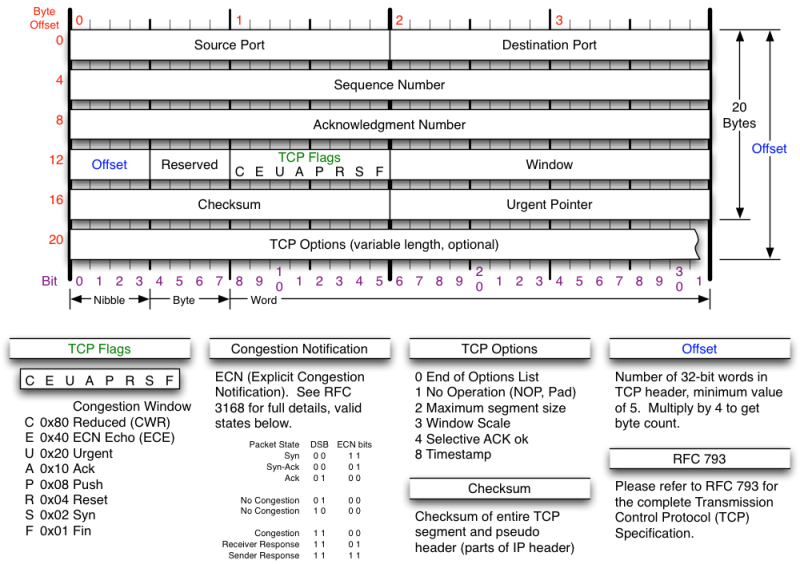
CS241 #36 Protocols at the speed of light

**1> Protocol at the TCP level - Handshaking**

TCP Packets: What is "SYN. SYN-ACK. ACK" ?

What is the sequence number and what is it used for? What is its initial value & why?

(I see the port number but where is the machine's IP address?)



**2> Examples of Denial of Service**

SYN flood

Distributed DOS

"Internet of Things DOS"

**3> Which TCP client or server call will result in the first "SYN" packet?**

**4> TCP Handshaking and the speed of light**

The moon is 1.3 light seconds distant. The TCP client is on the Earth and a lunar console runs a TCP server. Assume a new TCP connection is required each time.

3.1 Save the astronaut. How many seconds elapse between wanting to send a CLOSEAIRLOCK message and the server receiving the data?

fd=socket(...)  
connect(fd,...,...)  
write(fd,"CLOSEAIRLOCK!",13);

3.2 How many seconds elapse between requesting data from the server and receiving the result?

fd=socket(...)  
connect(fd,...,...)  
write(fd,"READTEMP!",9);

bytes=read(fd,buffer,256);

**5> TCP and web performance**

HTTP/1.0

If the client-server round trip time is 10 ms. What is the minimum time required to display a page with an image? Assume HTTP/1.0 (and that the image requires a separate request).

**6> Better... Faster...**

Performance improvements in HTTP/1.1

Improvements in HTTP/2.0

Why did Google create QUIC ?

**7> Remote Procedure Calls**

**void updateScoreBoard(char\*name, int score) {**

**char\* mesg; // todo: error checking!**

**asprintf(&mesg, "newscore,%s,%d",name,score);**

**write( fd, mesg, strlen(mesg+1));**

**free(mesg);  
 // Why did I also send the null byte?**

**}**

**// You could also send the message size  
// My protocol! So I'll choose bigendian binary format**

**uint16\_t mesglen = htons( strlen(mesg) );  
write( fd, & mesglen, sizeof(mesglen) );**

**write( fd, mesg , strlen(mesg) );**

**8> Subverting protocols**

**Case study: Heartbleed April 2014**

**/\* simplified \*/**

**sock\_fd = accept(server\_fd);**

**while(1) {**

**secureread(sock\_fd, &request, &reqsize);**

**switch(request->request\_type) {**

**case HEARTBEAT:**

**// echo the client message back**

**securewrite(sock\_fd,**

**request->content,**

**request->content\_length**

**);**

**break;**

**case (...):**

**...**

**break;**

**}**

**free(mesg);**

**}**

Change ONE character to fix this program to print 20 dashes.

#include <stdio.h>

int main()

{

int a;

int b = 20;

for( a = 0; a < b; a-- )

putchar('-');

return 0;

}

Actually there are THREE solutions (again, changing only ONE character). Find all three.

Bonus: Change one character to print 21 dashes.

Heartbleed

Heartbleed is a security bug disclosed in April 2014 in the OpenSSL cryptography library, which is a widely used implementation of the Transport Layer Security (TLS) protocol. Heartbleed may be exploited regardless of whether the party using a vulnerable OpenSSL instance for TLS is a server or a client. It results from improper input validation (due to a missing bounds check) in the implementation of the TLS heartbeat extension, thus the bug's name derives from "heartbeat". The vulnerability is classified as a buffer over-read.

17% of all web servers were vulnerable

In June 2014, 300,000 systems still vulnerable.

