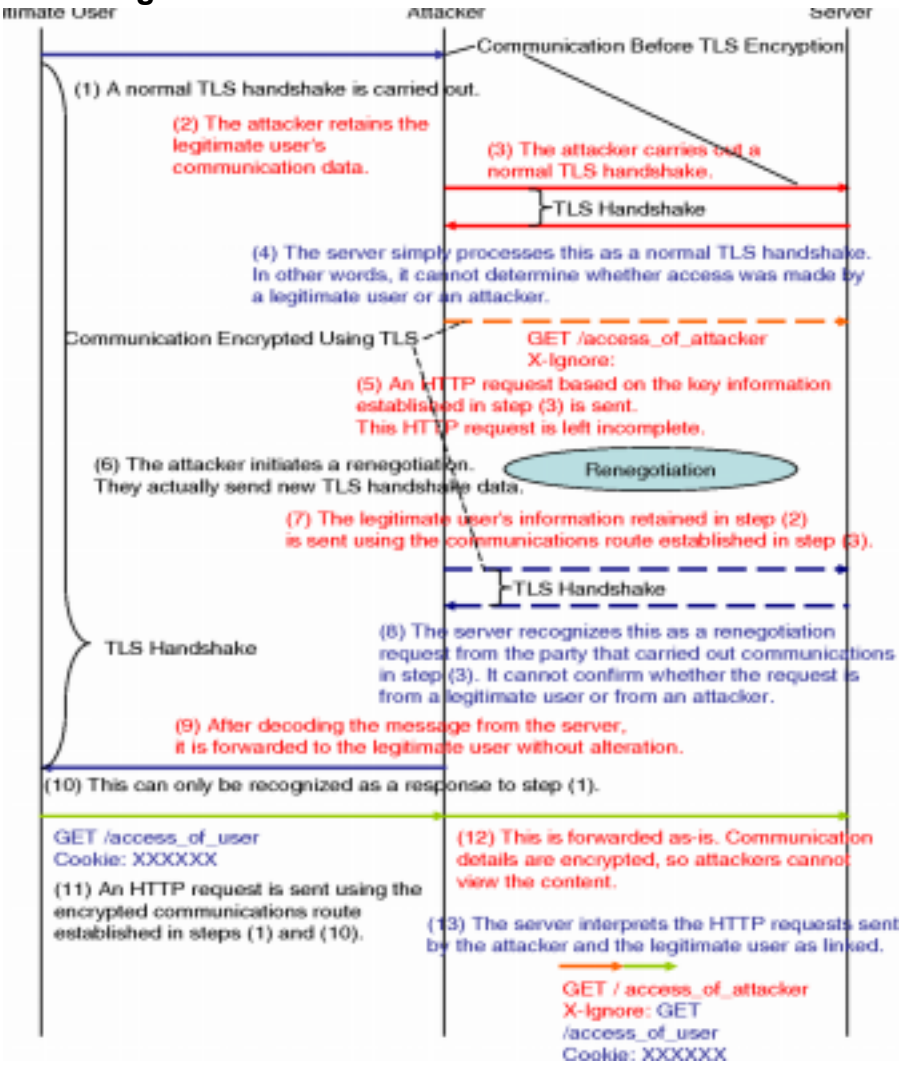
c. Signature extension in Server to Client: All previous msgs (except supported cipher suites and chosen cipher suite) are signed with private RSA key of Server to ensure no tampering. But led to FREAK, Logjam and CurveSwap in TLS 1.2. This is addressed in TLS 1.3 by signing entire handshake. So, no downgrade attack by MiTM possible.

d. Signing is done with private RSA key of Server which is 2048 or 4096 bits. So, even if session key or export_cipher key is weak, attacker can't crack private key of the server to doctor server's signature msgs

e. Finish msg: MAC of all prev msgs encryted with the server's handshake key; f. Client to Server: MAC is encrypted with pub key of Server? No: it is encrypted with secret sym handshake key derived by both parties

g. Server to Client: MAC is signed with private key of Server? No: it is encrypted with secret sym handshake key derived

TLS renegotiation Attacks



- **Mitigation techniques**

Disable TLS renegotiation esp from clients

Apache does not allow client side renegotiation

openssl s_client can be used to test if TLS renegotiation is really disabled. Sending R on an empty line triggers renegotiation.

$ openssl s_client -connect www.google.com:443 -tls1

Rate limiting of TLS handshakes

Increase server-side processing (scaling on-demand)

IETF's solution in RFC5746

Renegotiation Indication Extension to cryptographically tie renegotiations to the TLS connections they are being performed over, thus preventing this attack

6). Which of the following attacks affect all the three objectives of network security i.e., Confidentiality, Integrity and Availability?

A. Eavesdropping attacks
B. **Man-in-the-middle (MITM) attacks**
C. Denial of Service (DoS) attacks
D. None of them

7). Which of the following are used to generate a message digest by the network security protocols?

A. RSA **B. SHA-1** C. DES **D. MD5**

8). The length of the output of a cryptographic hash function is

A. the same as the input.
B. remains the same, regardless of the length of the input.
C. is much greater than that of the input, to avoid collisions.
D. is made unpredictable, to thwart to the adversary.

9) True/False:

1. Cookies can be used for phishing personal information from users (F)

2. The payload of SSL record packets is TCP packets (TCP packets are carried by SSL record packets) (F)

3. AES-ECB mode encrypts identical plaintext blocks into identical ciphertext blocks

(T)

4. Let's Encrypt issues wildcard certificates (T)

5. IITH website's X.509 certificate is an Organization Validation (OV) certificate (F)

10). Explain why key length is kept on increasing from 512-bit to 4096-bit in RSA? Order encryption, decryption, signing and sign verify operations using RSA in ascending order of computational costs? Why do you think such differences exist in computational costs in RSA? 3 M

RSA security relies on the fact that factoring a large number (n) is really hard. But, thanks to Moore's law, computers are becoming very powerful. So, key length is being increased to make it harder to launch brute-force attacks and/or factor large numbers.

Encryption<=Sign Verify < Decryption<=Signing

SInce signing is performed on message digest (relatively lower size than original message), so following order also considered

Sign verify <=Signing and Encryption < Decryption

Public exponent (e) is relatively small compared to private exponent (d). Large exponentiation in decryption, which makes it very hard to launch brute-force attacks.

11). Alice and Bob want to establish a secure communication channel using symmetric key crypto for exchanging messages with each other. Devise an efficient scheme (i.e., with less computational cost and message overhead) that offers confidentiality, authenticity and integrity by assuming Alice and Bob exchanged necessary symmetric keys out-of-band. Does your scheme offer non-repudiation? Explain. [4+1]

Alice and Bob have agreed upon MAC key K_mac and encryption key K_s out-of-band. Message m is fed into (H)MAC algo along with MAC key to generate MAC by Alice. <m+MAC> is then encrypted with encryption key K_s to achieve confidentiality. That is MAC-then-Encrypt

like in TLS 1.2. Even we can apply Encrypt-then-MAC or AEAD mechanisms here.

Bob decrypts using K_s and verifies MAC using K_mac. If the MACs match, message m is authentic and integrity protected. Usage of same MAC key by Alice helps in verifying authenticity by Bob.

Bob himself can compose a message and capable of generating MAC for it using shared K_mac and claim that the message indeed came from Alice. So, symmetric key crypto does not provide the property of non-repudiation.

12). Alice and Bob want to establish a secure communication channel using asymmetric key crypto for exchanging messages with each other. Devise an efficient scheme that offers confidentiality, authenticity, integrity and non-repudiation by assuming Alice and Bob exchanged necessary asymmetric keys out-of-band.

Alice and Bob exchanged their public keys out-of-band.

Alice encrypts m with Bob's public key, let's call it c.

Alice also generates secure digest d of m using SHA256/384. Unlike MAC, no key is involved in generating a digest.

Alice signs d with her private key, let's call it d_signed.

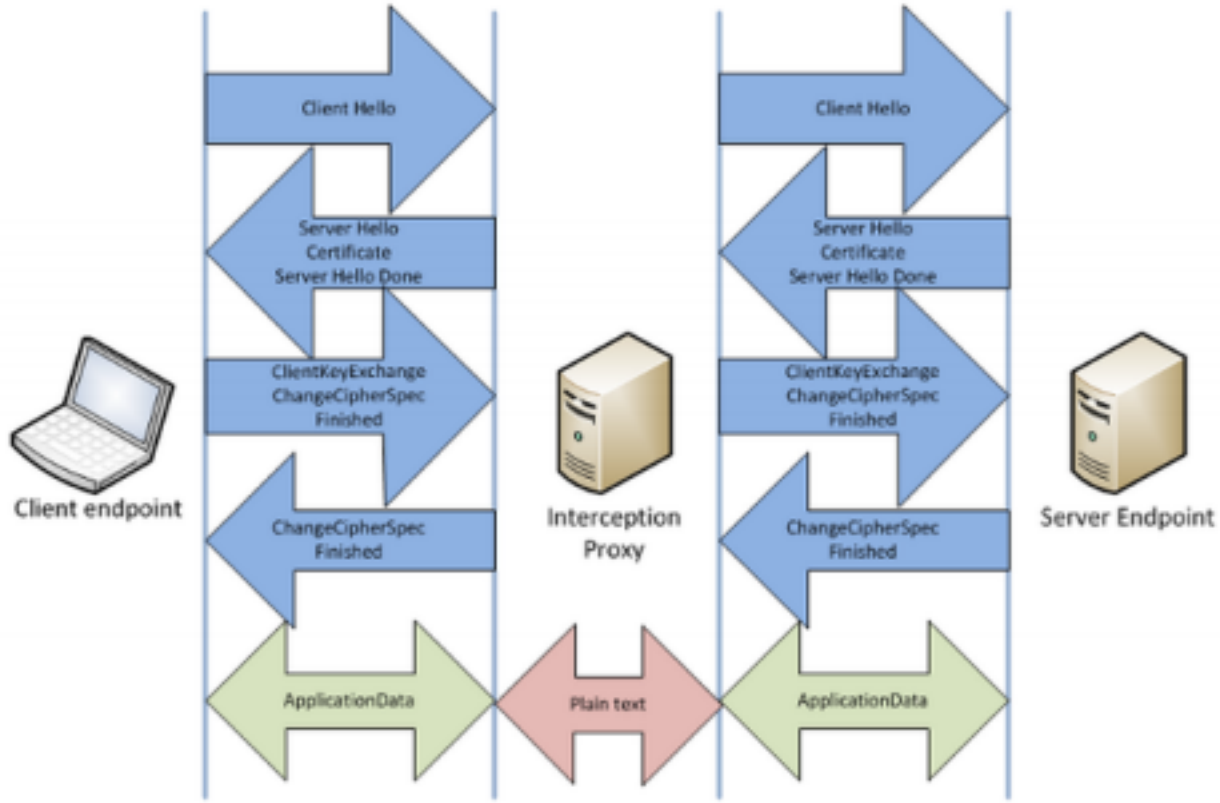
Alice sends <c, d_signed> to Bob.

This scheme offers confidentiality, authenticity, integrity and non-repudiation services.

13) . Every digital certificate contains a CA Flag in the Basic Constraints Extension that is set to 'true' if the public key being certified belongs to a CA and is 'false' otherwise. When the browser verifies a certificate chain (received after Server Hello Handshake message as a part of TLS handshake) for a webserver, it checks that all the certificates in the certificate chain have the CA flag set to 'true', except for the leaf certificate for which the CA flag is 'false'. Describe an attack that would be possible if the browser did not do this check; that is, it did not check that the CA flag is set to 'true' for the certificates in the certificate chain.

Let's assume Bob is a webserver who has his cert issued by an intermediate CA (ICA), whose cert is in turn issued by a root CA (RCA). Bob can start issuing bogus certificates to others, say to Eve who has setup a webserver which is a replica of a popular bank site. When Alice (client/browser) establishes TLS session to the bank site (requires DNS spoofing by MitM attacker), Eve's webserver (bank site) sends this bogus certificate including Bob's certificate (known as certificate chain). Since CA flag is not checked by the CertVerify() routine, it ends up thinking all the certificates from Eve's ☾ Bob's ☾ ICA's ☾ RCA's are valid.

14). An enterprise wishes to decrypt and examine all HTTPS traffic going through its border gateway. To do so it creates a CA public/private key pair and installs the CA public-key on all employee computers, so that employee browsers will trust certificates issued by this CA. The border gateway has the corresponding CA secret key. Explain the process by which the border gateway eavesdrops on an HTTPS connection established by a browser inside the enterprise and connecting to an external HTTPS web site like amazon.in. Make sure to explain exactly what the gateway does at each step of the HTTPS protocol. How can the user tell that his/her connection to, say amazon.in, is being intercepted using the existing browser UI. [10]



Proxy/GW modifies public key received from the server with a temp key pair generated, changes issuer name to its name and resigns the modified cert with his/her own private key. Since GW's cert is already installed with client's browser, verification succeeds and a TLS session is established b/w client and GW. GW establishes another TLS session with server by doing IP spoofing. When client checks certificate parameters in the browser UI, s/he can find out that GW is the one who issued the certificate of the server and GW is intercepting the session.

15). What part of handshake is encrypted in TLS 1.3, using which key? [1+1+1]

Extensions, Cert, certificate verify=sign, Finish msgs are encrypted using the handshake key to hide this critical info from MITM attackers from exploiting in future.

16). What part of handshake is signed in TLS 1.3, using which key? [3]

All of prev handshake messages hash using the Private key of Server's RSA key pair.
Sent in Certificate Verify=sign msg which is further encrypted with handshake key

17). Name key exchange algorithms that ensure perfect forward secrecy (PFS) and algorithms that do not ensure PFS.

PFS: DHE, ECDHE, PSK with ECDHE

No PFS: DH, RSA, PSK with 0-RTT data

18). RSA based key exchange in TLS: Note that MS is derived by feeding PMS and nonces of Alice and Bob as inputs to a PRF (that is known to all) by both Alice and Bob independently. Similarly, MS and nonces of Alice and Bob, and key_block size are fed as inputs to a PRF to derive key material which are split into MAC keys, session keys and IVs (IVs for AES-CBC only) by both Alice and Bob independently. To lessen the burden on Bob out of her love for Bob, Alice said that she would generate key material from MS and nonces of Alice and Bob and share it directly to Bob as part of client_key_exchange message by encrypting it with Bob's RSA public key. Trudy captured all the messages exchanged between Alice and Bob in this modified TLS session. Do you think Trudy can succeed in launching session replay attacks on Bob? Justify your answer. [4] [1+3]

NO. Bob is not in the process of generating any KM here. Though he knows nonces, since MS is not known, he can't check whether KM received is fresh or not. But when verifying FINISH msg, it fails as digest of client and server do not match as the server has generated a new server hello with new random no which is different from what was used in generating client's digest which attacker is replaying.