CS241 #24 Simple TCP Client Example + HTTP

What is the OSI (**Open Systems Interconnection)** Model?

OSI Model				
Layer		Data unit	Function[3]	Examples
Host layers	7. Application	Data	High-level APIs, including resource sharing, remote file access, directory services and virtual terminals	HTTP, FTP, SMTP
	6. Presentation		Translation of data between a networking service and an application; including character encoding, data compression and encryption/decryption	ASCII, EBCDIC, JPEG
	5. Session		Managing communication sessions, i.e. continuous exchange of information in the form of multiple back-and-forth transmissions between two nodes	RPC, PAP
	4. Transport	Segments	Reliable transmission of data segments between points on a network, including segmentation, acknowledgement and multiplexing	TCP, UDP
Media layers	3. Network	Packet/Datagram	Structuring and managing a multi-node network, including addressing, routing and traffic control	IPv4, IPv6, IPsec, AppleTalk
	2. Data link	Bit/Frame	Reliable transmission of data frames between two nodes connected by a physical layer	PPP, IEEE 802.2, L2TP
	1. Physical	Bit	Transmission and reception of raw bit streams over a physical medium	DSL, USB

Image Attribution: http://en.wikipedia.org/wiki/OSI_model

What is "U.D.P." and what are its main characteristics?

What is T.C.P. and what are its main characteristics?

Which one uses handshaking?

Which one requires more system resources?

Which one can be used with `read` and `write` system calls?

Which one encrypts the data payload?

If your application preferred to handle missing packets over late packets, which one would you use?

What is HTTP? Does it run over TCP or UDP?

Is HTTP version 1.0 and version 1.1 a text or binary protocol?

How do you make a TCP connection to a server?

What is the purpose of

`getaddrinfo`

`struct addrinfo`

`Why memset`

`AF_INET`

`SOCK_STREAM`

```
struct addrinfo {
int              ai_flags;
int              ai_family;
int              ai_socktype;
int              ai_protocol;
socklen_t        ai_addrlen;
struct sockaddr *ai_addr;
char            *ai_canonname;
struct addrinfo *ai_next;
};
```

```
int getaddrinfo(char*host,char *service, addrinfo* hints, addrinfo **res);

int socket(int domain, int type, int protocol);

int connect(int socket, struct sockaddr *address, socklen_t address_len);
```

```
01  int main() {
02    struct addrinfo _________, ____________;

03    memset(&hints, 0, sizeof(_________,) );

04    hints.ai_family = _________;

05    hints.ai_socktype = ____________________;

06    int s = getaddrinfo("illinois.edu", _________ , _________, _________);
07    if (s != 0) {
        fprintf(stderr, "getaddrinfo: %s\n", gai_strerror(s));
        exit(1);
08    }
09    int sock_fd = socket(_________, _________ , 0);
10    if(sock == -1) { perror("socket"); exit(1);}

11    int ok = connect(sock_fd, ____________________, ____________________________);
12    if( ok ==-1) {perror("connect"); exit(1);}
13
14
15
16
17
18
19
```

IPv4 Header Format

Offsets	Octet	0								1								2								3							
Octet	Bit	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	26	27	28	29	30	31
0	0	Version				IHL				DSCP						ECN		Total Length															
4	32	Identification																Flags			Fragment Offset											
8	64	Time To Live								Protocol								Header Checksum															
12	96	Source IP Address																															
16	128	Destination IP Address																															
20	160	Options (if IHL > 5)																															

TCP header:

Offsets	Octet	0								1								2								3							
Octet	Bit	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	26	27	28	29	30	31
0	0	Source port																Destination port															
4	32	Sequence number																															
8	64	Acknowledgment number (if ACK set)																															
12	96	Data offset				Reserved 0 0 0				NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size														
16	128	Checksum																	Urgent pointer (if URG set)														
20 ...	160 ...	Options (if *data offset* > 5. Padded at the end with "0" bytes if necessary.) ...																															

Image attribution – wikipedia.com

https://xkcd.com/195/

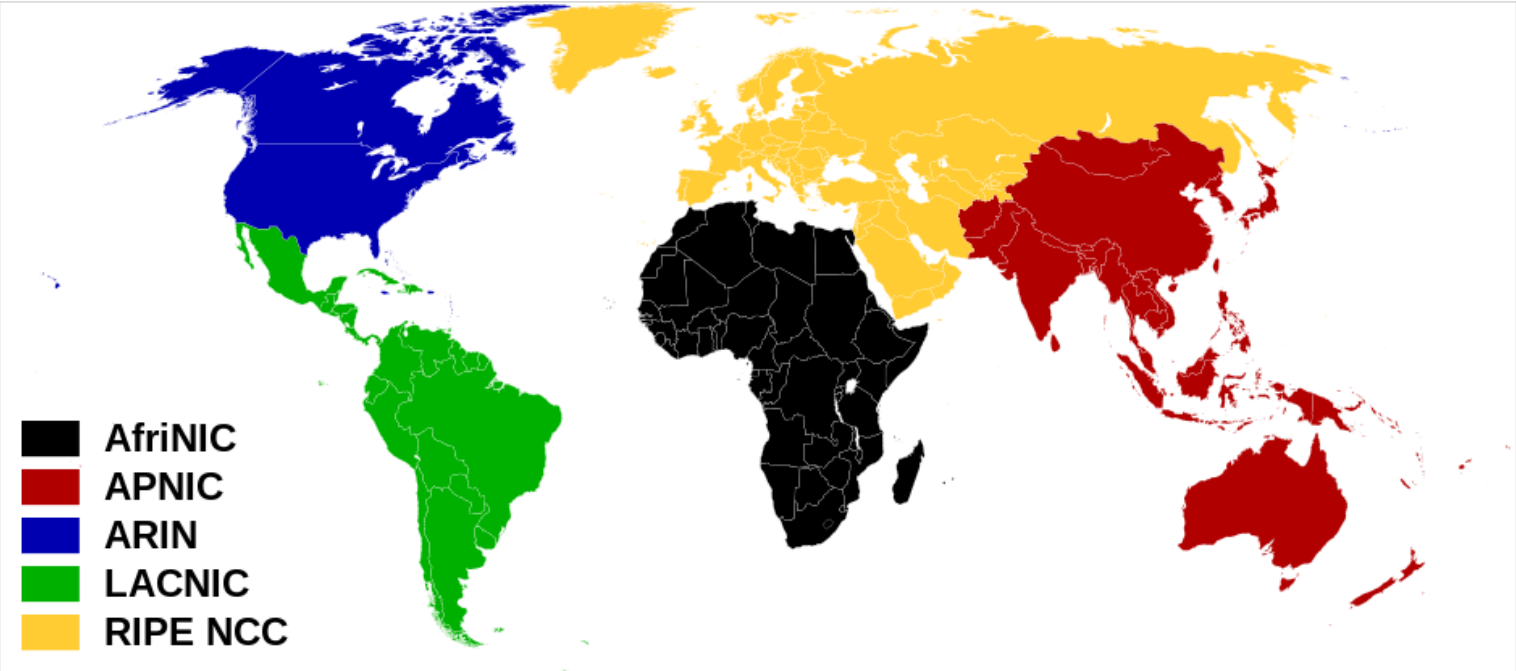
MAP OF THE INTERNET

THE IPv4 SPACE, 2006

THIS CHART SHOWS THE IP ADDRESS SPACE ON A PLANE USING A FRACTAL MAPPING WHICH PRESERVES GROUPING -- ANY CONSECUTIVE STRING OF IPs WILL TRANSLATE TO A SINGLE COMPACT, CONTIGUOUS REGION ON THE MAP. EACH OF THE 256 NUMBERED BLOCKS REPRESENTS ONE /8 SUBNET (CONTAINING ALL IPs THAT START WITH THAT NUMBER). THE UPPER LEFT SECTION SHOWS THE BLOCKS SOLD DIRECTLY TO CORPORATIONS AND GOVERNMENTS IN THE 1990's BEFORE THE RIRs TOOK OVER ALLOCATION.

socket
listen
accept

Exhaustion of IPv4 for each of the 5 regional authorities.
ARIN exhausted 24 September 2015
commons.wikimedia.org/wiki/File:Regional_Internet_Registries_world_map.svg



https://www.arin.net/resources/request/ipv4_countdown.html