CMSC417 Lecture #1 2/3/2016

Agenda

⇒ Syllabus (prereqs, books, late work, getting help)
⇒ Website (github)
⇒ Project Ø (github)
⇒ Piazza
⇒ Introduce Myself
⇒ Introduce TAs
⇒ Major Topics: Layering, Routing, Sockets, IP, BGP, P2P, TCP, DNS, Apps (HTTP, SMTP, ...), MAC, Wireless, SDN, MPLS?, Middleboxes?, NFV?
⇒ You can pick more topics

⇒ Intro/Layering
⇒ Sockets

CMSC 417 Spring 2016 Lecture #1 2/3/2016

Basic Networking (Layering)

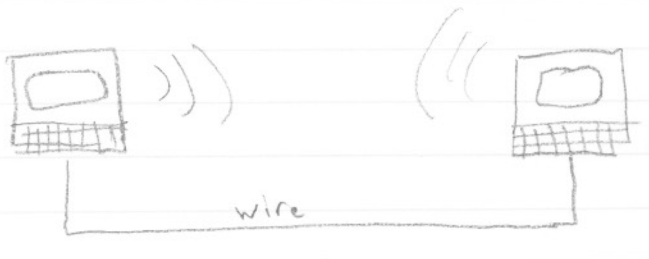
	OSI model
apps	7 apps (everything else)
process-to-process	6 presentation (data format)
host-to-host	5 session (logical connections)
h/w	4 transport (streams, big things, reliable)
	3 network (packets w/destinations)
	2 link (frames, grouped bits w/ends) dev-to-dev
	1 phy (physics, bits, voltages)

in reality, layers 1-4 and 7 are all that's talked about

convergence layers, e.g., ARP, which we'll talk about later

In reality

7	lots
4	TCP / UDP
3	IP
2	Ethernet 802.11 or 802.3
1	lots



what if we had 3? 100?

what about multiple apps per computer?

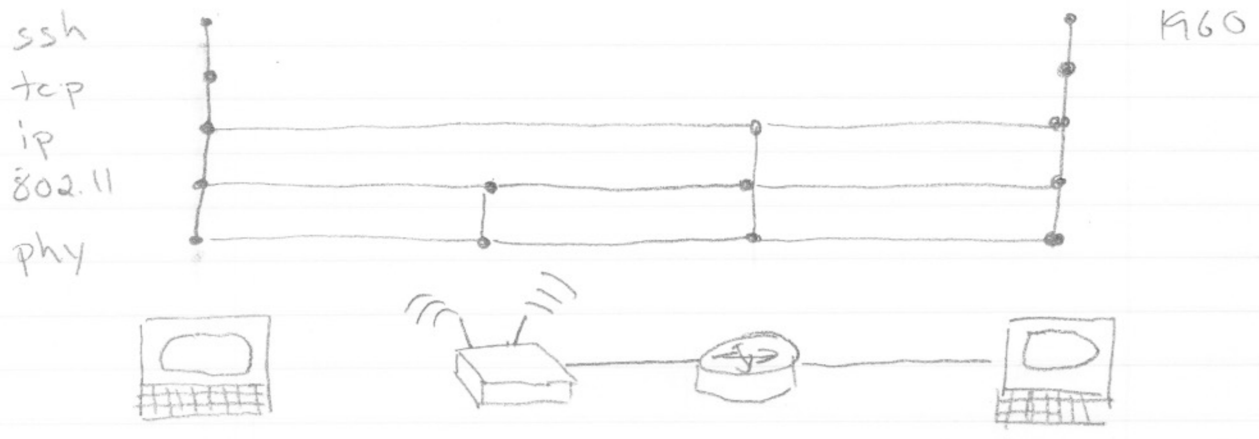
CMSC 417 Spring 2016 Lecture #1 2/3/2016

Layering continued

⇒ each layer typically has a header
⇒ layers are stacked

PHY
LINK
NETWK
TRANSPRT

⇒ different headers are stripped/added as appropriate



⇒ seperation of concerns
⇒ modularity
⇒ solve easier problems

CMSC 417 Spring 2016 Lecture #1 2/3/2016

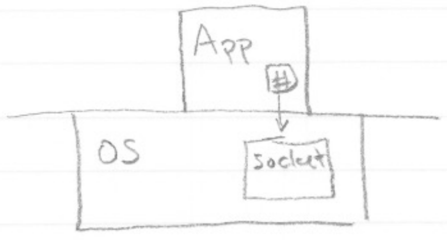
Sockets

socket() / close()	sockets
connect() / send() / recv()	TCP client
bind() / listen() / accept()	TCP server
sendto() / recvfrom()	UDP

⇒ inout parameters passed as pointers

Logically, a socket is something inside the OS.
⇒ socket() creates one
⇒ close() destroys one

Other calls tell the OS how it should make the object behave and allow you to retrieve information it's storing.



① allocate a socket and give me a pointer to it
② use the following local and/or remote address
③ get my messages (if any)
④ send a message