2024

Saturday, March 01, 2025 7:11 PM

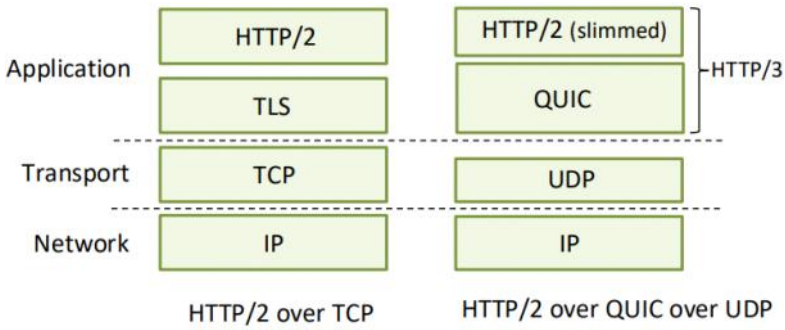
Question 1
Assume you open your browser and enter http://yourbusiness.com/about. html in the address bar. What happens until the webpage is displayed?
Provide details about the protocol(s) used and a high-level description of the messages exchanged.

1. DNS Resolution: the browser translates yourbusiness.com into an IP address by sending a request to the DNS server.
   If the browser has recently visited yourbusiness.com, it may have the IP address cached.
   If not, the DNS server responds with the IP address of the server hosting yourbusiness.com
2. TCP Connection: the browser establishes a TCP connection to the web server. Since the url begins with http:// the port used is 80.
   The browser sends a *SYN* packet to the server to initiate the connection.
   The server responds with a *SYN-ACK* packet acknowledging the request.
   The browser sends an *ACK*, completing the three-way handshake.
3. HTTP Request: the browser sends a HTTP request to the web server. This is a GET request, asking the server for the resource located at /about.html. The request also includes information about the browser and the types of content the browser can handle.
4. Server Response: The server looks for the requested file /about.html. If the file is not found it sends a 404 response code and if it is successful the server sends a 200 response code.
5. Render Page: the browser parses the received HTML
6. Close Connection: if the *Connection: keep-alive* header is not used, the connection is closed.
   The browser sends a *TCP-FIN* packet to close the connection.
   The server acknowledges with a *TCP-FIN-ACK* packet.

Consider sending over HTTP/2 a Web page that consists of one video file and three images. Suppose that the video clip is transported as 5000 frames, and each image captures four frames.
If all the video frames are sent first without interleaving, how many "frame times" are needed until all images are sent?
3 images x 4 frames/image = 12 frames for all images.
Video of 5000 frames takes 5000 frame times.
5012 frame times until the three images are sent after the video.

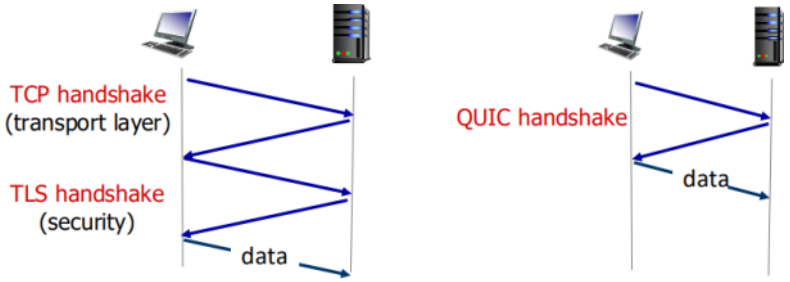
If frames are interleaved, how many frame times are needed until all three images are sent?
Sending the video frames and images together. This is allowed in HTTP/2.
In interleaving, the frames are sent in alternating order, meaning after each video frame, one of the image frames is sent if possible.
Since the images are sent alongside the video frames, the 12 image frames are sent within the first 12 frame times after starting to send the video.
After the first 12 frame times, the three images have been sent.

Highlight the differences and advantages of the QUIC protocol in terms of the protocol stack, connection establishment mechanism and efficient delivery of datagrams over existing transport layer protocols such as TCP. Use diagrams to illustrate your answer.
QUIC addresses some of the shortcomings of TCP and TLS.
QUIC operates on top of UDP rather than TCP.
Protocol Stack:

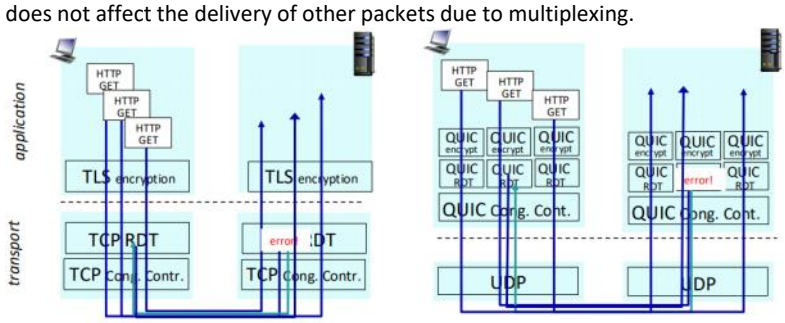


In the protocol stack, QUIC merges the operations of TCP and TLS into a single layer, providing secure transport and reliable delivery. QUIC eliminates the need for separate TLS and TCP layers, reducing overhead and simplifying the protocol stack.

Connection Establishment Mechanism:
TCP uses the three way handshake of *SYN, SYN-ACK, ACK* from browser to server to browser.
Establishing a connection in TCP therefore takes two round trip times.
QUIC can send encrypted data after one RTT as QUIC combines TLS handshake and connection establishment into one process.



Efficient Delivery of Datagrams over Existing Transport Layer Protocols:
In TCP, if a packet in the stream is lost or delayed, all subsequent packets are blocked until the lost packet is successfully retransmitted and received. Even if subsequent packets are available and ready.
In QUIC, multiple streams are used for delivering different types of data. Each stream is independent meaning that the packet loss in one stream does not affect the delivery of other packets due to multiplexing.



(a) HTTP 1.1 (b) HTTP/2 with QUIC: no HOL blocking

Draw the Finite State Machine (FSM) for the receiver side of protocol rdt3.0.

Consider distributing a file F = 10 Gbits to N peers.
The server has an upload rate of us = 1 Gbps, and each peer has a download rate of di = 200 Mbps and an upload rate of u.

For N = 10, 100 and 1,000 and u = 2 Mbps, 10 Mbps and 100 Mbps, prepare a table giving the minimum distribution time in seconds for each of the combination of N and u for both Client-Server and P2P distribution.

Client Server Distribution:
The server uploads F at Us = 1Gb/s
Each peer downloads at Di = 200Mb/s
The peer's upload speed U does not affect the download rate Di of that peer in a client-server distribution.

N	U (Mbps)	Min time for server to upload file	Min time for peer to download file	Speed
10	2	10000 Mbps	200 Mbps	50 seconds
100	2	10000 Mbps	200 Mbps	50 seconds
1000	2	10000 Mbps	200 Mbps	50 seconds
10	10	10000 Mbps	200 Mbps	50 seconds
100	10	10000 Mbps	200 Mbps	50 seconds
1000	10	10000 Mbps	200 Mbps	50 seconds
10	100	10000 Mbps	200 Mbps	50 seconds
100	100	10000 Mbps	200 Mbps	50 seconds
1000	100	10000 Mbps	200 Mbps	50 seconds

Peer to Peer Distribution:
Peers download and upload in parallel.
The server uploads F at Us = 1Gb/s
Each peer downloads at Di= 200Mb/s
In P2P, the server is not the only uploader.
Peers start sharing before the full file is downloaded.
Once peers start receiving pieces, they can upload to other peers.
The total system upload capacity is Us + N*u
Us = 1Gbps = 1000 Mbps
For 10 Gbits, File size = 10000 Mbits

No. peers	U (Mbps)	Min time for server to upload file	Min time for peer to download file	Sustainable Transfer Time
10	2	10000 Mbps	10000 / 200 = 50 seconds	10000 / 1000 + 10*2 = 9.8 seconds
100	2	10000 Mbps	10000 / 200 = 50 seconds	10000 / 1000 + 100*2 = 8.3 seconds

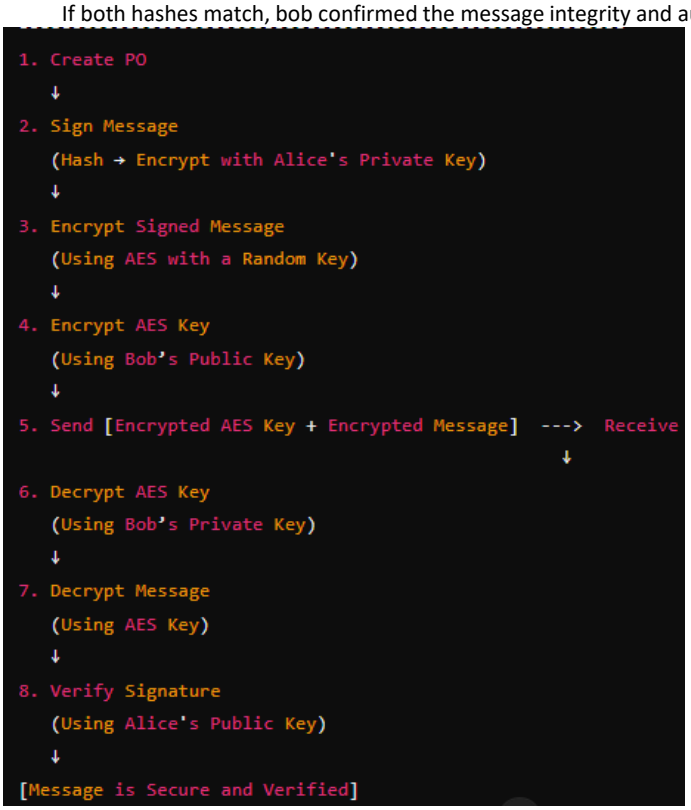
1000	2	10000 Mbps	10000 / 200 = 50 seconds	10000 / 1000 + 1000*2 = 3.3 seconds
10	10	10000 Mbps	10000 / 200 = 50 seconds	10000 / 1000 + 10*10 = 9.09 seconds
100	10	10000 Mbps	10000 / 200 = 50 seconds	10000 / 1000 + 100*10 = 5 seconds
1000	10	10000 Mbps	10000 / 200 = 50 seconds	10000 / 1000 + 1000*10 = 0.9 seconds
10	100	10000 Mbps	10000 / 200 = 50 seconds	10000 / 1000 + 10*100 = 5 seconds
100	100	10000 Mbps	10000 / 200 = 50 seconds	10000 / 1000 + 100*100 = 0.9 seconds
1000	100	10000 Mbps	10000 / 200 = 50 seconds	10000 / 1000 + 1000*100 = 0.099 seconds

Question 2

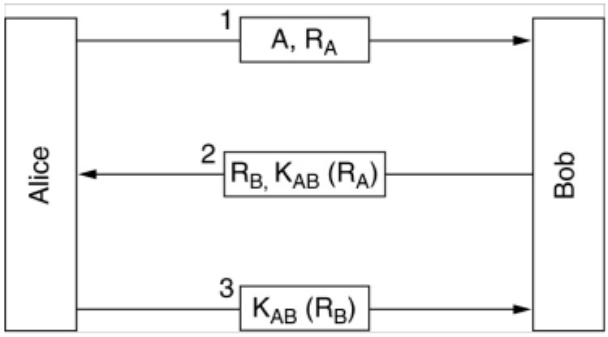
Digital Envelopes

Show with the aid of a diagram how Alice can efficiently create an "Enveloped and Signed Data" purchase order and send it securely to Bob without Trudy learning any of the details contained within the message.

1. Alice creates the purchase order.
2. Alice signs the message with a digital signature:
   She hashes the message with a cryptographic hash function.
   She encrypts the hash with her private key to generate a digital signature.
   She appends the digital signature to the message.
3. Alice encrypts the message with envelope encryption.
   She generates a random session key for the symmetric encryption.
   She encrypts the signed message using this random session key.
   She encrypts the session using Bob's public key.
   She sends the encrypted session key along with the encrypted message to Bob.
4. Bob decrypts the message.
   He decrypts the session key using his private key.
   He decrypts the signed message using the recovered session key.
5. Bob verifies the signature.
   He extracts the original hash from Alice's signature using Alice's public key.
   Bob computes the hash of the received message.
   If both hashes match, bob confirmed the message integrity and authenticity.



Why is the shortened two-way authentication protocol shown below vulnerable to a "reflection attack". Modify the diagram to show how such an attack can be mounted.



This is vulnerable to a reflection attack because Alice and Bob follow a mirrored challenge-response process.
Trudy can exploit this by reflecting Bob's challenge back to him, tricking him into authenticating her request.

1. Alice sends her identity and a challenge to Bob.
2. Bob responds with his challenge and encrypts Alice's challenge using the shared key.
3. Alice responds by encrypting Bob's challenge to complete authentication.

Trudy can impersonate Alice and initiate authentication with Bob.
When Bob responds with R_B, $K_{AB}(R_A)$, Trudy replays R_B back to Bob instead of computing the correct response.
Bob accepts his own challenge reflected back to him

Compute the two public keys and the common key for the Diffie-Hellman Key Exchange (DHKE) scheme with the parameters p = 467 and delta = 2, a = 3 and b = 5. Perform the computation for the common key for Alice and Bob.
Compute Alice's public key:
A = $delta^a$ mod p
A = 2^3mod467 = 8

Compute Bob's public key:
B = $delta^b$ mod p
B = 2^5mod467 = 32

Compute the common key:
Alice computes the common key using Bob's public key.
Bob computes the common key using Alice's public key.
Alice computes the common key as: $K_A = B^a$ mod p
Bob computes the common key as : $K_B = A^b$ mod p
$K_A = K_B$
Common key for Alice: 32^3 mod 467 = 78
Common key for Bob: 8^5 mod 467 = 78

What are the two fundamental building blocks of the proof-of-work (PoW) scheme that allow the Bitcoin protocol to be secure and popular amongst users?
Describe in detail the cryptographic techniques used to achieve "distributed consensus" in the Bitcoin network using the PoW algorithm.

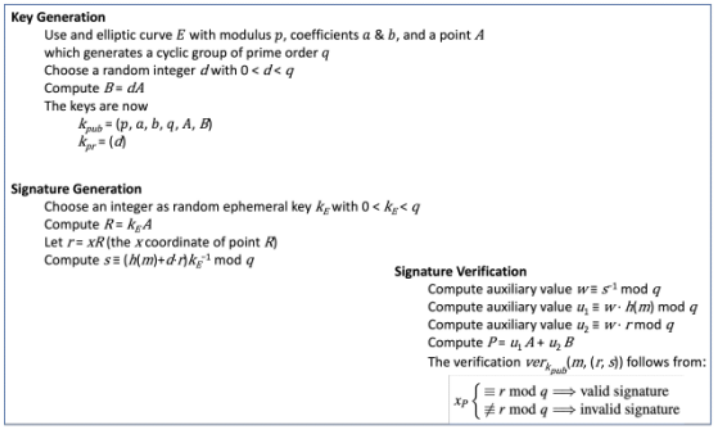
Elliptic Curve Digital Signature Algorithm

The parameters for Elliptic Curve Digital Signature Algorithm (ECDSA) are given by the curve
E : $y^2 = x^3 + 2x + 2$ mod 17,
the point A = (5, 1) of
order q = 19 and
Bob's private key d = 10.

Compute Bob's public key (B) and his signature (r,s) on the hash value h(m).

Show the verification process by Alice for the hash value h(m) = 12 and key KE = 10 by computing the values of B, R, s, w, u1, u2 and P as shown in the diagram below.

You can assume that the final computed value of P = (7, 11).



Key Generation

Use and elliptic curve E with modulus p, coefficients a & b, and a point A which generates a cyclic group of prime order q

Choose a random integer d with $0 < d < q$

Compute $B = dA$

The keys are now

$k_{pub} = (p, a, b, q, A, B)$

$k_{pr} = (d)$

Signature Generation

Choose an integer as random ephemeral key k_E with $0 < k_E < q$

Compute $R = k_E A$

Let $r = xR$ (the x coordinate of point R)

Compute $s \equiv (h(m)+d \cdot r)k_E^{-1} \bmod q$

Signature Verification

Compute auxiliary value $w \equiv s^{-1} \bmod q$

Compute auxiliary value $u_1 \equiv w \cdot h(m) \bmod q$

Compute auxiliary value $u_2 \equiv w \cdot r \bmod q$

Compute $P = u_1 A + u_2 B$

The verification $ver_{k_{pub}}(m, (r, s))$ follows from:

$$x_P \begin{cases} \equiv r \bmod q \implies \text{valid signature} \\ \not\equiv r \bmod q \implies \text{invalid signature} \end{cases}$$

For your convenience the points on the Elliptic curve are given below:

$$\begin{aligned}
2P &= (5,1)+(5,1) = (6,3) \\
3P &= 2P+P = (10,6) \\
4P &= (3,1) \\
5P &= (9,16) \\
6P &= (16,13) \\
7P &= (0,6) \\
8P &= (13,7) \\
9P &= (7,6) \\
10P &= (7,11)
\end{aligned}$$

$$\begin{aligned}
11P &= (13,10) \\
12P &= (0,11) \\
13P &= (16,4) \\
14P &= (9,1) \\
15P &= (3,16) \\
16P &= (10,11) \\
17P &= (6,14) \\
18P &= (5,16) \\
19P &= \mathcal{O}
\end{aligned}$$

Bob's public key (B): B = d*a, where a is the base point (5,1)
B = 10 * (5,1)
B = 10P
As given in the list, 10P = (7,11)

Bob's signature (r,s)
Given the hash value h(m) = 12 and KE = 10
R = KE*A = 10P = (7,11)
r = xR (the x coordinate of point R)
r = 7

s=KE ^{1}(h(m)+d*r) mod q
s=KE ^{1}(12+10*7) mod 19
s=KE ^{1}(82) mod 19
KE ^{1} mod 19 = 10^{-1} mod 19
10^{-1} mod 19 = 2 mod 19. 10 * 2 = 20. 20 is congruent to 1 mod 19

s=2*82 mod 19
s=164 mod 19
S=12

(r,s) = (7,12)

Verification by Alice:
W is congruent to s^{-1} mod q
s^{-1} mod q = 12^{-1} mod 19
12^{-1} mod 19 = 8
12*8 = 96, 96 is congruent to 1 mod 19
W = 8

U1 = w*h(m) mod 19
U1 = 8*12 mod 19
U1 = 96 mod 19
U1 = 1

U2 = w*r mod 19
U2 = 8*7 mod 19
U2 = 56 mod 19
U2 = 18

P = U1*A + U2*B
B = 10P from earlier

P = 1*(5,1) + 18*(7,11)
P = (5,1)
+ 18*10P = 180P
180 mod 19 = 9P = (7,6)

P = (5,1) + (7,6)

$y^2 = x^3 + 2x + 2 \mod 17$
Slope: 6-1/7-2 mod 17
Slope: 5/2 mod 17
Slope: $5*2^{-1} \mod 17$
$2^{-1} \mod 17 = 9$, $2*9 = 18$. 18 is congruent to 1 mod 17
Slope: 5*9 mod 17
Slope: 45 mod 19
Slope: 11

Compute the new point:
$x3 = \lambda^2 - x1 - x2 \mod 17$
$y3 = \lambda(x1 - x3) - y1 \mod 17$

$X3 = 11^2 - 5 - 7 \mod 17$
X3 = 109 mod 17
X3 = 7
Y3 = 11(5-7) - 1 mod 17
Y3 = -23 mod 17 = -6
Y3 = 17 – 6 = 11

(x3,Y3) = (7,11)

2023

Sunday, March 02, 2025 3:27 PM

Question 1
In order for a host to be able to send an HTTP request message to a Web server (www.somesite.com), the user's host must first obtain the IP address of the server.
Explain the steps through which the client obtains the IP address for such a hostname. Is it possible for an organization's Web server and Mail server to have exactly the same alias for a hostname (e.g. foo.com)?
What would be the type of Resource Record (RR) that contains the hostname for the Mail server?

Resolving Hostname:
The client checks if it has visited somesite.com and if the IP address for the hostname is stored in cache.
If not, the client sends a request to DNS to translate the hostname into an IP address.
The DNS responds with the IP address of the server hosting somesite.com.

Web Server and Mail Server with the same Alias:
An organisation's web server and mail server can have the same alias (foo.com) as the web server will use an IPv4 record and a mail server will use a Mail Exchange (MX) record which specifies mail handling for foo.com.
foo.com. - 192.168.1.1 - IP of the Web Server mail.
mail.foo.com. - 192.168.1.2 - IP of the Mail Server foo.com.
foo.com. - MX 10 mail.foo.com. - Mail Server for foo.com

The MX record tells email systems where to deliver emails sent to foo.com.

Type of the Resource Record:
The Mail Server Resource Record Type is MX Mail Exchange.
MX records specify which mail server handles delivery for a specific domain.
foo.com. MX 10 mail.foo.com.
Means that mails sent to foo.com should be delivered to mail.foo.com which then resolves them to IP

Suppose Host A sends two segments back to back to Host B over a TCP connection.
The first segment has sequence number 65; the second has sequence number 92.
How much data is in the first segment?
The first segment starts at sequence 65, the second segment starts at sequence 92.
92-65 = 27 bytes.
Suppose the first segment is lost but the second segment arrives at B. In the acknowledgement that Host B sends to Host A, what will be the acknowledgement number?
Host B will detect a gap in the received data.
TCP is cumulative so it expects data starting at sequence number 65.
Host B will send an ACK requesting the missing data
ACK number = 65.
Even though segment 92 has been received, it will not be acknowledged, Host B will continue to request sequence 65.

With the aid of an example describe the TCP "Fast Retransmit" algorithm and its advantages
Fast Retransmit detects and recovers from packet loss without waiting for a timeout by monitoring duplicate ACKs from the receiver.
This allows TCP to retransmit faster than waiting on a timeout.
Example:
1. Host A sends 4 packets with sequence numbers 1,2,3,4
2. Packet 2 is lost in transit but packets 3 and 4 arrive successfully at Host B
3. Host B receives packet 3 and sends a duplicate ACK to Host A to indicate packet 2 is missing
4. Host B keeps sending duplicate ACKs for packet 2 until packet 2 is retransmitted and received
5. After Host B acknowledges packet 2, normal transmission resumes

Consider distributing a file F = 15 Gbits to N peers.
The server has an upload rate of us = 30 Mbps, and each peer has a download rate of di = 2 Mbps and an upload rate of u.
For N = 10 and 100 and u = 300 Kbps and 700 Kbps calculate the minimum distribution time for both client-server and P2P configurations.
Client Server Configuration:
15 Gbits = 15000 Mbits

N	U	Min time for server to upload file	Min time for peer to download file	Speed
10	300	30Mbps	2Mbps	15000/30 = 500 seconds
10	700	30Mbps	2Mbps	15000/30 = 500 seconds
100	300	30Mbps	2Mbps	15000/30 = 500 seconds
100	700	30Mbps	2Mbps	15000/30 = 500 seconds

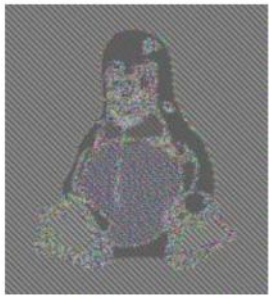
Peer to Peer Configuration:
15000 / 30 + N*u
1kb = 1000mb
300kbps=0.3mbps
700kbps=0.7mbps

No. peers	U (Mbps)	Min time for server to upload file	Min time for peer to download file	Sustainable Transfer Time
10	0.3	30Mbps	2Mbps	15000 / 30 + 10*0.3 = 454 seconds
10	0.7	30Mbps	2Mbps	15000 / 30 + 10*0.7 = 405 seconds
100	0.3	30Mbps	2Mbps	15000 / 30 + 100*0.3 = 250 seconds
100	0.7	30Mbps	2Mbps	15000 / 30 +100*0.7 = 150 seconds

What type of attacks is the Electronic Code Book (ECB) mode in symmetric key encryption vulnerable to?
ECB made in symmetric key encryption is vulnerable to pattern preservation attacks and replay attacks as it is deterministic in nature.
In ECB mode:
- Plaintext is divided into fixed sized blocks.
- Each block is encrypted separately using the same key.
- Identical plaintext blocks produce identical ciphertext blocks.

E.g. when encrypting an image, identical blocks of colour will be encrypted with the same key.

Original Image



Encrypted with ECB Mode

Explain with the help of examples why the Cipher Block Chaining (CBC) and Output Feedback (OFB) modes can help overcome the vulnerabilities of ECB mode.
In CBC mode, each block of plaintext is XORed with the previous ciphertext block before encryption.
The first plaintext block is XORed with a random initialisation vector.
In CBC mode, identical plaintext blocks do not produce identical ciphertext blocks due to the chaining nature.

In OFB mode, a key stream is generated using the encryption algorithm applied to an initialisation vector.
The key stream is then XORed with the plaintext.
In OFB mode, each key stream block depends only on the previous *encryption* output, not the plaintext itself.
Since the key stream is independent of the plaintext, this also overcomes the weaknesses of ECB mode.

Why is the OFB mode useful in building a "synchronous stream cipher"?
OFB mode is useful in building a synchronous stream cipher as the key stream is generated independently of the plaintext, making it resistant to pattern analysis, unlike ECB.
Unlike CBC, an error in transmission only affects the corresponding bit and not the entire block.
The encryption process can continue without waiting for the entire plaintext.

Question 2
What are the differences between message confidentiality and message integrity? Show how message integrity can be achieved using symmetric and asymmetric key cryptographic techniques.
Message Confidentiality: ensures that the content of a message is kept secret from unauthorized parties. It protects the message from being read or accessed by anyone other than the intended recipient. It is achieved by encryption that can be decoded by someone with the proper key.
Message Integrity: ensures that the message has not been altered or tampered with during transmission. Ensures the message received is the same as the message sent. It is achieved by cryptographic hash functions, digital signatures or message authentication codes.

Using Symmetric Key Cryptography - MAC:
In symmetric key cryptography, the sender and receiver share the same secret key. One way is by using a message authentication code.
The sender creates a hash of the message and combines it with a secret shared key using a MAC algorithm.
The combined result is send along with the original message.
The receiver recomputes the hash of the message and verifies it against the MAC using the shared secret key.
If the computed MAC matches the received one, the message is considered intact and untampered.

Using Asymmetric Key Cryptography – Digital Signatures:
The sender computes a hash of the message to create a message digest
The sender then encrypts this message digest using their private key, creating a digital signature.
The sender sends the original message along with the digital signature
The receiver, upon receiving the message and the signature, computes the hash of the message.
The receiver decrypts the signature using the sender's public key to retrieve the original hash
If the computed hash matches the decrypted hash, the integrity of the message is confirmed

Describe some of the components that comprise modern day block ciphers? In particular describe with the aid of an example the Vigenére Cipher.
Modern block ciphers such as AES operate on fixed-size blocks of plaintext using transformations.
Components include:

- Substitution: non-linear transformations to replace parts of the data
- Permutations: bits are shuffled
- Key expansion: a set of round keys is derived from the original secret key
- XOR operation

Example of the Vigenére Cipher:
Plaintext: LUCIABROWN
Key: COMPSCI (3,15,13,16,19,3,9)
Ciphertext: OJPYTEARLA

Suppose that a system uses a Public Key Infrastructure (PKI) based on a tree-structured hierarchy of Certification Authorities (CAs). Alice wants to communicate with Bob, and receives a certificate from Bob signed by a CA X after establishing a communication channel with Bob. Suppose Alice has never heard of X but knows the public-key of the root CA. Explain in detail what steps Alice needs to take to verify that she is talking to Bob?
In Public Key Infrastructure, Alice needs to verify that the certificate bob has sent her is valid and trustworthy.
The certificate contains:

- Bob's public key
- Certification Authority X's digital signature
- Issuer information
- Certificate validity period

The certificate that bob has provided Alice is signed by CA X, which may be signed by another CA, and so on.
The entire certificate chain needs to be validated to the root.
Alice knows the public key of the *root* CA but this may not be the public key of CA X.
She needs to look for CA X's certificate in the certificate chain and verify that the signature on Bob's certificate matches that of CA X's public key.
Once Alice has the certificate for CA X, she needs to verify that it has been signed by a higher-level CA. This process repeats up the chain. At each step, Alice will verify that the certificate's signature is correct and that the certificate has not expired.
Alice already knows the public key of the root CA. Once she has verified that each certificate in the chain is correctly signed by the CA above it, she can trust that the root CA's public key is valid.

Summary of Steps for Alice

1. **Obtain CA X's certificate** (if not already available).
2. **Verify that Bob's certificate is signed by CA X.**
3. **Verify that CA X's certificate is signed by a trusted higher-level CA.**
4. **Continue verifying certificates upwards** until reaching the root CA.
5. **Verify the root CA's certificate** using the known public key.
6. **Ensure Bob's certificate is valid** (check expiration and revocation status).
7. **Verify Bob's identity** by matching the subject field in his certificate.

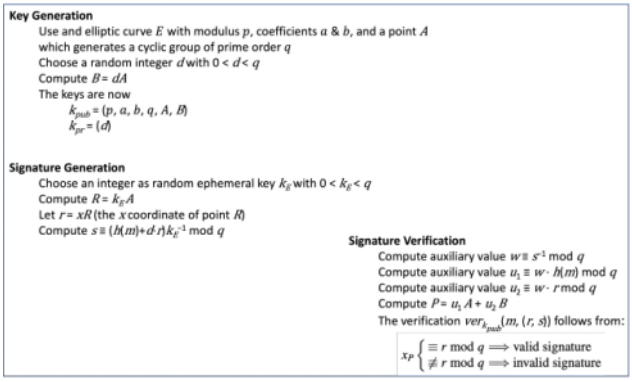
To show that you understand the security of the RSA algorithm, find d if you know that e = 17 and n = 187.
Public exponent e = 17
Modulus n = 187

Step 1 – Factor n:
187 factorises down into two primes: 17 * 11
The prime factors of n are p = 11, q = 17

Step 2 – Calculate φ(n):
φ(n) = (p-1)(q-1)
= (10)(16) = 160
φ(187)=160.

*Step 3 – Find d, such that d*e is congruent to 1 mod φ(n)*:
17 * d ≡ 1mod160
This is equivalent to finding the modular inverse of 17mod160.
160 = 17 * 9 + 7
17 = 7 * 2 + 3
7 = 3 * 2 + 1
3 = 3 * 1 + 0
The gcd is 1, so 17 and 160 are coprime, meaning a modular inverse exists.
1 = 5 * 160 – 47 * 17
1 = -47*17 + 5*160
-47*17 ≡ 1mod160
d= -47 mod 160
d=160−47=113
Thus, the private key exponent d is 113.

The parameters for Elliptic Curve Digital Signature Algorithm (ECDSA) are given by the curve E : y 2 = x 3 + 2x + 2 mod 17. the point A = (5, 1) or order q = 19 and Bob's private key d = 7. Compute Bob's public key (B) and his signature (r,s) on the hash value h(m). Show the verification process by Alice for the hash value h(m) = 26 and key KE = 10. You can assume that the value of P = (7, 11).



$2P = (5,1) + (5,1) = (6,3)$
$3P = 2P + P = (10,6)$
$4P = (3,1)$
$5P = (9,16)$
$6P = (16,13)$
$7P = (0,6)$
$8P = (13,7)$
$9P = (7,6)$
$10P = (7,11)$
$11P = (13,10)$
$12P = (0,11)$
$13P = (16,4)$
$14P = (9,1)$
$15P = (3,16)$
$16P = (10,11)$
$17P = (6,14)$
$18P = (5,16)$
$19P = \mathcal{O}$

Bob's public key:
B = dA
B = 7P
B = (0,6)

Bob's signature (r,s):
R = KE*A
R = 10P
R = (7,11)
r= 7

S = (h(m)+ d*r)KE^{-1} mod q
S = (26 + 7*7)10^{-1} mod 19
10^{-1} mod 19 = 2, 10*2 = 20, 20 is congruent to 1 mod 19

S = (26 + 7*7)*2 mod 19
S = 150 mod 19
S = 17

(r,s) = (7,17)

Alice's signature verification
W = s^{-1} mod q
W = 17^{-1} mod 19
W = 9. 17*9 mod 19 is congruent to 1 mod 19

U1 = w * h(m) mod q
U1 = 9 * 26 mod 19
U1 = 234 mod 19
U1 = 6

U2 = w * r mod q
U2 = 9 * 7 mod 19
U2 = 63 mod 19
U2 = 6

P = U1*A + U2*B
U1*A = 6P = (16,13)
U2*B = 6*7P, 6*7P = 42P, 42P mod 19 = 4P = (3,1)

Find new point (x3,y3): $y^2 = x^3 + 2x + 2$ mod 17
Slope: 1-13/3-16 = -12/-13 = 12/13mod 17
12*13^{-1} mod 17
13^{-1} mod 17 = 4
12*4 mod 17
48 mod 17 = 14

$x3 = \lambda^2 - x1 - x2 \bmod 17$
$y3 = \lambda(x1 - x3) - y1 \bmod 17$

$X3 = 14^2 - 16 - 3 \bmod 17$
X3 = 7
Y3 = 14(16-7)-13 mod 17
Y3 = 11

(x3,y3) = (7,11)

2022

Monday, March 03, 2025 9:41 PM

Question 1

Explain how the transmission control protocol (TCP) uses the services of an unreliable network layer to provide reliable end-to-end communications?

Reliable TCP operates over the unreliable IP. IP uses a best-effort delivery, meaning that packets may be lost, duplicated or arrive out of order. TCP overcomes this in the following ways:

- Three-Way Handshake: ensures that both the sender and receiver are ready for communication. First the sender sends a *SYN* packet to the receiver asking to synchronise, the receiver sends back a *SYN-ACK* acknowleding this request and finally the sender confirms with an *ACK*.
- Acknowledgement Retransmission: each transmitted segment must be acknowledged by the receiver. The receiver will send a double acknowledgement if a packet arrives out of sequence. For example if the sender is sending packets with sequence numbers 1,2,3,4,5 and packet 2 is lost, packets 3 and 4 can still arrive at the receiver but the receiver will continue to send a duplicate acknowledgement requesting packet 2 to be retransmitted.

Show with the aid of an example how TCP can correctly demultiplex packets arriving from two hosts (A and B), who by coincidence happen to pick the same source port number (5099), and are communicating with the same web server on port 80.

Even though the source ports are the same, TCP can still correctly deliver the packets using the below four-tuple:

Host	Source IP	Source Port	Destination IP	Destination Port
A	192.168.1.10	5099	192.168.1.100	80
B	192.168.1.20	5099	192.168.1.100	80

When a packet arrives at the web server 192.168.1.100, TCP looks at the four-tuple to determine the correct connection.

Even though Host A and Host B have the same source port, their source IP will be different and TCP is built on top of IP.

The server will maintain separate connection states for (192.168.1.10, 5099, 192.168.1.100, 80) and (192.168.1.20, 5099, 192.168.1.100, 80).

This ensures that packets are delivered in the correct session.

Consider transferring an enormous file of L bytes from Host A to Host B. Assume a maximum segment size (MSS) of 536 bytes.

What is the maximum value of L such that the TCP sequence numbers are not exhausted?

TCP uses a 32-bit sequence number from 0 to 2^{32}-1, this is 4,294,967,295 bytes.

The maximum amount of data that can be sent before TCP sequence numbers is exhausted is 4.29GB

For the L you obtain in (i), find how long it takes to transmit the file? Assume that a total of 66 bytes of transport, network and data-link header are added to each segment before the resulting packet is sent out over a 155 Mbps link. Ignore flow and congestion control, and assume that Host A can pump out the segments back to back and continuously.

MSS = 536 bytes

Overhead = 66 bytes

Packet size = 536+66=602 bytes

Bandwidth = 155Mbps = 155×10^6 bytes/second

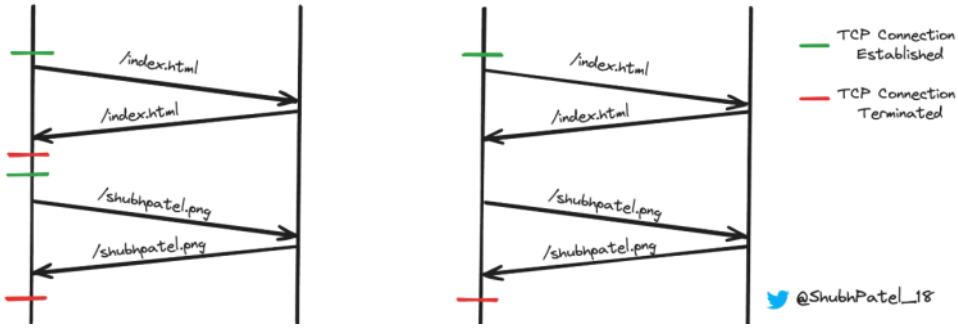
4,294,967,295 / 536 = 8,014,896 segments.

8,014,896 segments * 602 bytes per segment = 4,822,967,792 bytes

4,822,967,792 * 8 bits in one byte =38,583,742,336 bits

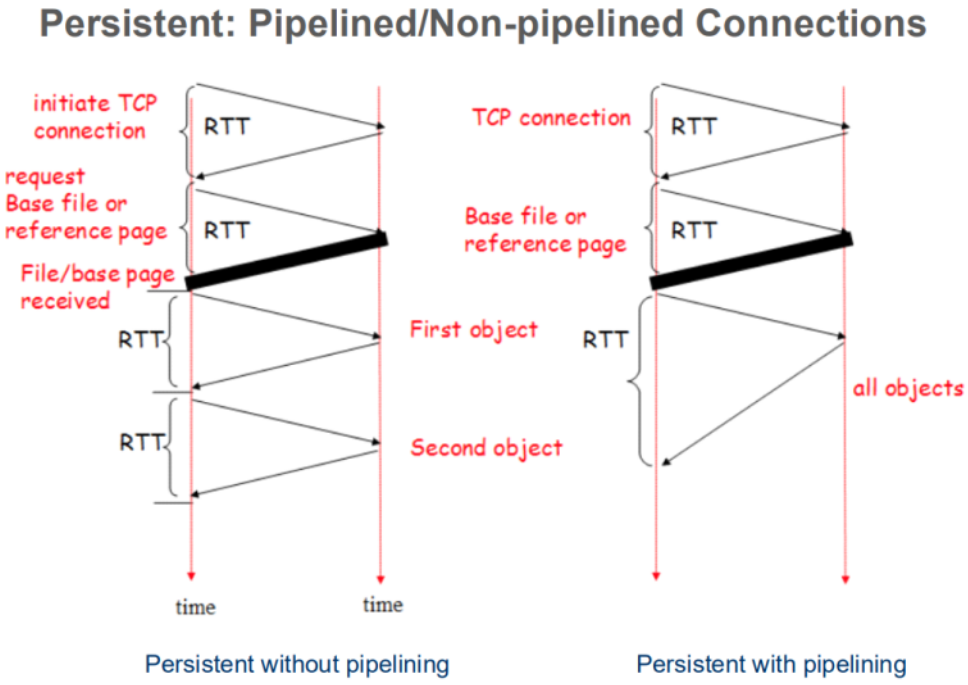
Total bits / bandwidth:
38,583,742,336 / 155x10^6 = 248.28 seconds

With the aid of a diagram distinguish between "persistent" and "non-persistent" HTTP connections in terms of the round trip time (RTT) and TCP overhead.
In non-persistent HTTP connections, at most one object is sent over TCP before the connection is closed. Downloading multiple objects requires multiple connections.
In persistent HTTP, multiple objects can be sent over a singular HTTP connection between client and server. The advantage of this is a more efficient transfer between client and server.



Therefore, non-persistent HTTP takes two RTTs per object, with OS overhead for each TCP connection/closure.
Persistent HTTP leaves the connection open after sending a response. So there can be as little as one RTT for all referenced objects.

What improvements can "pipelining" bring to a persistent HTTP connection in terms of the RTT?
The default mode for HTTP is a persistent connection with pipelining.
With pipelining, the TCP connection is established then a base file/reference page is requested and received.
In pipelining, all requested objects can be requested and received in a single RTT, instead of each individual object requiring a RTT each.



Imagine a peer-to-peer network (P2P) where N users want to communicate in an authenticated and confidential manner without a centralized trusted third party (TTP).
How many keys are collectively needed if symmetric key algorithms are deployed?
With symmetric key encryption, the sender uses a secret key to encrypt the message and the receiver uses the same key to decrypt the message. This can be N people, but everyone must have the same key.

If everybody uses the same symmetric key, then only one key is required.

However, if the P2P network wants every pair of people in the N users to be able to communicate securely without the rest of the N-2 people knowing. This will require a key for each combination of two people who can communicate.
N C 2 keys are required.
e.g. for 5 users: 5C2 = 10 keys required. As each user must store N-1 keys

How are the numbers changed if we make use of a key distribution center (KDC)?
Instead of each user having to store N-1 keys, using a KDC means that each user only has to store their own key.
For N users there will be N keys.
The user needs a key to communicate with the KDC, then to communicate with another user the KDC will generate session keys.

How many keys are needed if we make use of asymmetric key algorithms?
In asymmetric key cryptography, each user has a key-pair. A public key shared with everyone and a private key kept secret.
Therefore there will be 2N keys needed in the system collectively when using asymmetric key cryptography.

Question 2
One of the most attractive applications for public-key cryptography is the establishment of a secure "session key".
In practice it is desirable that both communication parties influence the selection of the session key, so as to prevent one party from choosing a weak key.
Develop a protocol in which both Alice and Bob who possess a pair of public/private keys of the RSA cryptosystem are able to influence the selection of the session key.
Alice has an RSA key-pair:
Public key (eA)
Private key (dA)
Bob has an RSA key-pair:
Public key (eB)
Private key (dB)

Alice selects a random value rA.
Alice encrypts rA with Bob's public key eB. Call this encrypted value cA.
Alice sends cA to Bob.

Bob selects his own random value rB.
Bob decrypts cA using his private key dB. This gives Bob rA.
Bob encrypts rA XOR rB using Alice's public key eA. Call this encrypted value cB.
Bob sends cB to Alice.

Alice decrypts cB using her private key dA. This gives Alice rA XOR rB
Both Bob and Alice now have rA and rB and can derive the session key from the hash of rA and rB

Using the Miller-Rabin primality test show how one can assume with a high probability that the number 269 is a prime number, given that the decomposition of an odd prime candidate can be represented by $p-1=2^u \cdot r$, where r is odd.
P = 269
P-1= 268
$268 = 2^u * r$ where r is odd
268 = 2 * 134
134 = 2 * 67 (ODD), found after 2 iterations

$269 - 1 = 2^2 * 67$

Choose a random base a such that $2 \leq a \leq p-2$.
A = 2

Compute a^r mod p:
2^{67} mod 269 = 187.
187 is not congruent to 1 or –1.

Compute a^{2r} mod p:
2^{134} mod 269 = 268. 268 is congruent to –1 mod 269.
Since this result is $-1 \bmod 269$, we pass the Miller-Rabin test for base a=2.

Let Z_{269} be a finite cyclic group. Find the number of primitive roots in Z_{269}. Find the least primitive root of Z_{269}.

Given the elliptic curve E over Z_{29} and the base point P = (8, 10):
E : $y^2 = x^3 + 4x + 20 \bmod 29$
Calculate the following point multiplication k · P (where k = 2) using the formulae provided below:

$$x_3 = s^2 - x_1 - x_2 \bmod p$$
$$y_3 = s(x_1 - x_3) - y_1 \bmod p$$

where

$$s = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} \bmod p \text{ ; if } P \neq Q \text{ (point addition)} \\ \frac{3x_1^2 + a}{2y_1} \bmod p \text{ ; if } P = Q \text{ (point doubling)} \end{cases}$$

You are required to show the extended Euclidean algorithm (EEA) calculation for computing the multiplicative inverse.

What are the advantages of a virtual private network (VPN) over a dedicated private secure Internet link?
Describe the functionality of the IPsec "Transport" and "Tunnel" modes.
In particular, describe with the aid of a diagram how the header and payload of an IP datagram can be protected by deploying IPsec in "Tunnel Mode with ESP" (ESP - encapsulation security payload).
Note that you are required to identify the relevant IPsec header and trailer fields.

2019

Wednesday, April 16, 2025 11:50 AM

How TCP uses the services of an unreliable network layer to provide reliable end-to-end communication

Connection Establishment:
TCP starts a 3-way handshake to establish connection.
Clent sends a SYN packet to the server.
The server responds with a SYN-ACK packet.
Finally, the client sends an ACK.
Segmentation & Sequencing:
TCP breaks data into segments and assigns each segment a sequence number.
These numbers allow the receiver to reorder out of order segments, detect missing packets, etc.
ACKs & Retransmissions:
Each received segment is ACKnowledged by the receiver, if a packet is lost, the TCP receiver will continue to send an ACK for the last good packet until the missing packet is received.

Show how TCP can correctly demux packets arriving from two hosts who by coincidence pick the same source port 5099 and are communicating with the same web server on port 80.

TCP is a 4-tuple: <Source IP, Source Port, Destination IP, Destination Port>
Host A: <192.161.1.0, 5099, 182.34.1.3, 80>
Host B: <192.161.1.1, 5099, 182.34.1.3, 80>
Different connections due to different Source IP. As long as one of the elements of the tuple is different.

NAK-free reliable data transfer additions to make it NAK-free:

Seuqnece numbers are assigned to each packet (0 or 1)
Instead of NAK on corrupted packets, the receiver sends an ACK on the last good packet
Timeouts on the sender side to deal with corrupted/lost ACKs

Transferring an enormous file L. Assume a maximum segment size of 536 bytes.

What is the maximum value of L such that TCP numbers are not exhausted

TCP numbers range from 0 to 2^{32}-1 which is 4,294,967,296 bytes.
The maximum amount of data that can be sent before TCP number exhaustion is 4,294,967,296 bytes.

How long does it take to transmit the file with a total of 66 bytes of overhead and being sent over a 155Mbps link.

4,294,967,296 bytes / 536 bytes = 8012998 total packets.
536 + 66 = 602 bytes size for one packet.
602 * 8 = 4816 bits per packet
8012998 * 4816 = 3.859059837 x 10^{10} total bits to be sent
3.859059837 x 10^{10} / 155 x 10^{6} speed = 248.97 seconds

Persistent vs. Non-Persistent HTTP

In non-persistent HTTP, at most one object is sent before the TCP connection is closed.
In persistent HTTP, multiple objects can be sent over a single HTTP connection.
Non-persistent HTTP takes 2 RTTs per object with TCP overhead.
Persistent HTTP can take as little as 1 RTT for all referenced objects.

What improvements can pipelining bring to a persistent connection?

Pipelined HTT can reduce RTTs required. In a pipelined connection, all referenced objects are sent in one RTT

For N users, how many keys are needed if symmetric key algorithms are used
One key is needed for every pair of users. (NC2) keys required. N(n-1)/2 keys needed.
How many keys are needed if a KDC is used?
Speaking to a KDC, each user has a single shared key with the KDC. N keys are required.
How many keys are needed if asymmetric key algorithms are used?
Each user maintains a public key and a private key. 2N keys required

Protocol where two people who possess a pair of RSA (public/private) keys can both influence the session key.
Diffie-Hellman key exchange allows two users with a public and private key can compute a common session key.
P = a large prime number.
Alice computes the common session key as (Bob's public key)$^{\text{(Alice's private key)}}$.
Bob computes the common session key as (Alice's public key)$^{\text{(Bob's private key)}}$.

Miller-Rabin primality test to prove that 269 is a prime number
$269-1 = 2^u r$, where r is odd.
$268 = 2^2 67$
If a^r is congruent to 1 mod 269, 269 is prime
1^{67} =
2^{67} =
3^{67} =
4^{67} =
5^{67} =

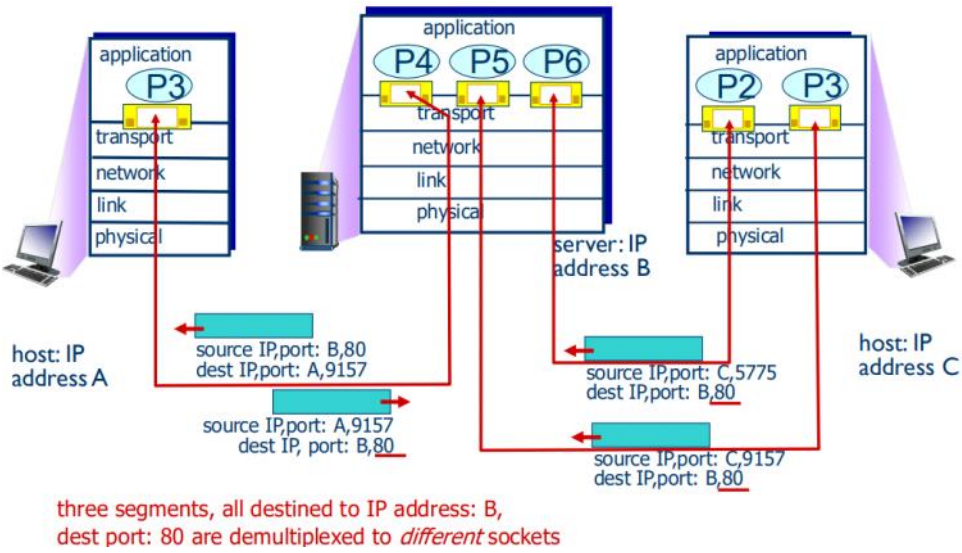
20 Jan 2025

Monday, January 20, 2025 9:01 AM

Transport Layer:
Provide logical communication between app processes running on different hosts
Transport protocols run in end systems
– Send side: breaks app messages into segments, passes to network layer
– Rcv side: reassembles segments into messages, passes to app layer
More than one transport protocol available to apps

Internet Transport-Layer Protocols
TCP
- Reliable, in-order delivery
– Congestion control
– Flow control
-Connection setup (needs to know the port numbers)
UDP
- Unreliable, unordered delivery
– No-frills extension of “best effort” IP
Services not available
-No bandwidth guarantees
-Delays guaranteed

Multiplexing and Demultiplexing
How Demultiplexing Works
Host uses IP addresses and port numbers to direct segment to appropriate socket
Each datagram has source IP address, destination IP address
Each datagram carries one transport-layer segment
Connection-oriented demux
TCP socket identified by 4-tuple
– Source IP address
– Source port number
– Dest IP address
– Dest port number
Receiver uses all four values to direct segment to appropriate socket
Web servers have different sockets for each connecting client
Server host may support many simultaneous TCP sockets



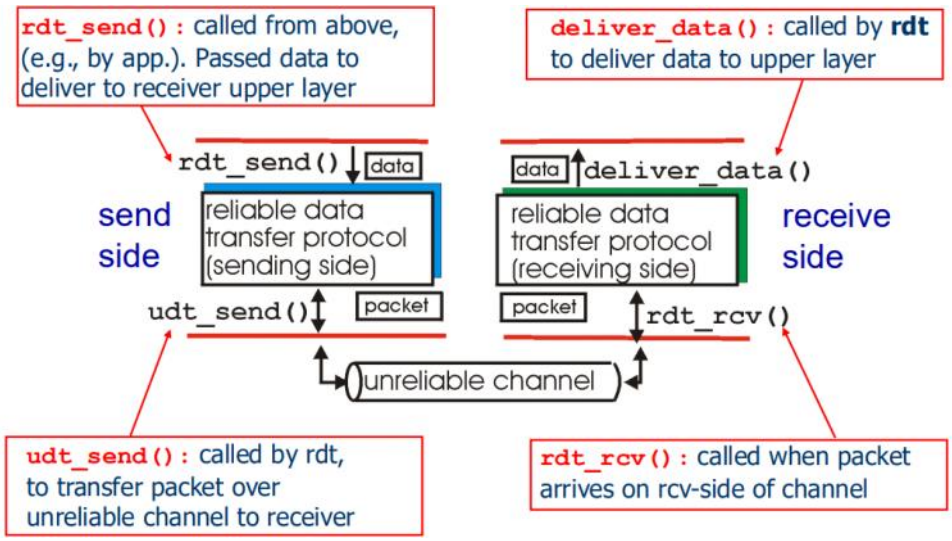
UDP

"No frills", "bare bones" Internet transport protocol
Connectionless - Each UDP segment handled independently of others
Streaming multimedia apps (loss tolerant, rate sensitive)

27 Jan 2025 – Transport Layer

Monday, January 27, 2025 8:57 AM

Principles of Reliable Data Transfer:
Characteristics of the unreliable channel will determine complexity of *reliable data transfer* protocol
RDT



We will incrementally develop sender/receiver sides of RDT protocol
Consider only unidirectional data transfer
But control info will flow on both directions
Use finite state machines to specify sender/receiver

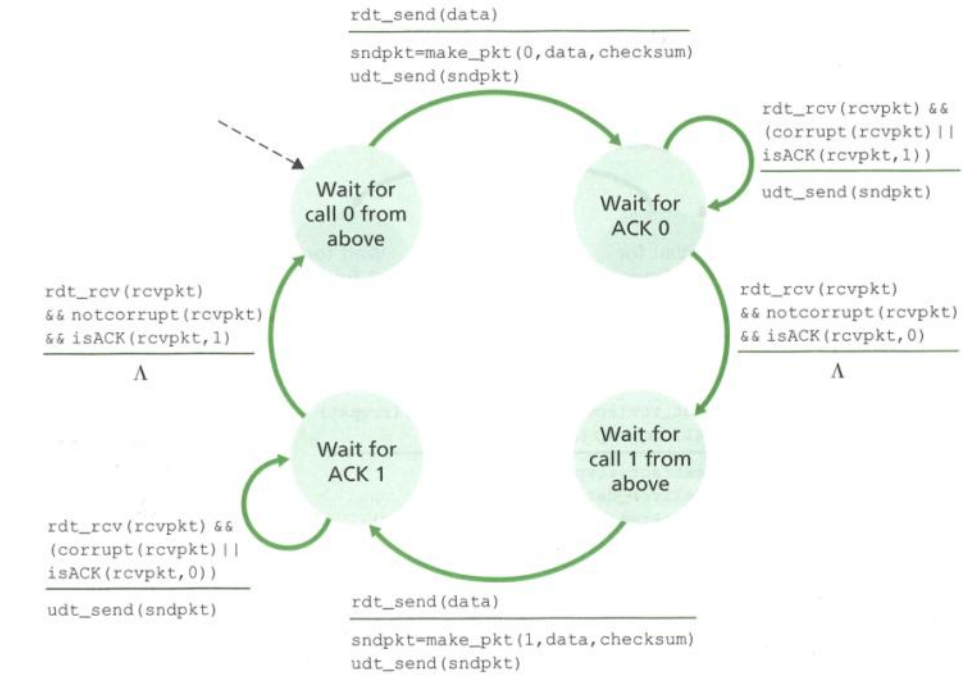
RDT over a reliable channel
Underlying channel perfectly reliable
No bit errors
No loss of packets
Separate FSMs for sender and receiver

Channel with bit errors
Checksum to detect bit errors
How to recover from errors (ACKs, receiver explicitly tells sender that packet received OK, sender can drop that packet from its buffer)
NAK (receiver explicitly tells sender that packet had errors, sender immediately retransmits packet on NAK)
New mechanisms in RDT2.0: error detection, feedback

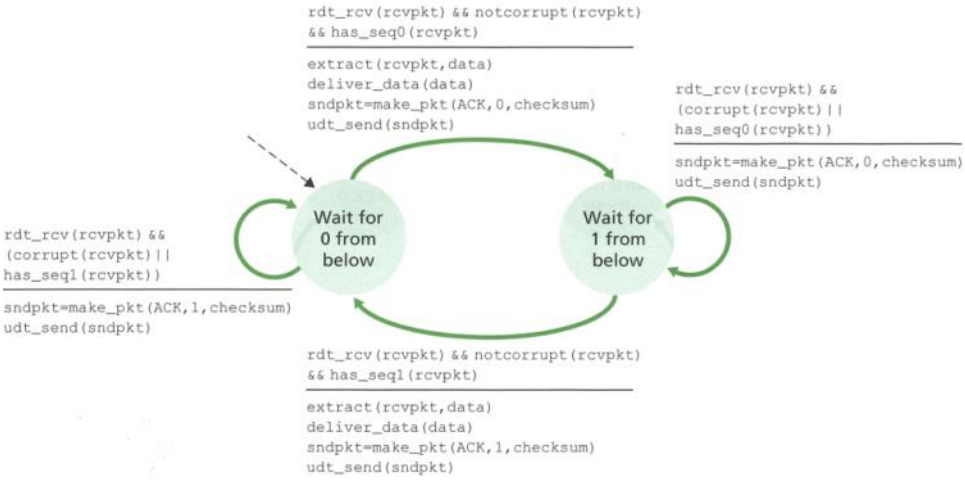
RDTv2.0 Fatal Flaw
What if the ACK/NAK packet itself is corrupted
Sender cannot just retransmit, possible duplication
Handling duplicates with sequence numbers
Sender retransmits current packet if ACK/NAK is corrupted, sender adds sequence number to each packet
Receiver discards duplicate packets and does not push them up to the application layer

RDT NAK-free protocol
Same functionality as RDT
Instead of NAK, receiver sends ACK for last packet received OK
Duplicate ACK at the sender has the same effect as a NAK, sender will retransmit the current packet

rdt2.2: Sender FSM

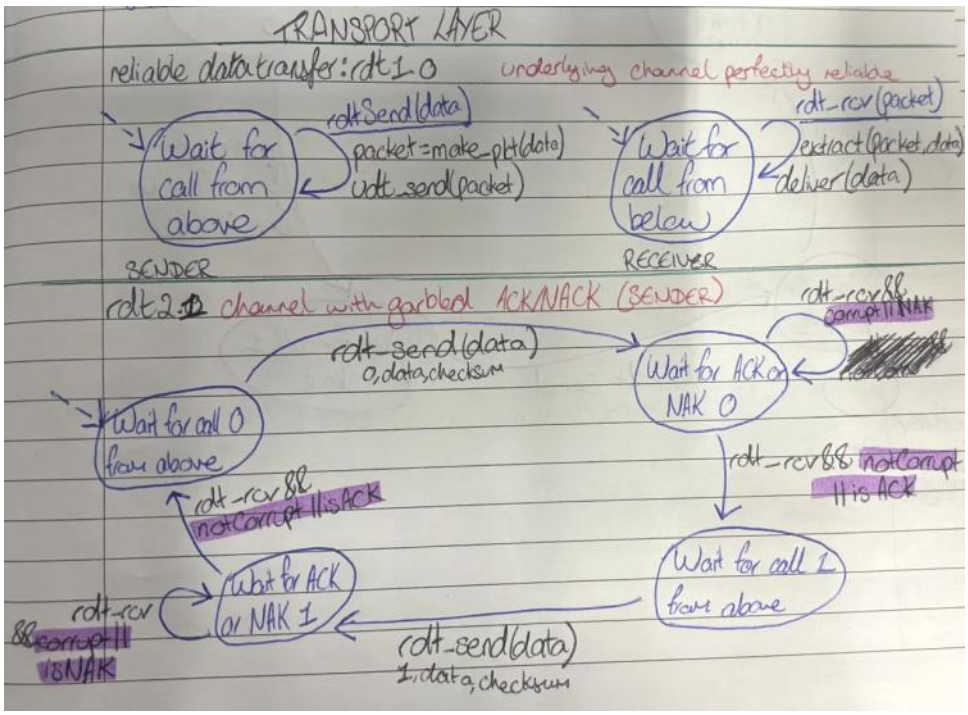


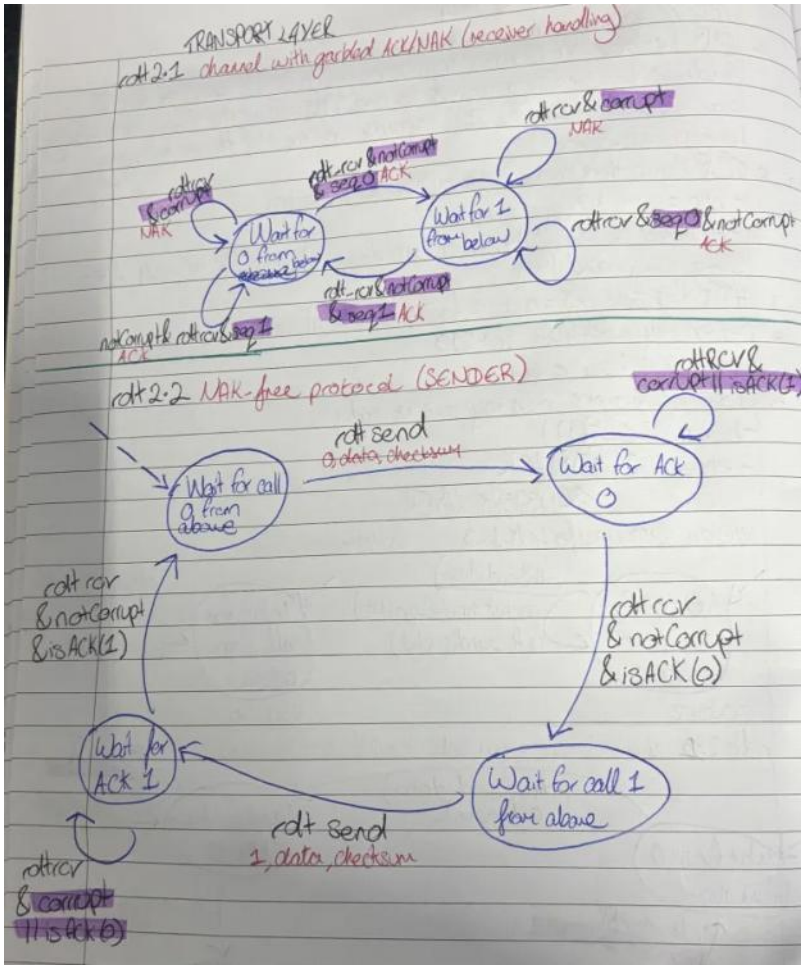
rdt2.2: Receiver FSM

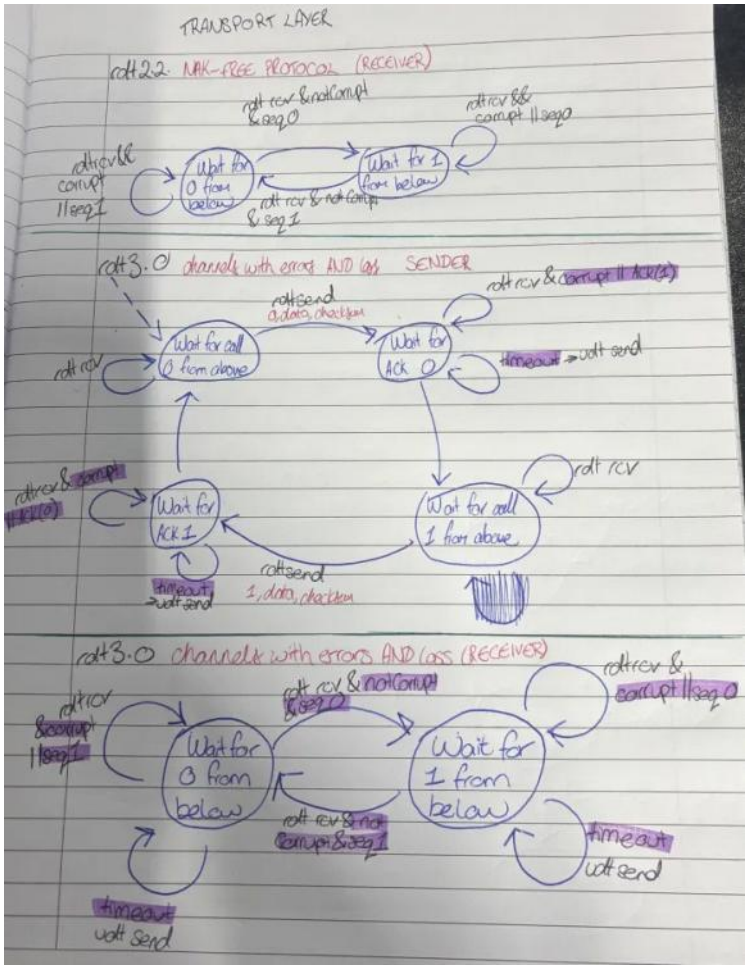


RDT Diagrams

Wednesday, April 02, 2025 1:37 PM





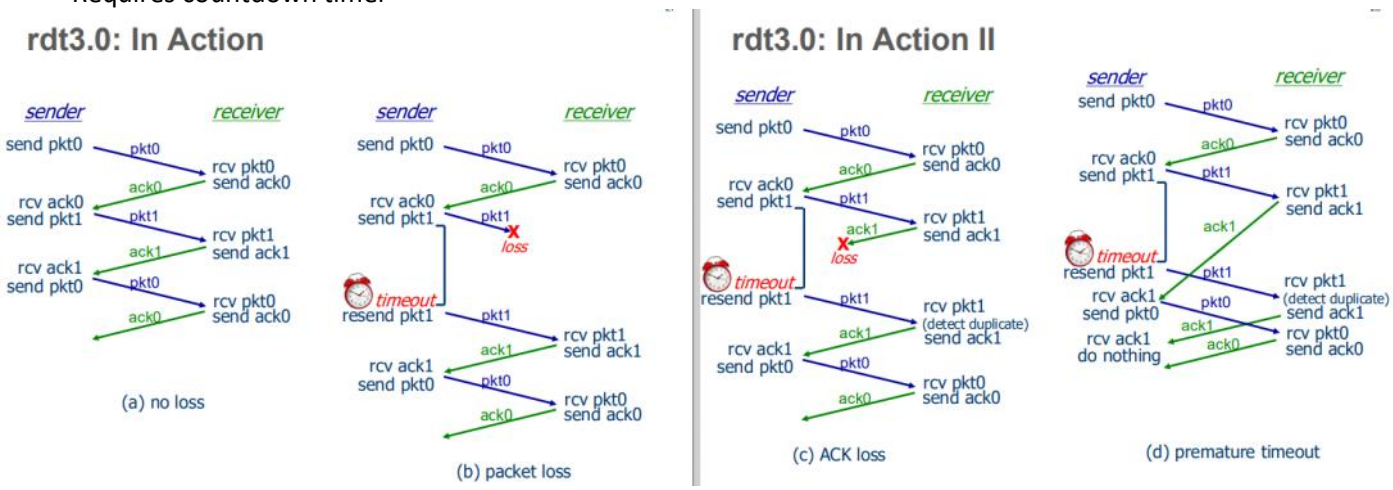

TRANSPORT LAYER
rdt2.2: NAK-FREE PROTOCOL (RECEIVER)
rdt rcv & notCorrupt & seq0
rdt rcv && corrupt || seq0
rdt rcv && corrupt || seq1
Wait for 0 from below
Wait for 1 from below
rdt rcv & notCorrupt & seq1
rdt3.0 channels with errors AND loss SENDER
rdt rcv & corrupt || ACK(1)
rdt send data, checksum
Wait for call 0 from above
Wait for ACK 0
rdt rcv
timeout → udt send
rdt rcv
rdt rcv & corrupt || ACK(0)
Wait for ACK 1
Wait for call 1 from above
rdt rcv
timeout → udt send
rdt send 1, data, checksum
rdt3.0 channels with errors AND loss (RECEIVER)
rdt rcv & notCorrupt & seq0
rdt rcv & corrupt || seq0
rdt rcv & corrupt || seq1
Wait for 0 from below
Wait for 1 from below
rdt rcv & notCorrupt & seq1
timeout udt send
timeout udt send

31 Jan 2025 – Transport Layer

Friday, January 31, 2025 1:58 PM

RDT – Channels with Errors and Losses

- The underlying channel can also lose packets (data, ACKs)
- Checksum, sequence numbers and retransmissions are not enough
- Sender waits a "reasonable" amount of time for ACK
- Retransmits if no ACK received in this time
- If packet or ACK just delayed (not lost) the retransmission will result in a duplicate packet but sequence numbers at the receiver side will stop this
- Requires countdown timer



1 Gbps link (R), 15ms prop. delay, 8000 bit packet (L)
Transmission = 8000 bits / 10^9 bits per second = 8 microseconds
Round trip time = 30ms
.008ms/30.008ms = 0.00027
Effective thruput of 267kbps over a 1Gbps link

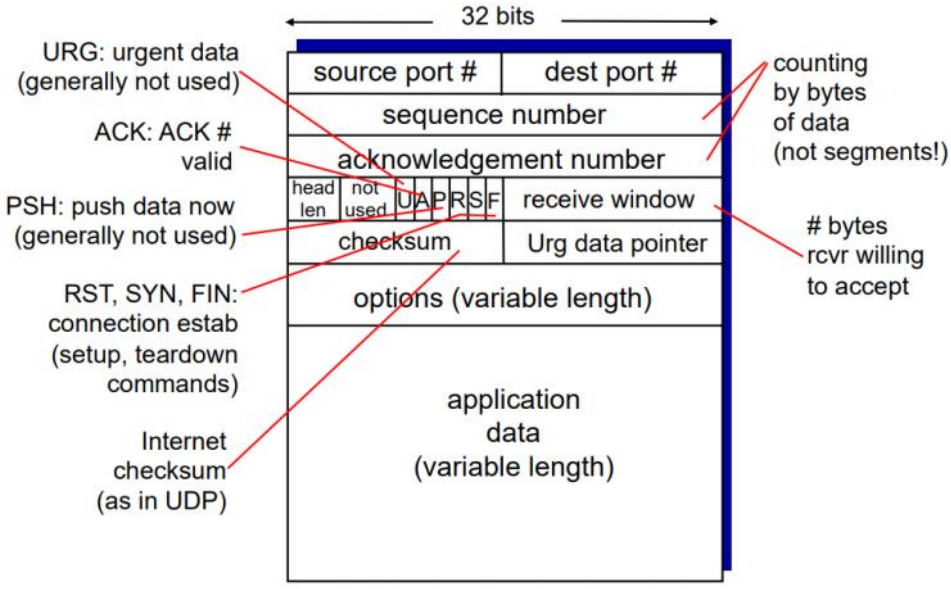
3-packet pipelining increases utilization by a factor of 3
(3L/R) / (RTT +L/R)
(24,000/10^9) / (30+0.008) = .0024/30.008 = 0.00081

Selective Repeat
Receiver individually acknowledges all correctly received packets
Buffers packets as needed for eventual in-order delivery to upper layer
Sender only resends packets for which ACK not received, timer for each unacked packet

TCP

- Only endpoints need to run TCP – one sender, one receiver
- Intermediate hops do not need TCP, can just run IP
- In terms of reliable, in-order byte stream, no "message boundaries", no sequence numbers in the header
- Pipelined TCP congestion and flow control set window size
- Full duplex data (both sides can transmit to each other simultaneously)
- Bi-directional data flow in same connection
- MSS: maximum segment size, maximum data that can be sent in one TCP packet
- MTU minus 40 bytes (for TCP and IP header)
- Connection-oriented, handshaking exchange of control messages
- Initialize sender, receiver and state before data exchange
- Flow controlled – sender will not overwhelm receiver

TCP Header


32 bits
source port #
dest port #
sequence number
acknowledgement number
head len
not used
U
A
P
R
S
F
receive window
checksum
Urg data pointer
options (variable length)
application data (variable length)
URG: urgent data (generally not used)
ACK: ACK # valid
PSH: push data now (generally not used)
RST, SYN, FIN: connection estab (setup, teardown commands)
Internet checksum (as in UDP)
counting by bytes of data (not segments!)
bytes rcvr willing to accept

7 Feb 2025 – Transport Layer

Friday, February 07, 2025 2:01 PM

TCP - Three-way handshake in details

SYN Flood Attack (Denial of Service)
Attacker sends large number of TCP SYN Segments without completing the third handshake step
SYN Cookies
Server doesn't know if SYN segments is coming from a legitimate user
Server creates initial sequence number (ISN/cookie) from the hash of:
Source IP and port, Destination IP and port, secret seed
Server sends SYNACK – maintains no state info corresponding to the SYN
A legitimate client will return an ACK segment using cookie information (ISN+1) in the ACK

3.6 Principles of Congestion Control

Principles of Congestion Control
Slightly different from flow control – not about the endpoints, congestion on the network itself
Too many sources sending too much data too fast for the *network* to handle

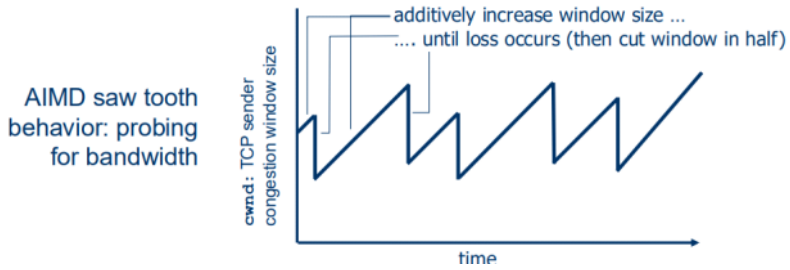
Manifestations - lost packets from buffer overflow at routers, not enough buffer space so packets are dropped.
Also packets dropped because of long delays due to queues in router buffers

End-to-End Congestion no explicit feedback from the network. Congestion *inferred* from end-system observed loss or delay.
Approach taken by TCP.
Network Assisted Congestion Control routers provide feedback to end systems with single-bit in the header indicating congestion/packet has been dropped along the way
Explicit rate for sender to send at

Additive Increase Multiplicative Decrease (AIMD)
Sender increases transmission rate (window size) for useable bandwidth until a loss occurs – additive increase.
Increasing the congestion window (cwnd) by 1 (maximum segment size) MSS every (every round trip time) RTT until a loss is detected

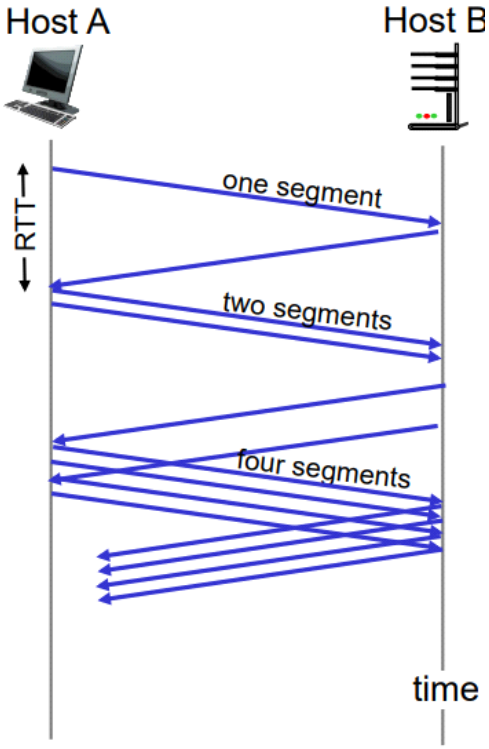
The **maximum segment size** (**MSS**) is a parameter of the *Options* field of the TCP header that specifies the largest amount of data, specified in bytes, that a computer or communications device can receive in a single TCP segment. It does not count the TCP header

Multiplicative Decrease when a loss occurs, cut the cwnd in half



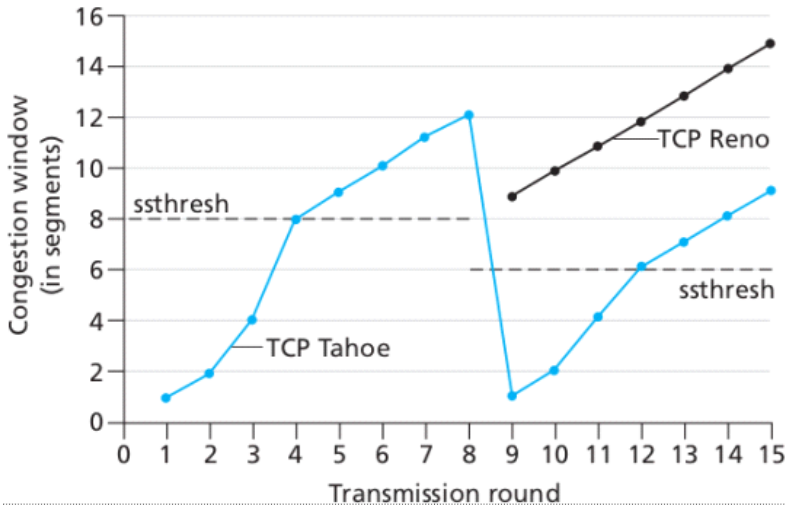
TCP congestion control details
TCP sending rate:
Send cwnd bytes, wait RTT for ACKs, then send more bytes
Sender limits transmission:
LastByteSent – LastByteACKed /< cwnd

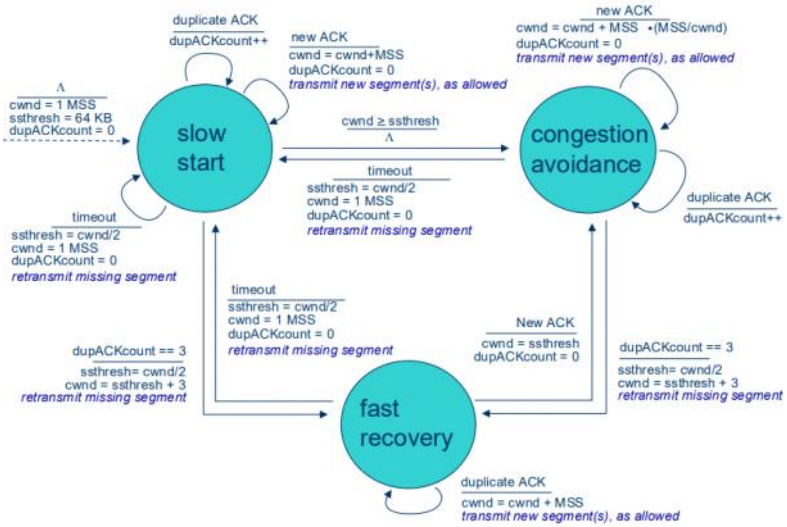
When connection begins, increase rate exponentially until first loss
Initially cwnd = 1MSS
Double cwnd every RTT
Done by incrementing cwnd for every ACK received
Initial rate is low but ramps up exponentially fast



Congestion Avoidance
Loss is indicated by timeout:
Cwnd set to 1MSS and *slow start threshold* = cwnd/2
Window grows exponentially (as in slow start) to the threshold (*ssthresh*)
Once the ssthresh is reached, the cwnd only grows linearly instead of exponentially to slow down

Loss is indicated by triple ACK in newer TCP (TCP Reno)
Duplicate ACKs indicate network capable of delivering some segments
Cwnd is cut in half + 3MSS, then grows linearly
TCP Tahoe always sets cwnd to 1 – timeout or triple ACKs





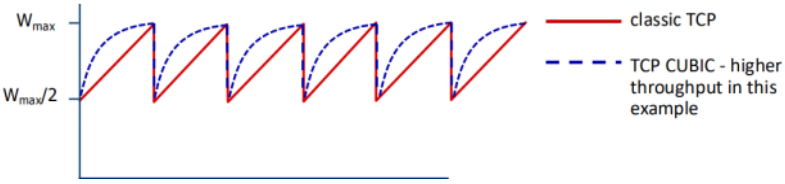
TCP Cubic

Better way than AIMD to probe for useable bandwidth?

W_{max}: sending rate at which congestion loss was detected

Congestion state of the bottleneck link probably hasn't changed much

After cutting the rate window in half on a loss, initially ramp back up to W_{max} faster but then approach W_{max} more slowly as the network likely hasn't changed much



K: point in time when TCP window size will reach W_{max}, K itself is tunable

Increase W as a function of the cube of the distance between current time and K.

Larger increases when further away from L

Smaller, cautious increases when nearer K

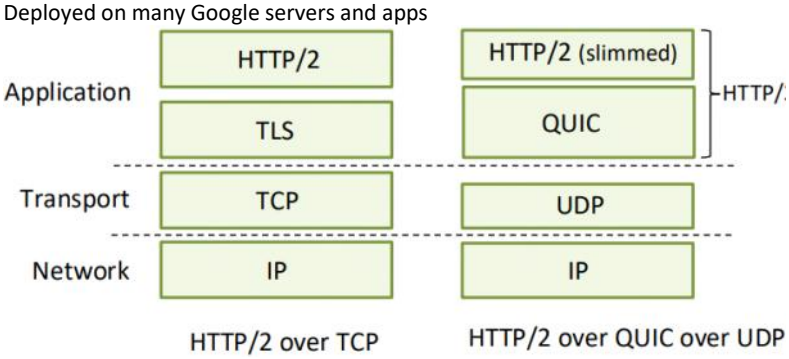
Evolving Transport Layer Functionality

Different flavors of TCP developed for different scenarios

Scenario	Challenges
Long, fat pipes (large data transfers)	Many packets "in flight"; loss shuts down pipeline
Wireless networks	Loss due to noisy wireless links, mobility; TCP treat this as congestion loss
Long-delay links	Extremely long RTTs
Data center networks	Latency sensitive
Background traffic flows	Low priority, "background" TCP flows

Moving of transport layer functions to application layer, on top of UDP – HTTP3/QUIC

QUIC – Quick UDP Internet Connections
Application-layer protocol on top of UDP
Increase performance of UDP
Deployed on many Google servers and apps



Connection establishment, reliability, congestion control, authentication, encryption, state established in one RTT
Multiple application level streams multiplexed over a single QUIC connection – separate reliable state transfer, security, common congestion control

7 Feb 2025 – Application Layer

Friday, February 07, 2025 3:00 PM

Connecting a Network App:
Write programs that run on different end systems, communicate over network e.g. webserver software communicates with browser software
No need to write software for network-core devices – network core devices do not run user applications. Applications on end systems allows for rapid app development, propagation
Possible structure of applications:
Client-server, Peer-to-Peer (P2P)

Client Server Architecture
Server: host is always on, permanent IP address, data centers for scaling
Client: communicate with server, may be intermittently connected, not always on, can change IP address, they do not communicate directly with each other

P2P Architecture
No always-on server
Can be a client, a server or both depending on what you do e.g. torrenting network, sometimes receiving data sometimes serving data out
Arbitrary end-systems directly communicate
Peers request service from other peers, provide service in return to other peers
No centralization
Self-scalability – new peers bring new service capacity as well as new service demands
Peers are intermittently connected and can change IP address – complex management

Processes
Programs running within a host
Client process: process that initiates communication
Server process: processes that wait to be contacted
Within the same host, two processes communicate using inter-process communication defined by OS
Processes in different hosts communicate by exchanging messages
Applications with P2P architectures have client processes and server processes

What Transport Services Does an App Need?
Data Loss: some apps (file transfer, web transactions) require 100% reliable data transfer, other apps like audio can tolerate some loss
Timing: some apps (Internet telephony, interactive games) require low delay to be effective
Other apps make use of whatever throughput they get
Security: encryption, authentication, data integrity

10 Feb 2025 – Application Layer

Monday, February 10, 2025 8:58 AM

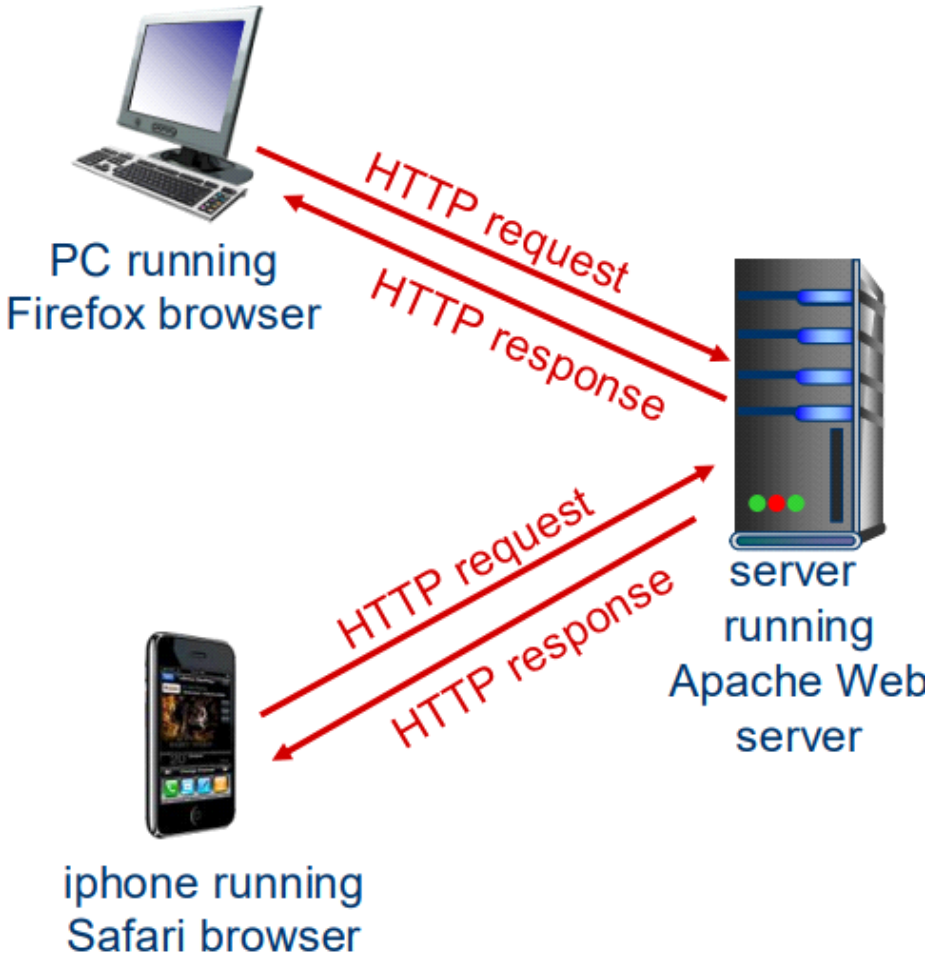
WWW & HTTP

A web page consists of objects. An object can be HTML file, JPEG image, Java applet, audio file, etc.
A web page consists of base HTML file which includes several referenced objects. Each object is addressable by a Uniform Resource Locator (URL)

Use DNS to convert that string into an IP address

HTTP

HTTP uses TCP.
Client initiates TCP connection (creates socket) to server over port 80
Server accepts TCP connection from client
Blocking server only handles one connection and everyone else is queued up
HTTP messages exchanged between Web Browser (HTTP client) and web server (HTTP server)
TCP connection then closed
HTTP is "stateless" - server maintains no information about past client requests. Keeps server as efficient as possible to deal with the amount of requests

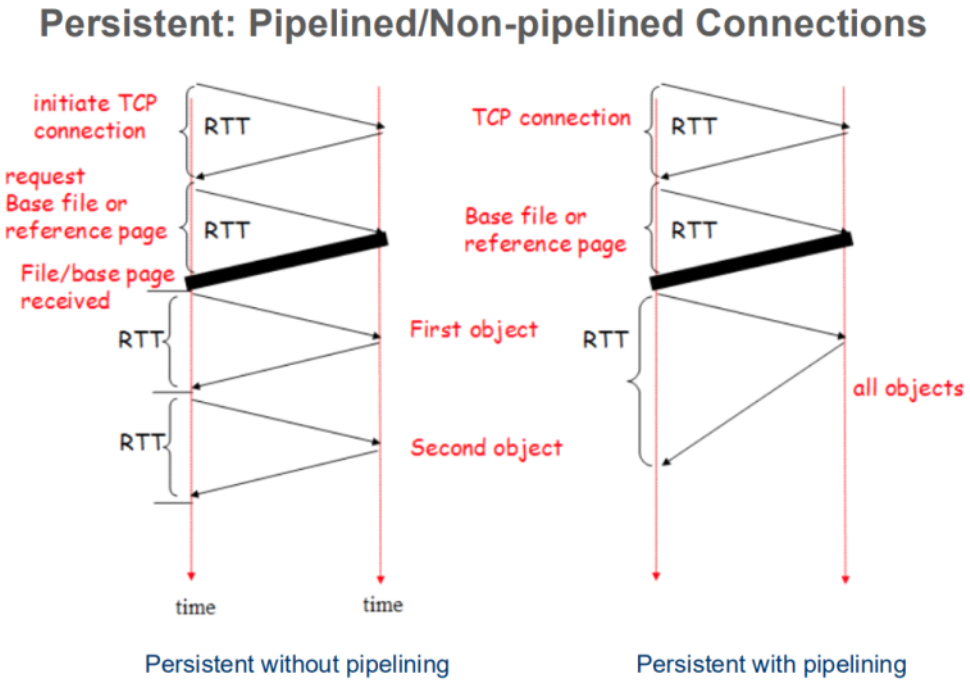


HTTP Connections

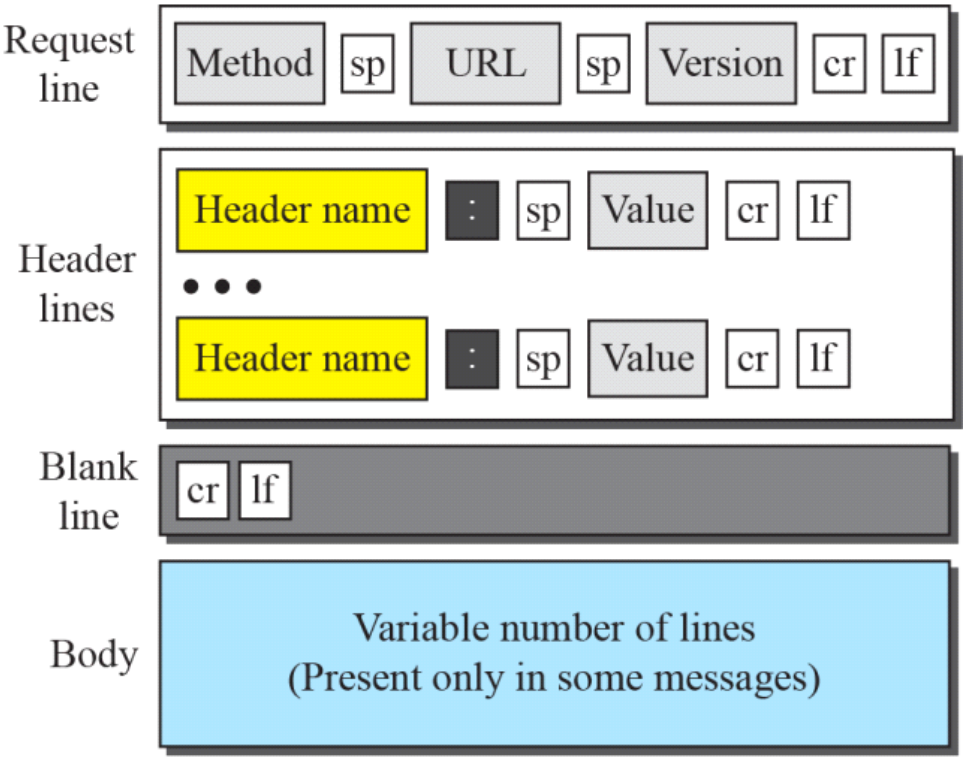
Non-persistent HTTP: at most one object sent over TCP connection, connection then closed.
Downloading multiple objects requires multiple connections.

Persistent HTTP: multiple objects can be sent over single TCP connection between client and server. Advantage is a more efficient transfer between client and server. Wait on TCP connection to stay alive
Default mode of HTTP: persistent connections with pipelining

Non-Persistent HTTP
Response time 2RTT + file transmission time
Persistent HTTP
Non-persistent requires 2RTTs per object.
OS overhead for each TCP connection.
Browsers often open parallel TCP connections to fetch referenced objects
Persistent HTTP server leaves connection open after sending response.
Subsequent HTTP messages between same client/server sent over open connection
Client sends requests as soon as it encounters a referenced object
As little as one RTT for all referenced objects



HTTP Request Message
Two types of HTTP messages: request, response
HTTP request message: ASCII human-readable format
HTTP Request Message Format



Request message

HTTP Method Types

GET, POST, HEAD

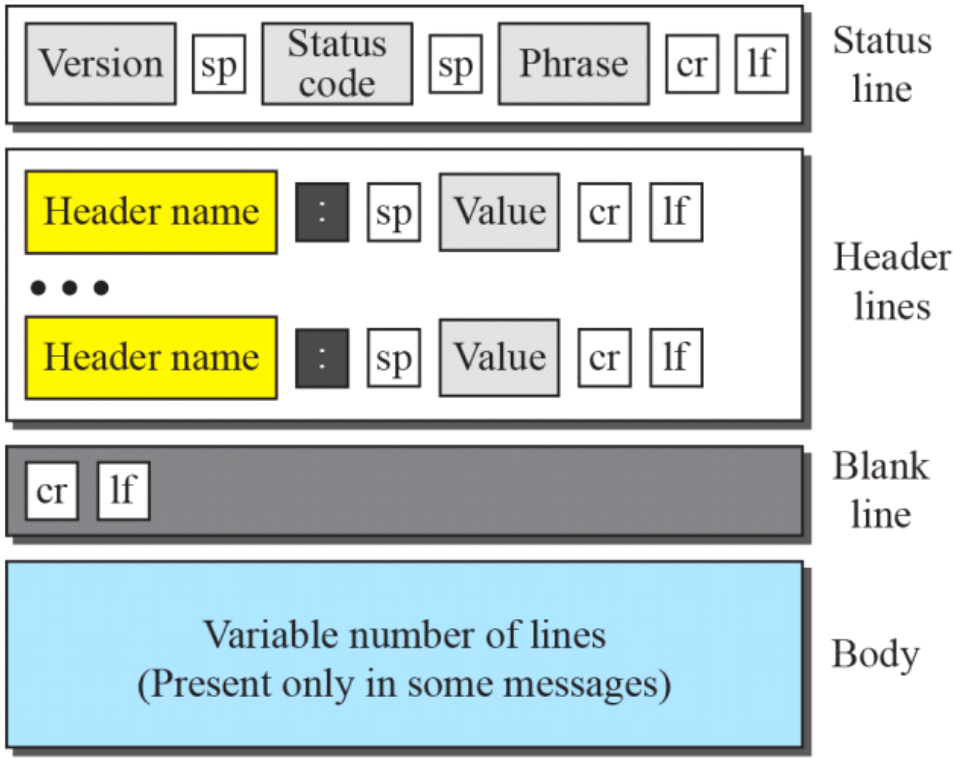
PUT – uploads file in entity body to path specified in URL field in conjunction with Web publishing tools

DELETE – deletes files specified in URL field

POST – input uploaded to server in entity body

URL method – uses GET method, input is uploaded in URL field of request line

HTTP Response Message Format



Response message

HTTP Response Codes

200 - OK – request succeeded
301 - Moved permanently, requested object moved to new location specified in this message
400 – bad request, request message not understood by server
404 – not found, requested document not found on this server
505 – http version not supported

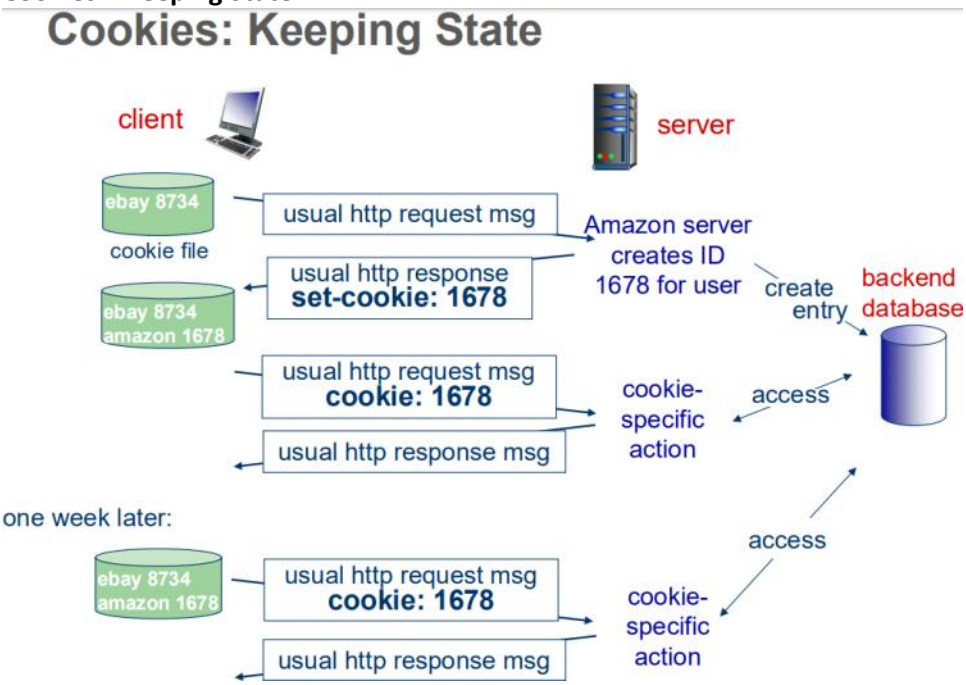
14 Feb 2025 – Application Layer

Friday, February 14, 2025 2:02 PM

Cookies
Cookies can be used for authorisation (automatically log you in), shopping carts (you can come back and your cart will still be there), recommendations, user session state (Web Mail)
Cookies permit sites to learn a lot about you

How to keep "state" - HTTP has been designed as a stateless protocol, maintain state at sender/receiver over multiple transactions – cookies

Cookies – Keeping State



Web Caches (proxy server)
Goal: satisfy client request without involving the origin server
User sets browser to accesses the web via proxy
Browser sends all HTTP requests to proxy server – object in cache, proxy returns object
Else proxy requests object from origin server then caches it and returns object to client

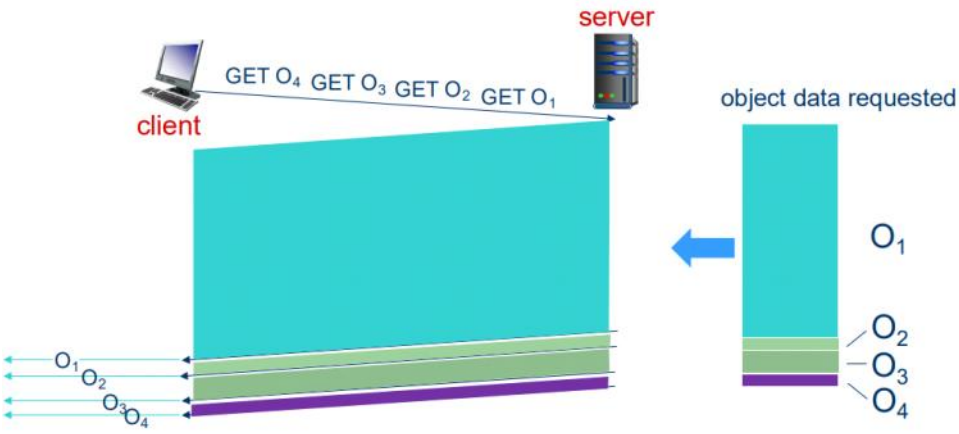
Cache acts as both client and server – server for original requesting client and client to the origin server
Cache installed by ISP
Web caching reduces response time for client request
Reduce traffic on an institution's access link

HTTP/2
Decreased delay in multi-object HTTPS requests
Serer responds in-order, First-come first-served scheduling to GET requests
Small object may have to wait for transmission, head of line blocking behind large objects
Loss recovery stalls object transmission

33

HTTP/2: Mitigating HOL Blocking

- HTTP 1.1: client requests 1 large object (e.g., video file, and 3 smaller objects)

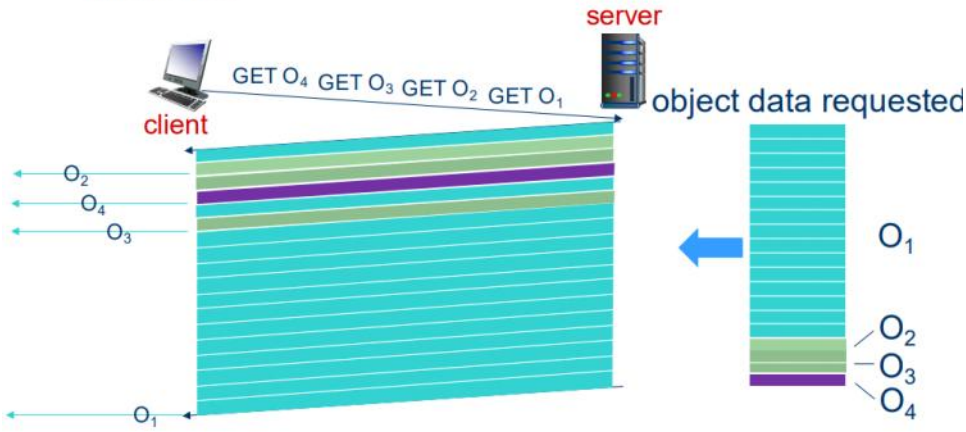


Objects delivered in order requested: O_2, O_3, O_4 wait behind O_1

34

HTTP/2: Mitigating HOL Blocking (cont.)

- HTTP/2: objects divided into frames, frame transmission interleaved



O_2, O_3, O_4 delivered quickly, O_1 slightly delayed

21 Feb 2025 – Application Layer

Friday, February 21, 2025 2:02 PM

Two key attributes – performance and rich GPU
Rich Internet Applications approximate the look, feel and usability of desktop applications
RIA performance comes from Ajaz which uses client-side scripting to make web applications more responsive

Synchronous Web Communication
Synchronous: user must wait while new pages load. Typical communication pattern used in web pages (click, wait, refresh)
While a synchronous request is being processed on the server, the user cannot interact with the client web browser

Web Application: a dynamic web site that mimics the feel of a desktop app presents a continuous use experience rather than disjoint pages e.g. Twitter feed

Ajax: Asynchronous XML and JavaScript – Not a programming language; a particular way of using JavaScript. It downloads data from a server in the background. Allows dynamically updating a page without making the user wait. Avoids the "click-wait-refresh" pattern

Typical Ajax Request
1. user clicks, invoking an event handler
Handler's code creates an XML HTTP Request (XHR) object
2. XHR Object requests page from server
3. Server retrieves appropriate data and sends it back
4. XHR object fires an even when data arrives
   Often called a callback. You can attach a handler function to this event

Callback even handler processes the data and displays it

XHR Object
Resides on the client
Layer between the client and the server that manages asynchronous requests in Ajax applications
To initiate an asynchronous request create an instance of the XHR object
Use its open method to set up the request and its send method to initiate the request
For security purposes the XHR object does not allow a web application to request resource rom domains other than the one that served the application – Known as the Same Origin Policy (SOP)

The WebSocket Protocol
Ajax always has to poll the server for data rather than receive it via a push from the server.
If you're moving high volumes of data – overhead of creating an HTTP connection every time is going to be a bottleneck.
WebSockets allow your client-side JavaScript to open and persist a connection to a server.
With WebSockets, data is exchanged as messages, which can happen very quickly due to the persistent connection.
Another powerful aspect of WebSockets is a capability called full-duplex. Contrast that with Ajax where the server has no method for pushing messages to the client

How WebSockets work
Defines an API establishing a two-way "socket" connection between a web browser and server.
Persistent connection between the client and server.
Both parties can start sending data at any time.
The client established a WebSocket connection through a process known as the WebSocket handshake. Process starts with the client sending a regular HTTP request to the server.

Upgrade header is included in this request. Informs the server that the client wishes to establish a WebSocket connection

WebSocket Efficiency
With websockets you can transfer as much data as you like in both directions simultaneously.
Without incurring the overhead associated with traditional HTTP requests.
Data is transferred though a WebSocket as messages. Each of which consists of one or more frames containing the data you are sending with 2-byte framing overhead only.
Using this frame-based messaging system helps to reduce the amount of non-payload data that is transferred – leading to significant reductions in latency.

21 Feb 2025 – Network Security

Friday, February 21, 2025 2:32 PM

What is network security?
Confidentiality – Only sender and intended receiver should "understand" the message contents.
Sender encrypts msg and receiver decrypts msg
Authentication – sender and receiver want to confirm the identity of each other
Message integrity – sender and receiver want to ensure a message is not altered in transit or afterwords without detection.

Bob and Alice want to communicate securely. Carol wants to intercept, delete and add

What can the bad guys do
Passive attack – eavesdrop or intercept messages
Active attack – actively insert messages into connection
Impersonation – fake source address in packet
Hijacking – take over ongoing connection by removing sender or receiver, inserting himself in place

Cryptography
Original data to be transferred is called plaintext or cleartext
After encryption it is called ciphertext
Plaintext denoted as *P*, ciphertext denoted as *C*
Encryption function *E* operates on *P* to produce *C*
E(P) = C
Decryption function *D* operates on *C* to produce *P*
D(C) = P

Following identity function must also hold true for the cryptosystem to function correctly:
D(E(P)) = P

Cryptographic Key
All modern encryption algorithms use a key denoted by *K*
The key can take on many possible values range of possibilities is called the key space
The encryption and decryption functions now become
Ek(P) = C and Dk(C)=P

Substitution Ciphers
Each letter or group of letter is replaced by another letter or group of letters to disguise it – Caeser Cipher
Vigenere Cipher 1

Vigenere Cipher

24 Feb 2025 – Network Security

Monday, February 24, 2025 9:02 AM

Symmetric Key Encryption
Based on the sender and receiver of a message knowing and using:
-the same secret key.
-the same cryptosystem.
Sender uses secret key to encrypt the message.
Receiver uses the same secret key to decrypt the message.
Can be multiple (N) people but everyone has to get the same key
Main problem: getting the sender and receiver to agree on a secret key without anyone else finding out especially if there is no cryptosystem in place to begin with.
Examples of symmetric key algorithms: DES, Triple DES, IDEA, AES

Data Encryption Standard (DES)
Standard encryption method adopted by the US gov.
Origins in an internal IBM project Lucifer.
Algorithm is complex but easily implemented in hardware.
Software implementations are widely available.
DES is a block cipher, operates on a single chunk of data at a time, 64 bits/8 bytes and produces a 64 bit output (can be adapted to become a stream cipher).
DES key length is 56 bits – often express as an 8-character string with the extra bits used as a parity check
DES algorithm has 19 distinct stages.
First stage re-orders the bits of the 64-bit input block by applying a fixed permutation (P-box).

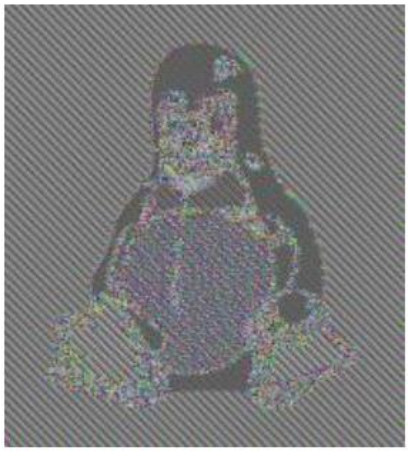
Last stage is the exact inverse of this permutation
Stage penultimate to the last one exchanges the leftmost 32 bits with the rightmost 32 bits.
Remaining 16 stages are called rounds – functionally identical but take as an input a quantity computed. from the key and the round number.

Cracking DES
56 bits is a short key.
Brute force: try every key – 2^{56} encryptions to try all keys
Special chips can check 4 million keys/sec.
$1 million DES cracking machine could break it in a few hours. Tested 245 billion keys per second.
Improvement: Triple DES – makes use of 2 or 3 keys (112 or 168 bits). Uses EDE or DED mode

Modes of Operation for Block Ciphers
DES is known as Electronic Code Book (ECB mode).
Each 64-bit block is encoded independently of all other blocks. No dependency on the blocks. Blocks could be removed or moved around.
Block ciphers operating in ECB mode can be parallelized – high-speed implementations

ECB Problems
ECB mode encrypts in a highly deterministic manner.
All blocks encrypted in exactly the same way. Using the same input and the same key.
Identical plaintext blocks result in identical ciphertext – the same encryption as long as the key does not change.

Original Image

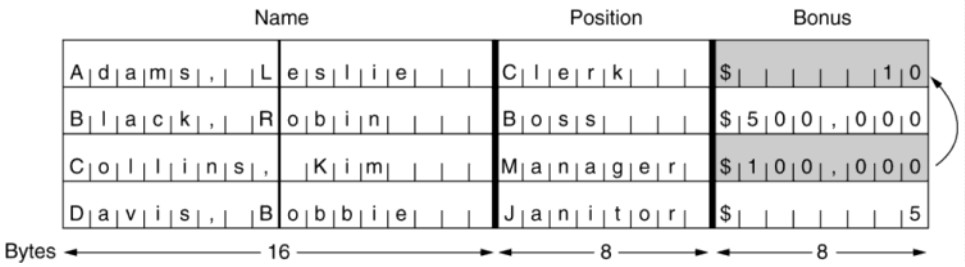


Encrypted with ECB Mode

ECB Substitution Attacks

Each block is encrypted independently of all the other blocks. There is no dependency on the other blocks.

Allows for a passive intruder to manipulate the ciphertext blocks without the receiver noticing



10 Mar 2025 – Network Security

Monday, March 10, 2025 8:53 AM

Public Key Cryptography
When Alice wants to send an encrypted message to Bob, she looks up his public key in a public directory.

RSA
Standard algorithm for implementing *asymmetric*- key cryptography
Security is based on the difficulty of factoring very large numbers. One way trapdoor function. E.g. prime factoring.
Easy to calculate product of 2 large prime numbers. Difficult to calculate the prime factors from the product

Modular Arithmetic
Most number sets we are used to are infinite e.g. set of real numbers
Most cryptographic algorithms are based on arithmetic with a finite set of numbers.

Euler's Phi Function
Important to know if there is a solution to the equation: a.x = b(mod m)
If gcd(a,m) = 1 exactly one solution
Number of positive integers less than m and relatively prime to m – relatively prime = gcd(a,m)=1
Phi(10) = 4. 1,3,7,9 are relatively prime to 10
Phi(21) = 12. 1,2,4,5,8,10,11,13,16,17,19,20 are relatively prime to 21

In general, let m have the following canonical factorization

$$m = {p_1}^{e_1} \cdot {p_2}^{e_2} \cdot \ldots . {p_n}^{e_n}$$

Exercise: Calculate $\phi(240)$?
M = 240 = 16*15
$2^4 * 3 * 5$
64 integers are co-prime to 240

14 Mar 2025 – Network Security

Friday, March 14, 2025 1:12 PM

Multiplicative Inverse

The multiplicative inverse (a-1) of a modulo m is an integer x such that a*x is congruent to 1 mod m

Extended Euclidean Algorithm (EEA)

- Division of a by b modulo m is a product of a and b^{-1} modulo m
  - bla is equivalent to $a \cdot b^{-1} \pmod{m}$
- Q: What is 4^{-1} modulo 11?
  - $x \equiv 4^{-1} \pmod{11}$

- $\gcd(11, 4) = 1$
  - $11 = 2 \cdot 4 + 3$
  - $4 = 1 \cdot 3 + 1$
  - $3 = 3 \cdot 1 + 0$
- Back substitution

- The modular multiplicative inverse of an integer modulo m can be found with the Extended Euclidean Algorithm
  - $s \cdot r_0 + t \cdot r_1 = \gcd(r_0, r_1)$
  - $s \cdot r_0 + t \cdot r_1 = 1$
  - $s \cdot 0 + t \cdot r_1 \equiv 1 \pmod{r_0}$
  - $t \equiv r_1^{-1} \pmod{r_0}$

Exercise: Compute 15^{-1} mod 26?

24 Mar 2025 – Network Security

Monday, March 24, 2025 9:02 AM

Groups & Fields
A group is set with one operation and the corresponding inverse operation. If the operation is addition, the inverse operation is subtraction.
A finite field is a set with a finite number of elements in which we can add, subtract, multiply, invert

A group is a set if elements G together with an operation which combines two elements of G and has the following:

- The group operation is closed
- The group operation is associative
- There is an element e in G called the identity element
- For all elements in G, there exists an inverse of the element

In order to have all four basic arithmetic operations (i.e., addition, subtraction, multiplication, division) in one structure – Need a set which contains an additive and a multiplicative group, I.e. field

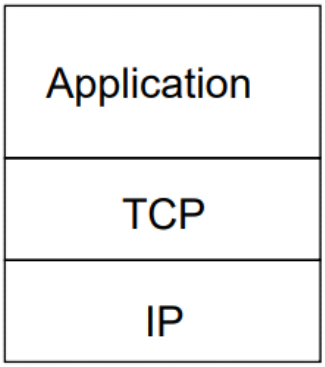
A field F is a set of elements with the following properties
All elements of F form an additive group with the group operation "+" and the identity element 0
All elements of F except 0 form a multiplicative group with the group operation "·" and the identity element 1
When the two group operations are mixed, the distributivity law holds.
The set $\mathbb{R}$ of real numbers is a field with neutral element 0 for the additive group, 1 for multiplicative
Every real number a has an additive inverse –a
Every nonzero element has a multiplicative inverse 1/a

7 Apr 2025 – Network Security

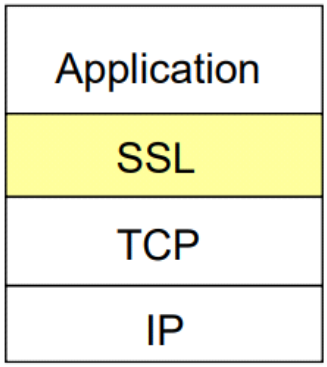
Monday, April 07, 2025 9:02 AM

Secure Sockets Layer

Transport layer security provides an end to end connection between browser and web server using SSL services



normal application



application with SSL

Tunnel between browser and web server that no one can see

Symmetric-Key Cryptography

Saturday, April 12, 2025 8:48 PM

Substitution Ciphers:

Caeser Cipher: mono-alphabetic

Substitution ciphers can be broken by identifying commonly occurring characters e.g. 'e', 'a', 'n'

Or commonly occurring phrases "system", "password", "money"

Vigenére Cipher is poly-alphabetic

Let's say we settled on the key = LEMON and our plaintext message was "Hello world".

We are going to encrypt each letter of the plain text by a shift cipher which replaces the letter A with the appropriate letter of the key.

KEY	L	E	M	O	N	L	E	M	O	N
Shift	11	4	12	14	13	11	4	12	14	13
Plain text	H	E	L	L	O	W	O	R	L	D

KEY	L	E	M	O	N	L	E	M	O	N
Shift	11	4	12	14	13	11	4	12	14	13
Plain text	H	E	L	L	O	W	O	R	L	D

Now we know what shift we are using for encryption for each letter we just go through and do the normal shift cipher encryption on letter at a time

This gives the cipher text " Sixzb hsdzq"

Transposition Ciphers

M	E	G	A	B	U	C	K
7	4	5	1	2	8	3	6
p	l	e	a	s	e	t	r
a	n	s	f	e	r	o	n
e	m	i	l	l	i	o	n
d	o	l	l	a	r	s	t
o	m	y	s	w	i	s	s
b	a	n	k	a	c	c	o
u	n	t	s	i	x	t	w
o	t	w	o	a	b	c	d

Plaintext

pleasetransferonemilliondollarsto
myswissbankaccountsixtwotwo

Ciphertext

AFLLSKSOSELAWAIATOOSSCTCLNMOMANT
ESILYNTWRNNTSOWDPAEDOBUOERIRICXB

Symmetric Key Encryption

The sender and receiver know the same secret key and cryptographic system.

Issue with getting the two parties to have the same secret key in the first place without anyone else knowing

DES (Symmetric)

A block cipher that operates on 64 bits (8 bytes) at a time.
Produces a 64-bit output
Key length is 56 bits. This is a short key. Brute force attacks is 2^{56} encryptions to try all keys.
Triple DES makes use of 3 keys (168 bits)

ECB
Each 64-bit block is encrypted independently of each other. Individual blocks can be changed/moved/deleted
Identical plaintext blocks result in identical ciphertext blocks if the key doesn't change
Plaintext Block 1 ⟶ Encrypt ⟶ Ciphertext Block 1

Plaintext Block 2 ⟶ Encrypt ⟶ Ciphertext Block 2

CBC
Each block of plaintext is XORed with the previous block's ciphertext block.
Each ciphertext block depends on all plaintext blocks processed up to that point (blocks are not independent of each other).
Initialization vector is used for the first block.
One error effects the next and next block
IV ⟶ XOR Plaintext 1 ⟶ Encrypt ⟶ Cipher 1 ...
Cipher 1 ⟶ XOR Plaintext 2 ⟶ Encrypt ⟶ Cipher 2 ...

OFB
Outputs are encrypted *before* being XORed with the plaintext.
Ciphertexts are independent of each other, NOT chained.
Bit errors do not propagate.
IV ⟶ Encrypt ⟶ Output1 ⟶ XOR with Plaintext 1 ⟶ Cipher 1
Output1 ⟶ Encrypt ⟶ Output2 ⟶ XOR with Plaintext 2 ⟶ Cipher 2
Output2 ⟶ Encrypt ⟶ Output3 ⟶ XOR with Plaintext 3 ⟶ Cipher 3

CFB
Chained like in CBC. Ciphertext chains are encrypted *before* being XORed with the plaintext
IV ⟶ Encrypt ⟶ XOR with Plaintext 1 ⟶ Cipher 1

Cipher 1 ⟶ Encrypt ⟶ XOR with Plaintext 2 ⟶ Cipher 2

AES

Symmetric cipher with variable key and block sizes of 128,192 and 256 bits

A plaintext block X undergoes n rounds of operations to produce an output block Y

Each operation is based on the value of the nth round key

Round keys are derived from the Cipher Key by first expanding the key then selecting parts of the expanded key for each round

Asymmetric-Key Cryptography

Saturday, April 12, 2025 8:48 PM

Asymmetric-Key Cryptosystems

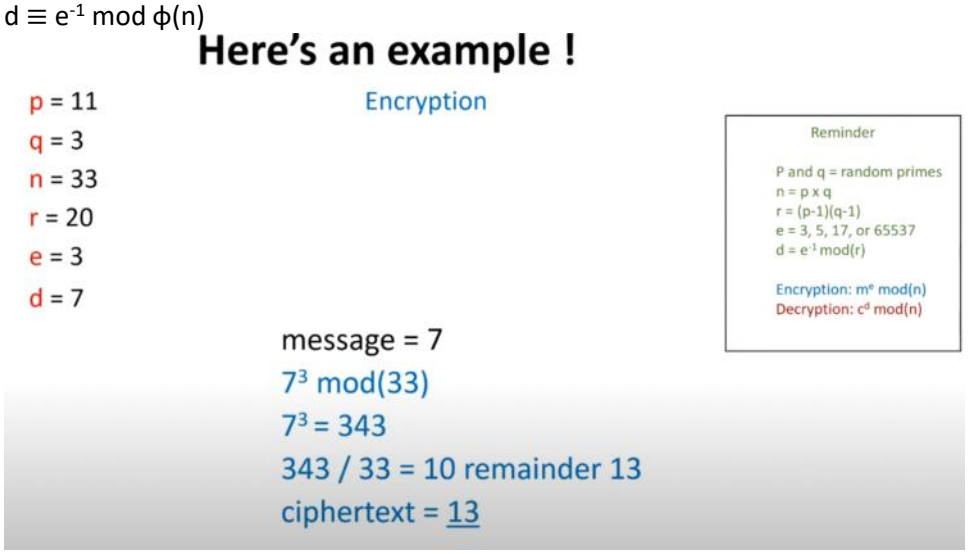
Each person generates a public and a private key

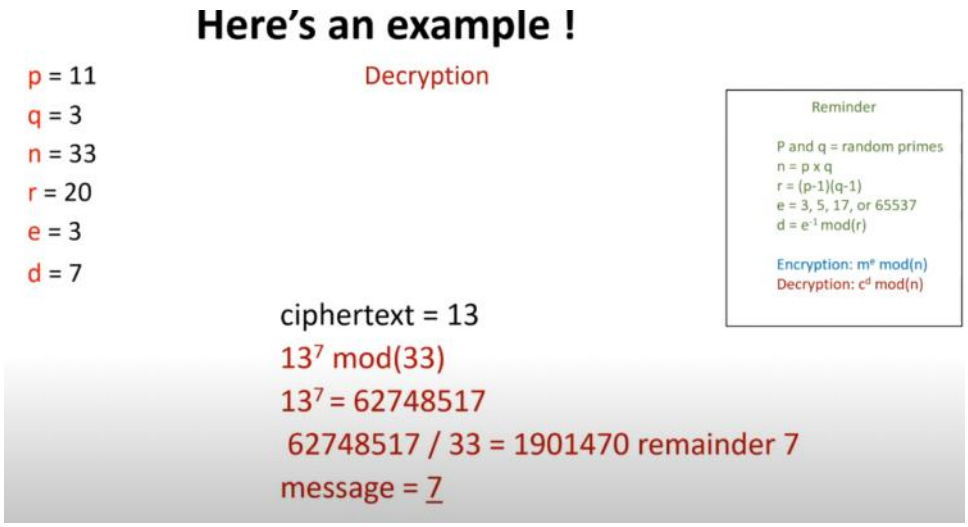
Public key is published and widely distributed

Need for exchanging secret keys is eliminated

RSA

Security is based on the difficulty of factoring very large numbers
Choose two large distinct primes p and q
Compute the product (modulus): n = p*q
Phi(n) = (p-1)*(q-1)
Randomly choose an encryption/public key e, less than n that has no common factors with ϕ(n)
e and ϕ(n) are relatively prime, gcd=1
d is the decryption/private key such that
$e \cdot d \equiv 1 \pmod{\phi(n)}$
$d \equiv e^{-1} \bmod \phi(n)$





Fast Exponentiation

(1a) Compute 240^262 mod 14 using the fast modular exponentiation method.

The probability that a randomly picked integer p is prime is 1/ln(p).
In reality we only test odd numbers so the probability doubles to 2/ln(p)
For RSA with a 1024-bit modulus n, the primes p and q each should have length 512 bits i.e., p, q ≈ 2^{512}

$$\frac{2}{\ln(2^{512})} = \frac{2}{512\ln(2)} \approx \frac{1}{177}$$

A crypto scheme is malleable if an attacker is capable of transforming the ciphertext into another ciphertext that can reveal the plaintext.
RSA is deterministic if for a particular key, a particular plaintext is always mapped to a particular ciphertext.
In practice, RSA has to be used with padding.
RSA Side Channel attacks exploit information about the private key which is leaked through physical channels such as power consumption and timing data.

Asymmetric key algorithms are not a substitute for symmetric key algorithms like DES and AES.
We can use a public key algorithm to securely transfer a session key k and use the session key for bulk encryption and decryption.

PrivateKey(PublicKey(Message)) = Message = PublicKey(PrivateKey(Message))

Digital Signatures, X.509 Certs & PKI

Saturday, April 12, 2025 8:48 PM

Digital Signatures
Bob receives a message from Alice and wants to ensure that the message originally came from Alice and has not been altered since being sent by Alice.

Sender digitally signs the document, establishing they are the creator.
Recipient can prove that no one else but the sender must have signed the document.

X.509 Certs
The trusted third party will construct the certificate

Assumes that every user in the system has the public key of the trusted third party
Allows one to verify the digital signature on the certificate.
Guaranteeing the public key is associated with the named user.
TTPs that issue certificates are called certification authorities CAs
The root CA issues certificates to other CAs.
Users in the system only need to hold the public key of the root CA

Authentication Protocols

Saturday, April 12, 2025 8:48 PM

The Discrete Logarithm Problem (DLP)

Given a finite cyclic group $Zp*$ of order p - 1 and a primitive root $\alpha \in Zp*$ and another element $\beta \in Zp$

The Discrete Logarithm Problem (Solved Example)

Computing discrete logarithms modulo a prime is a very hard problem if the parameters are sufficiently large.

In practice it is desirable to have DLP in groups with prime cardinality to prevent the Pohlig-Hellman attack.

To avoid brute-force attacks on DLP based cryptosystems in practice The cardinality $|G|$ of the underlying group must thus be sufficiently large

Diffie-Hellman Key Exchange (DHKE)

A protocol that allows strangers to establish a shared symmetric key without them having to meet

Compute the two public keys and the common key for the Diffie-Hellman Key Exchange (DHKE) scheme with the parameters p = 467 and delta = 2, a = 3 and b = 5. Perform the computation for the common key for Alice and Bob.

Compute Alice's public key:

A = $delta^a$ mod p

A = 2^3mod467 = 8

Compute Bob's public key:

B = $delta^b$ mod p

B = 2^5mod467 = 32

Compute the common key:

Alice computes the common key using Bob's public key.

Bob computes the common key using Alice's public key.

Alice computes the common key as: $K_A = B^a$ mod p

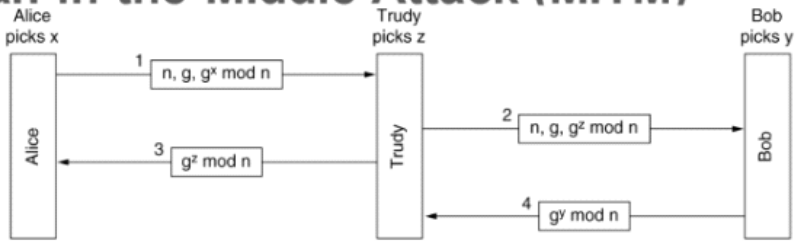
Bob computes the common key as : $K_B = A^b$ mod p

$K_A = K_B$

Common key for Alice: 32^3 mod 467 = 78

Common key for Bob: 8^5 mod 467 = 78

Man-in-the-Middle Attack (MITM)



- Alice computes the key g^{xz} mod n
  - So does Trudy (for messages to Alice)
- Bob computes g^{yz} mod n
  - So does Trudy (for messages for Bob)
- Alice thinks she is talking to Bob and so establishes a session key (with Trudy) and so does Bob
  - Also known as the bucket brigade attack

The domain parameter p should have a length of 1024 bits (308 digits) or longer
α should be a primitive root whose powers modulo p generate all integers from 1 to p –1

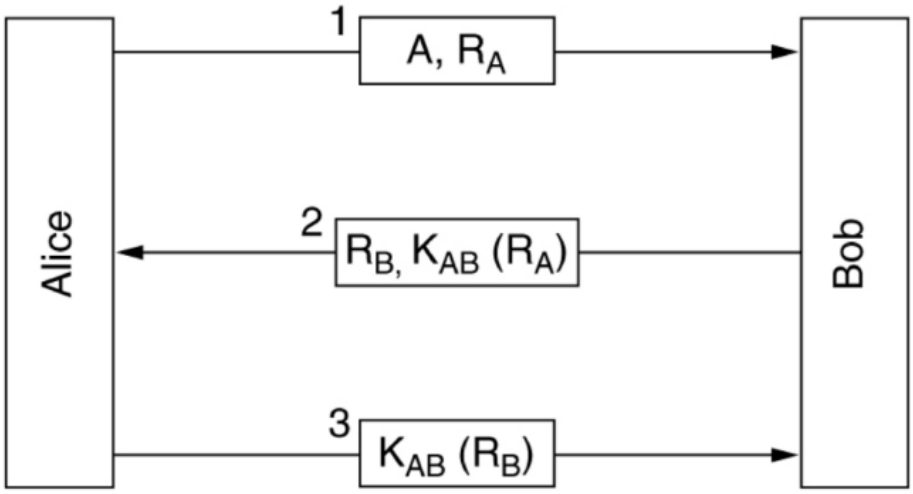
The ElGamal Encryption Scheme
The ElGamal encryption scheme can be viewed as an extension of the DHKE protocol.
If Alice wants to send an encrypted message x to Bob, both parties first perform a DHKE to derive a shared key kM.
The new idea here is that Alice uses kM as a multiplicative mask to encrypt x

Elliptic Curve Cryptography (ECC)
Elliptic Curve Cryptography is based on the generalized discrete logarithm problem of finding the integer x such that:

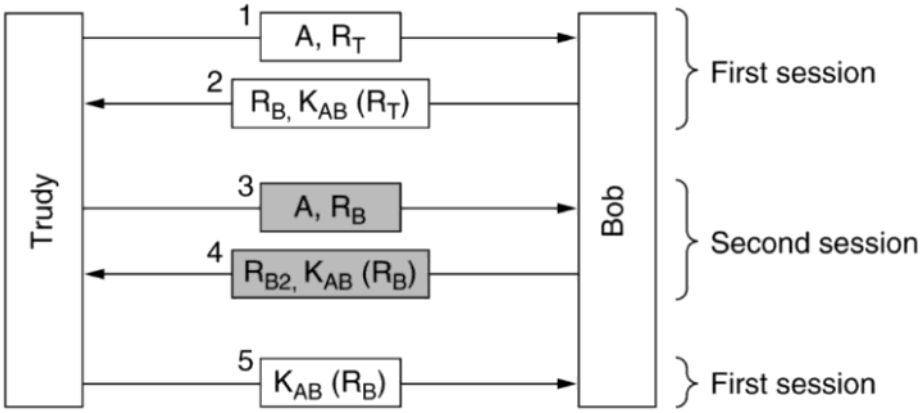
$$\beta = \underbrace{\alpha \circ \alpha \circ \ldots \circ \alpha}_{x \text{ times}} = \alpha^x$$

ECC is more efficient, fewer bits needed for same security and fewer computations
A feature of elliptic curves is that there is a natural way to take two points on an elliptic curve and "add" them to produce a third point.
ECC cryptosystems are based on the idea that d is large, kept secret, and attackers cannot compute it easily.
256-bit ECC key provides the same security as a 3072-bit RSA key. Can be up to 10 times faster depending on the implementation

Challenge-Response Protocol
RB is a random number which is only used once, known as a nonce.

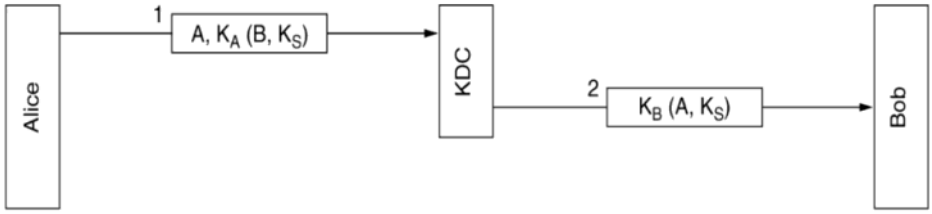


Trudy can easily break this if multiple sessions can be opened with Bob.

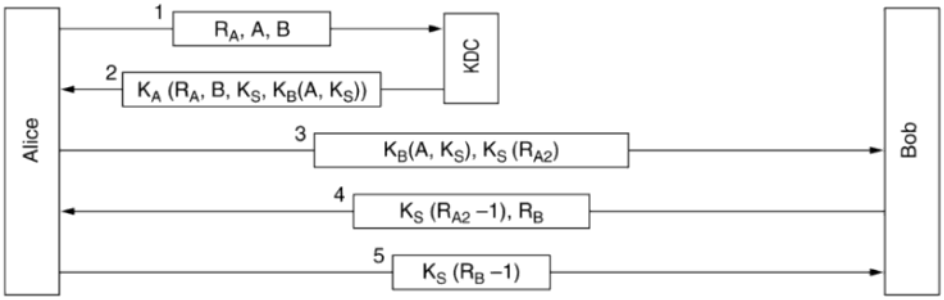


Message Authentication Code from Hash Functions (HMAC)
Calculated using a cryptographic hash function in combination with a secret key.
Like digital signatures they can be used to quickly verify data integrity and authenticity of a msg.

Authentication Using a KDC
To talk to n people using a shared secret would require setting up n*(n-1)/2 keys.
Alternative is to introduce a Key Distribution Center (KDC)
Each user has a single shared key with the KDC – authentication and session key management happen through KDC.



Needham-Schroder Authentication Protocol
Alice tells the KDC that she wants to talk to Bob. Sends a msg containing a random number RA as a nonce.
The KDC sends back msg 2 containing RA, a session key and a ticket that she can send to Bob.
Alice now sends the ticket to Bob along with a new random number RA2 encrypted with the session key KS.
Bob sends back KS(RA2-1) to prove to Alice that she is talking to the real Bob.

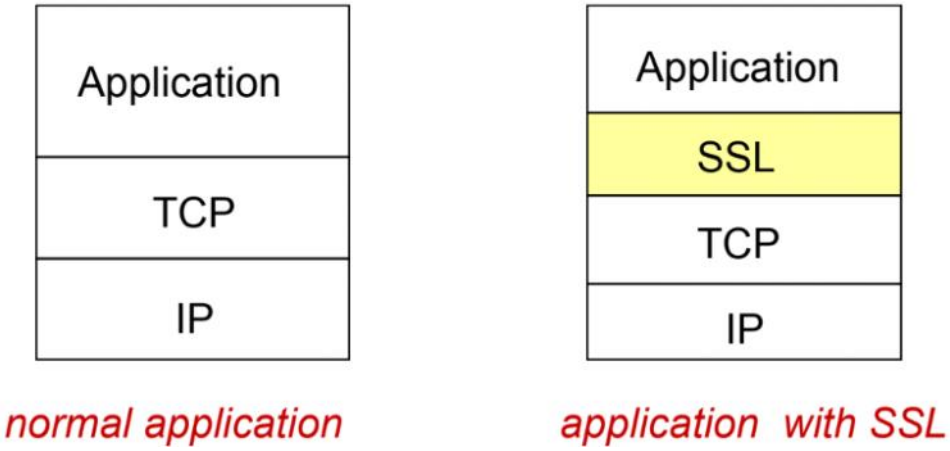

Alice
KDC
Bob
1
R_A, A, B
2
K_A (R_A, B, K_S, K_B(A, K_S))
3
K_B(A, K_S), K_S (R_{A2})
4
K_S (R_{A2} −1), R_B
5
K_S (R_B −1)

Transport Layer Security (TLS)

Saturday, April 12, 2025 8:48 PM

Secure Sockets Layer
Provides transport layer security to any TCP-based application using SSL services
Between Web browsers and servers for E-commerce
In practice Transport Layer Security (TLS) v1.2/v1.3 should be used

Security services:
Server authentication, data encryption, client authentication



Electronic Payments

Saturday, April 12, 2025 8:48 PM

Non-Mathematical Questions

Wednesday, April 30, 2025 10:40 AM

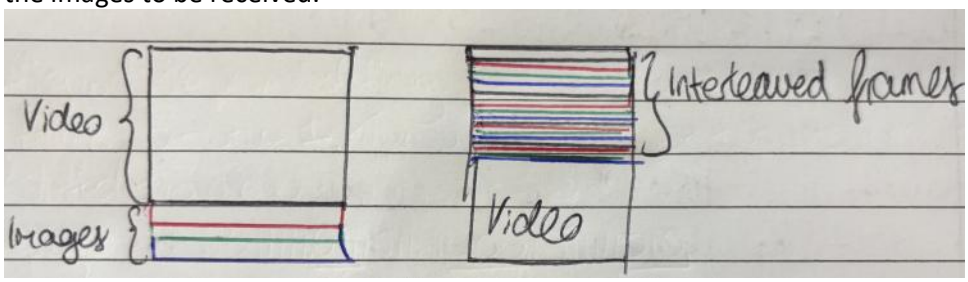
1. (a) Assume you open your browser and enter `http://yourbusiness.com/about.html` in the address bar. What happens until the webpage is displayed? Provide details about the protocol(s) used and a high-level description of the messages exchanged. [10 marks]

DNS Resolution: when the client makes a request to http://yourbusiness.com/about.html. The server first checks if the IP address of the server hosting yourbusiness.com is in cache. If it is in cache, it will return the cached result. If not, it will make a request to the local DNS requesting the IP address of the resource located at yourbusiness.com
TCP Connection: a TCP connection is set up between the client and server. The client sends a SYN packet to initiate the TCP handshake. The server responds with a SYN-ACK. The client must respond to this with an ACK to complete the 3-way handshake.
HTTP Request: a request is made to the resource at /about.html. As this is a HTTP connection it uses port 80.
Rendering: The page is rendered to the screen
Close TCP: unless the "keep-alive" header has been used, the TCP connection is closed by the client sending a TCP-FIN packet and the server acknowledging with a TCP-FIN-ACK packet.

(b) Consider sending over HTTP/2 a Web page that consists of one video file and three images. Suppose that the video clip is transported as 5000 frames, and each image captures four frames.

- If all the video frames are sent first without interleaving, how many "frame times" are needed until all images are sent?
- If frames are interleaved, how many frame times are needed until all three images are sent?

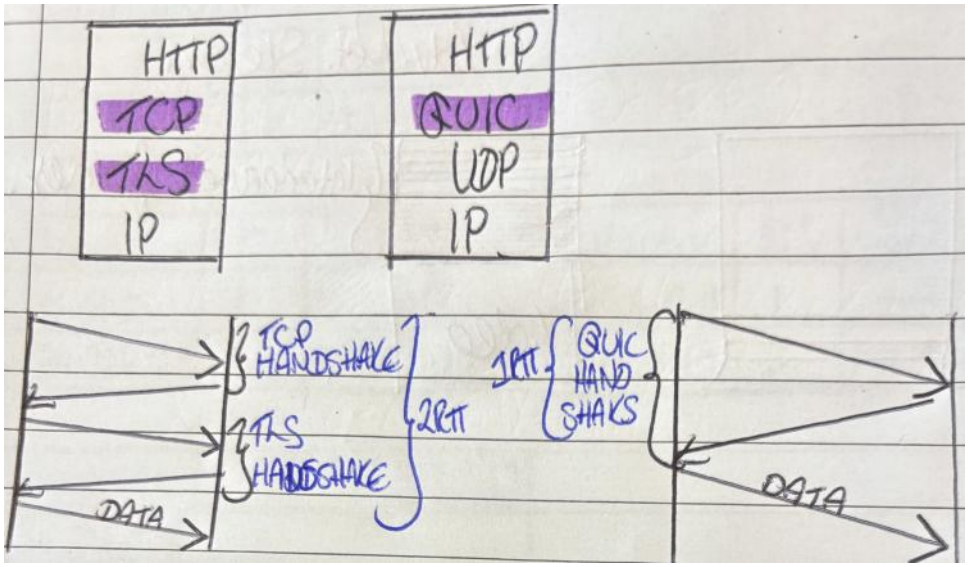
Without interleaving: The frames are sent in order. All the video frames are sent first. It takes 5012 frames to send the three images of four frames per image
With interleaving: The protocol splits the frames up into packets and interleaves the packets of each object. E.g. if the video is Object 1 and the images are Objects 2,3,4. Then one packet from Object 1 is sent, followed by a packet from Object 2, Object 4, etc. At this rate, it takes 16 frame times for all the images to be received.



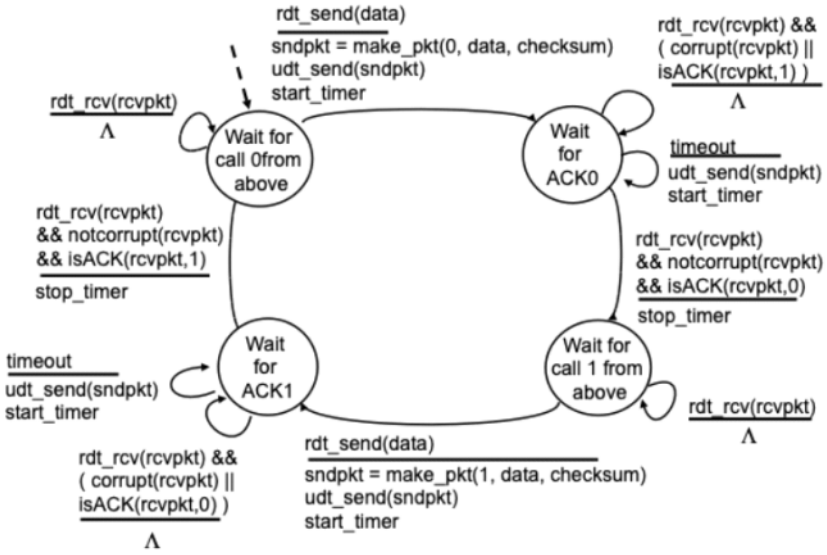
(c) Highlight the differences and advantages of the QUIC protocol in terms of the protocol stack, connection establishment mechanism and efficient delivery of datagrams over existing transport layer protocols such as TCP. Use diagrams to illustrate your answer. [10 marks]

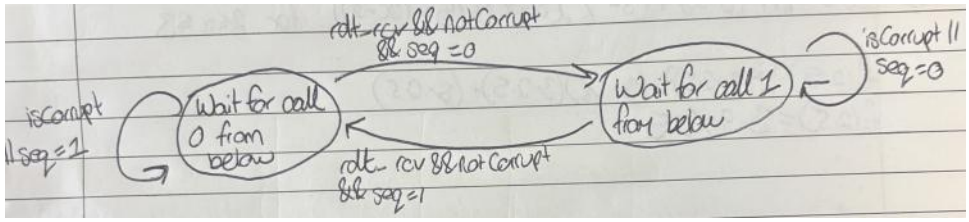
The QUIC protocol combines TCP and TLS into one protocol which sits on top of UDP.
QUIC's connection establishment mechanism and delivery only take one serial handshake instead of the two required for TCP, as seen below. QUIC requires one less round trip time than the use of TCP

and TLS:



(d) Draw the Finite State Machine (FSM) for the receiver side of protocol `rdt3.0` as depicted in the figure below.





2. (a) Show with the aid of a diagram how Alice can efficiently create an "Enveloped and Signed Data" purchase order and send it securely to Bob without Trudy learning any of the details contained within the message. [10 marks]

Alice creates the purchase order.

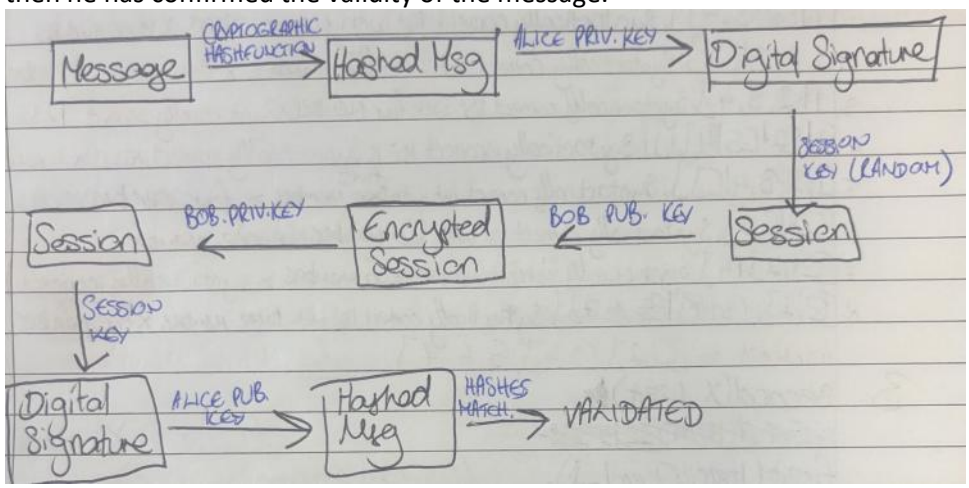
Alice signs the purchase order with a digital signature:

Alice hashes the message with a cryptographic hash function.

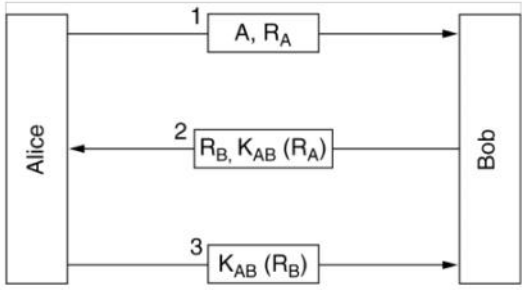
She encrypts the hash with her private key (asymmetric cryptography). This is her digital signature.

She appends her digital signature to the end of the purchase order.

Alice encrypts the message with envelope encryption:
Alice generates a random session key.
She encrypts the entire message using the session key (Symmetric cryptography).
She encrypts the session using Bob's public key and sends the message to Bob.
Bob decrypts the message:
Bob decrypts the session using his private key. He has found the session key.
Bob uses the session key to see the entire message that Alice has sent him, including her digital signature.
Bob verifies the digital signature:
Bob uses Alice's public key to decrypt the hash function/digital signature.
If the hash that Bob has computed matches the hash that Alice has sent him in the message, then he has confirmed the validity of the message.



(b) Why is the shortened two-way authentication protocol shown below vulnerable to a "reflection attack". Modify the diagram to show how such an attack can be mounted.



The key K_{AB} is used in both directions, which allows Trudy to know what this key is used for. When Bob sends back RB, Trudy now has everything needed for a Reflection Attack. She can send back K_{AB} and Bob's challenge (R_B), and Bob thinks he is communicating with Alice.

(d) What are the two fundamental building blocks of the proof-of-work (PoW) scheme that allow the Bitcoin protocol to be secure and popular amongst users? Describe in detail the cryptographic techniques used to achieve "distributed consensus" in the Bitcoin network using the PoW algorithm. [8 marks]

Hash Function: BitCoin miners repeatedly apply SHA-256 until the resulting hash meets a certain condition. Miners are rewarded for helping validate transactions in the blockchain that are not yet validated.
Consensus Mechanism: Everyone in the network agrees on a single valid transaction history. This allows for popularity and security without the need for a TTP

1. (a) In order for a host to be able to send an HTTP request message to a Web server (www.somesite.com), the user's host must first obtain the IP address of

1. (a) In order for a host to be able to send an HTTP request message to a Web server (www.somesite.com), the user's host must first obtain the IP address of the server. Explain the steps through which the client obtains the IP address for such a hostname. Is it possible for an organization's Web server and Mail server to have exactly the same alias for a hostname (e.g. foo.com)? What would be the type of Resource Record (RR) that contains the hostname for the Mail server? [12 marks]

The client obtains the IP address of the server through DNS resolution. A request is made to local DNS requesting the IP address of the URL www.somesite.com. If the website has recently been visited by the client, the IP address may still be in cache and DNS does not need to fetch from the origin server.
It is possible for an organization's web server and mail server to have the same alias as they will have slightly different tuples. The information retrieved from DNS also deals with mail server aliasing.
For example, when a client wants to make a request to the web server and the mail server, the TCP tuple (Source IP Address, Source Port Number, Destination IP Address, Destination Port Number) will differ slightly for the web server and mail server.
For example the web server may have destination IP Address as 134.56.78.1 and the Mail Server may have 134.56.78.2
The RR type for the mail server will be an MX (mail exchange) record.

(b) Suppose Host A sends two segments back to back to Host B over a TCP connection. The first segment has sequence number 65; the second has sequence number 92.

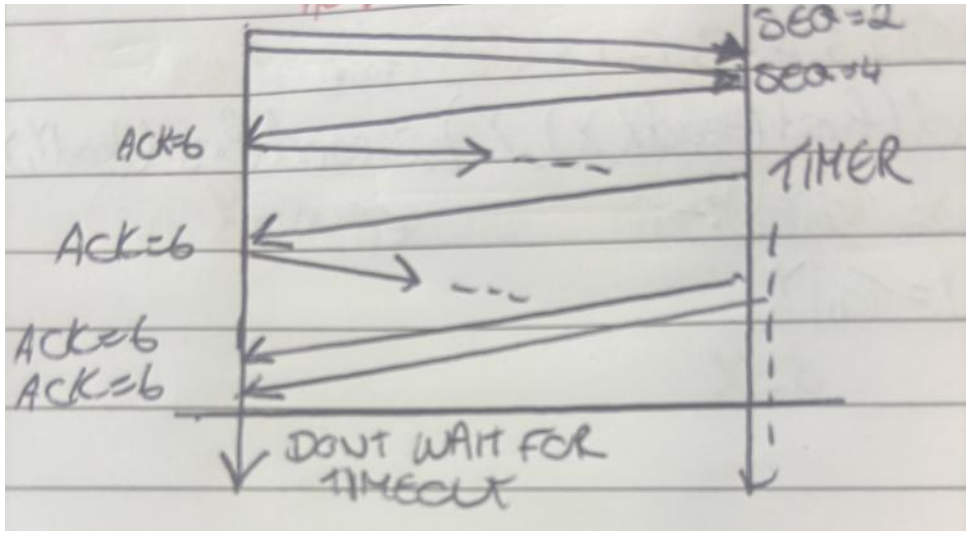
- How much data is in the first segment?
- Suppose the first segment is lost but the second segment arrives at B. In the acknowledgement that Host B sends to Host A, what will be the acknowledgement number?

[12 marks]

92-65 = 27 bytes of data in the first segment
The acknowledgement number will be 65

(c) With the aid of an example describe the TCP "Fast Retransmit" algorithm and its advantages. [8 marks]

TCP fast retransmit has the advantage of saving bandwidth. The timeout period for TCP can be relatively slow to recognise that there are missing packets. If the sender receives 3-duplicate ACKs from the receiver, the sender knows that a packet is failing to be received, therefore the sender should retransmit that packet without waiting for the timeout to finish elapsing.



(e) What type of attacks is the Electronic Code Book (ECB) mode in symmetric key encryption vulnerable to? Explain with the help of examples why the Cipher Block Chaining (CBC) and Output Feedback (OFB) modes can help overcome the vulnerabilities of ECB mode. Why is the OFB mode useful in building a "synchronous stream cipher"? [6 marks]

ECB mode is vulnerable to brute force attacks. In ECB mode, equal plaintext blocks produce equal ciphertext blocks. If an attacker is able to guess commonly occurring letters (e) or commonly used words "money" then they may be able to read the ciphertext.

CBC mode and OFB mode both use XOR operations on the plaintext blocks to ensure that they do not produce identical ciphertext blocks.

CBC mode starts with some random initialisation vector IV and XORs the IV with the plaintext block 1. This produces ciphertext block 1.
Ciphertext block 1 is XORed with plaintext block 2 to produce ciphertext block 2. This is why it is referred to as a "chaining" mode.

OFB mode also starts with an initialisation vector. OFB mode is useful in building a synchronous stream cipher as the blocks of plaintext and output are encrypted before being XORed into ciphertext.
OFB starts with an initialisation vector, the IV is encrypted to produce some Output 1. Output 1 is XORed with Plaintext 1 to produce Ciphertext block 1.
Output 1 is encrypted before being XORed with Plaintext block 2 to produce Ciphertext block 2, and so on.

2. (a) What are the differences between message confidentiality and message integrity? Show how message integrity can be achieved using symmetric and asymmetric key cryptographic techniques. [10 marks]

Message confidentiality ensures that the message is being sent and received only by the intended sender and recipients.
Message integrity refers to ensuring that the received message has not been intercepted or changed in any way by a man in the middle attack.
Message integrity can be achieved by first encrypting the message with asymmetric key algorithms I.e. the sender's private key. This creates a digital signature. The entire message with the digital signature can be sent to the recipient with a session key which uses symmetric key cryptographic algorithms. The sender can once again use asymmetric cryptography to encrypt the entire session with the recipient's public key. The recipient must decrypt the session with their private key to reveal the symmetric session key and use the sender's public

key to decrypt the message. If cryptographic hashing functions are also applied, then the recipient can also validate message integrity.

(b) Describe some of the components that comprise modern day block ciphers? In particular describe with the aid of an example the Vigenère Cipher. [8 marks]

The Vigenére Cipher works by taking some word e.g. "fire" as an offset.
The alphabet is written out:
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
Let the sentence being encrypted be:
THIS IS A SENTENCE = 19 7 8 18 8 18 0 18 4 13 19 4 13 2 4
FIRE FI R EFIREFIR = 5 8 17 4 5 8 17 4 5 8 17 4 5 8 17
The plaintext is offset by the alphabetic position of the offset

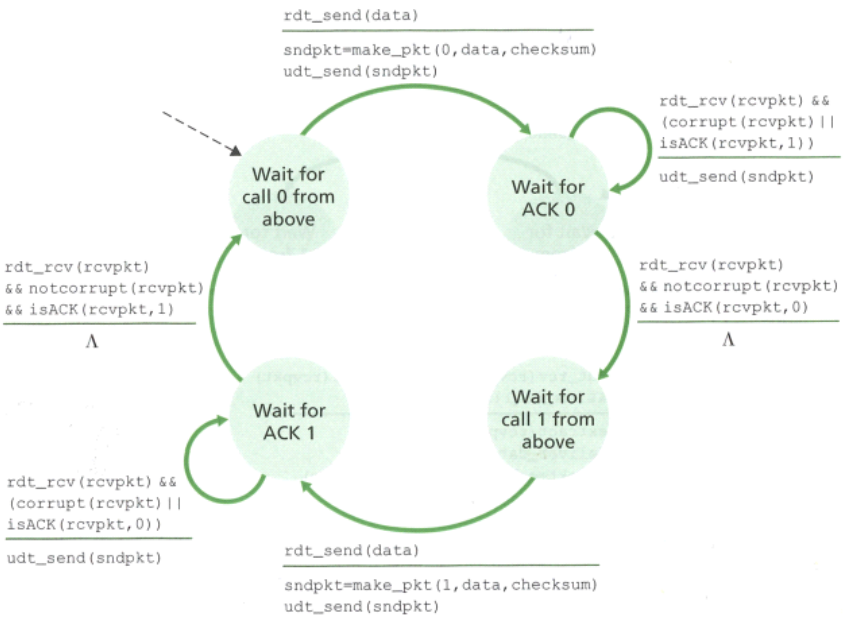
(c) Suppose that a system uses a Public Key Infrastructure (PKI) based on a tree-structured hierarchy of Certification Authorities (CAs). Alice wants to communicate with Bob, and receives a certificate from Bob signed by a CA X after establishing a communication channel with Bob. Suppose Alice has never heard of X but knows the public-key of the root CA. Explain in detail what steps Alice needs to take to verify that she is talking to Bob? [10 marks]

Alice should obtain Bob's certificate and extract X's identity from the message.
Since Alice does not know X, she sees who has signed X's certificate, which may be another CA that is not the root CA.
Alice keeps doing this until she reaches the root CA which she can validate.
Once this is completed, Alice knows that she can trust and communicate with Bob

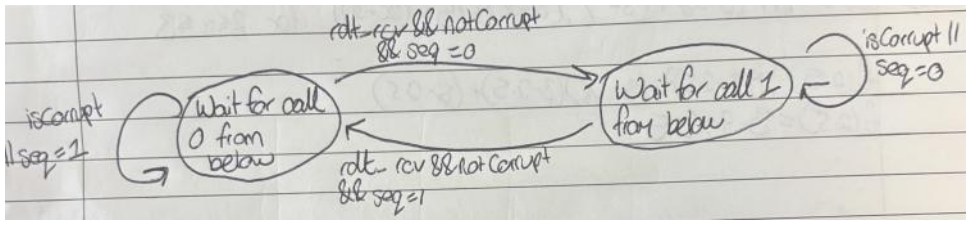
1. (a) Explain how the transmission control protocol (TCP) uses the services of an unreliable network layer to provide reliable end-to-end communications? Show with the aid of an example how TCP can correctly demultiplex packets arriving from two hosts (A and B), who by coincidence happen to pick the same source port number (5099), and are communicating with the same web server on port 80. [8 marks]

The TCP tuple is a four tuple consisting of (Source IP Address, Source Port Number, Destination IP Address and Destination Port Number).
We know that host A and host B have chosen the same source port, lets say 6 and they are communicating with the same destination web server, lets say 132.45.36.1 on port 80.
Therefore the two TCP Tuples are as follows:
(Host A Source IP, 6, 132.45.36.1, 80)
(Host B Source IP, 6, 132.45.36.1, 80)
As long as Host A and Host B have different Source IP Addresses e.g. 100.90.80.1 and 100.9.80.2, they will have different TCP tuples and can both safely communicate with the same web server

(b) The figure below depicts a finite state machine (FSM) for a NAK-free reliable data transfer protocol for a channel with bit errors.



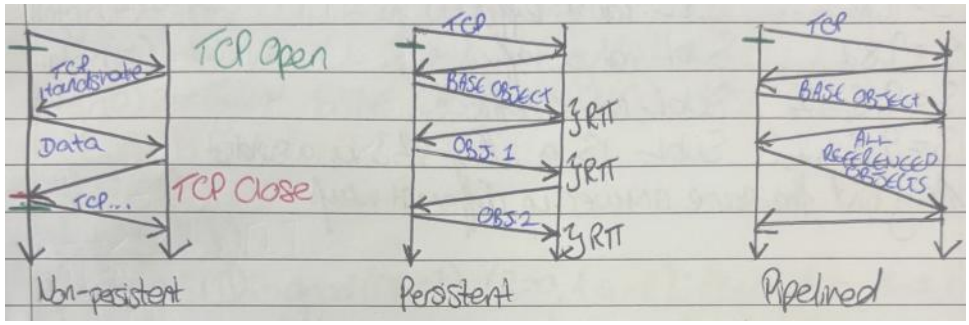
Draw the corresponding FSM for the receiver side of the protocol. What additions are required to the FSM on the receiver side in order to use an ACK only primitive? [14 marks]



In order for a NAK-free protocol to work, sequence numbers needed to be added to the ACKnowledgements, this allows ACKS to have a value of 0 or 1. If the opposite sequence number than expected is received, that acts as a NAK without the overhead of a NAK

(d) With the aid of a diagram distinguish between "persistent" and "non-persistent" HTTP connections in terms of the round trip time (RTT) and TCP overhead. What improvements can "pipelining" bring to a persistent HTTP connection in terms of the RTT? [7 marks]

Non-Persistent: in non-persistent HTTP, the TCP connection is closed. At most one object can be transmit per each TCP instantiation.
Persistent: in persistent HTTP, the TCP connection is kept alive and does not need to be re-instantiated each time. But only one object is sent per RTT
Persistent with Pipelining: with pipelining, after the base object is sent, then all of the referenced objects can be sent which greatly reduces the time taken to transmit objects,



(e) Imagine a peer-to-peer network (P2P) where N users want to communicate in an authenticated and confidential manner without a centralized trusted third party (TTP).

i. How many keys are collectively needed if symmetric key algorithms are deployed?

ii. How are the numbers changed if we make use of a key distribution center (KDC)?

iii. How many keys are needed if we make use of asymmetric key algorithms?

[9 marks]

Symmetric: n(n-1)/2
KDC: n
Asymmetric: 2n

2. (a) One of the most attractive applications for public-key cryptography is the establishment of a secure "session key". In practice it is desirable that both communication parties influence the selection of the session key, so as to prevent one party from choosing a *weak key*. Develop a protocol in which both Alice and Bob who posses a pair of public/private keys of the RSA cryptosystem are able to influence the selection of the session key. [8 marks]

(e) What are the advantages of a virtual private network (VPN) over a dedicated private secure Internet link? Describe the functionality of the IPsec "Transport" and "Tunnel" modes. In particular, describe with the aid of a diagram how the header and payload of an IP datagram can be protected by deploying IPsec in "Tunnel Mode with ESP" (ESP - encapsulation security payload). Note that you are required to identify the relevant IPsec header and trailer fields. [12 marks]

Mathematical Questions

Wednesday, April 30, 2025 10:40 AM

(e) Consider distributing a file $F = 10$ Gbits to N peers. The server has an upload rate of $u_s = 1$ Gbps, and each peer has a download rate of $d_i = 200$ Mbps and an upload rate of u. For $N = 10$, 100 and 1,000 and $u = 2$ Mbps, 10 Mbps and 100 Mbps, prepare a table giving the minimum distribution time in seconds for each of the combination of N and u for both Client-Server and P2P distribution. [12 marks]

(c) Compute the two public keys and the common key for the Diffie-Hellman Key Exchange (DHKE) scheme with the parameters $p = 467$ and $\alpha = 2$, $a = 3$ and $b = 5$. Perform the computation for the common key for Alice and Bob. [10 marks]

(e) The parameters for Elliptic Curve Digital Signature Algorithm (ECDSA) are given by the curve $E : y^2 = x^3 + 2x + 2 \ mod\ 17$, the point $A = (5, 1)$ of order $q = 19$ and Bob's private key $d = 10$. Compute Bob's public key (B) and his signature (r, s) on the hash value $h(m)$. Show the verification process by Alice for the hash value $h(m) = 12$ and key $K_E = 10$ by computing the values of B, R, s, w, u_1, u_2 and P as shown in the diagram beow. You can assume that the final computed value of $P = (7, 11)$.

(d) Consider distributing a file $F = 15$ Gbits to N peers. The server has an upload rate of $u_s = 30$ Mbps, and each peer has a download rate of $d_i = 2$ Mbps and an upload rate of u. For $N = 10$ and 100 and $u = 300$ Kbps and 700 Kbps calculate the minimum distribution time for both client-server and P2P configurations. [12 marks]

(d) To show that you understand the security of the RSA algorithm, find d if you know that $e = 17$ and $n = 187$. [12 marks]

(e) The parameters for Elliptic Curve Digital Signature Algorithm (ECDSA) are given by the curve $E : y^2 = x^3 + 2x + 2 \ mod \ 17$. the point $A = (5, 1)$ or order $q = 19$ and Bob's private key $d = 7$. Compute Bob's public key (B) and his signature (r, s) on the hash value $h(m)$. Show the verification process by Alice for the hash value $h(m) = 26$ and key $K_E = 10$. You can assume that the value of $P = (7, 11)$. [10 marks]

(c) Consider transferring an enormous file of L bytes from Host A to Host B. Assume a maximum segment size (MSS) of 536 bytes.

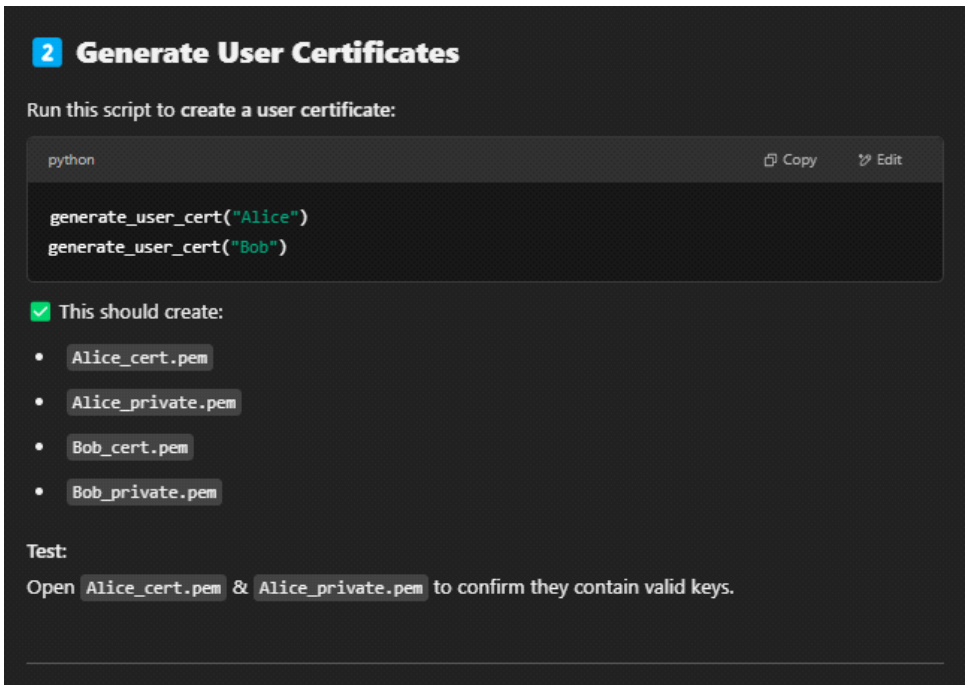
i. What is the maximum value of L such that the TCP sequence numbers are not exhausted?

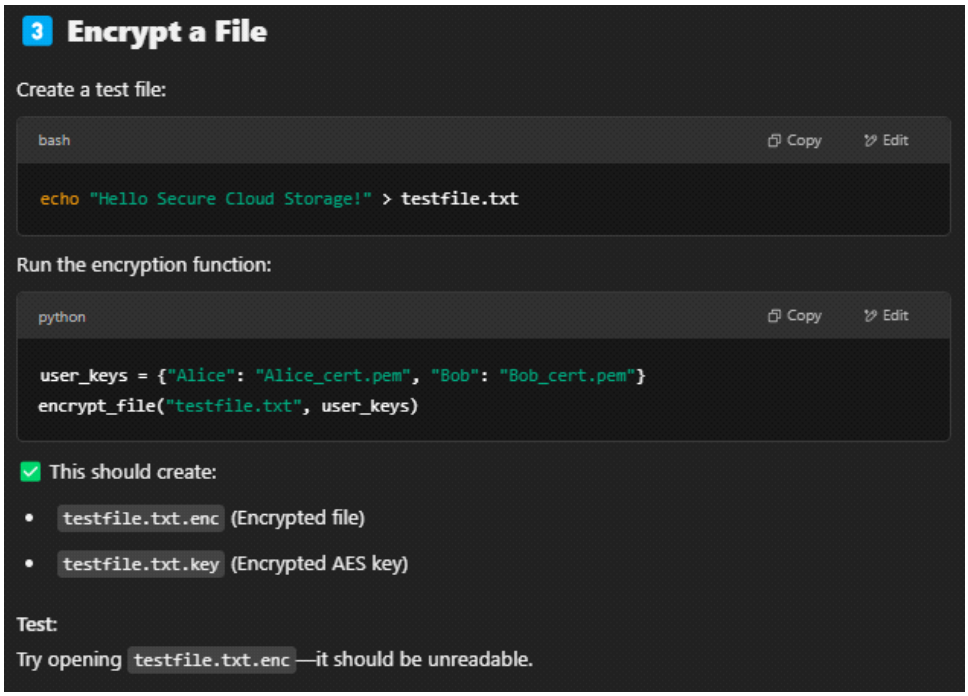
ii. For the L you obtain in (i), find how long it takes to transmit the file? Assume that a total of 66 bytes of transport, network and data-link header are added to each segment before the resulting packet is sent out over a 155 Mbps link. Ignore flow and congestion control, and assume that Host A can pump out the segments back to back and continuously.

(b) Using the Miller-Rabin primality test show how one can assume with a high probability that the number 269 is a prime number, given that the decomposition of an odd prime candidate can be represented by $\tilde{p} - 1 = 2^u \cdot r$, where r is odd. [10 marks]

Tests

Tuesday, March 18, 2025 10:21 AM





4 Upload to Cloud

Upload the encrypted files:

```python
upload_to_cloud("testfile.txt.enc")
upload_to_cloud("testfile.txt.key")
```

✅ **Test:**

- Check Google Drive / Dropbox to see if `testfile.txt.enc` and `testfile.txt.key` were uploaded.

---

5 Decrypt the File

Run the decryption function:

```python
decrypt_file("testfile.txt.enc", "testfile.txt.key", "Alice_private.pem")
```

✅ **Expected Output:**

A file named `decrypted_testfile.txt` should appear, containing:

```nginx
Hello Secure Cloud Storage!
```

Test: Open `decrypted_testfile.txt` and check if it matches `testfile.txt`.

6 Add a New User

```python
generate_user_cert("Charlie")
add_user_to_group("Charlie", "Charlie_cert.pem")
```

✅ **Test:** Check if `Charlie`'s public key is now in `group_keys.json`.

---

7 Remove a User

```python
remove_user_from_group("Bob")
```

✅ **Test:**

Bob should no longer be able to decrypt `testfile.txt.enc`.

8 Final Security Check

Try decrypting `testfile.txt.enc` with **Bob's private key** after removal:

```python
decrypt_file("testfile.txt.enc", "testfile.txt.key", "Bob_private.pem")
```

❌ **Expected Output:**

Bob should get an **"Access Denied!"** error.