

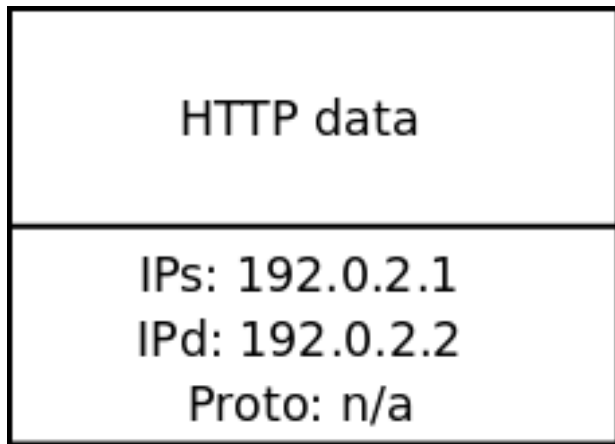
# Network Protocols

## Transport Layer and UDP (TCP in another slide-deck)

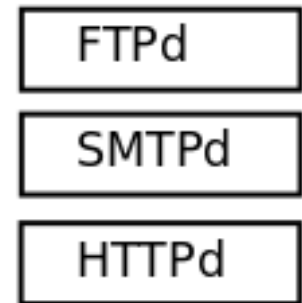
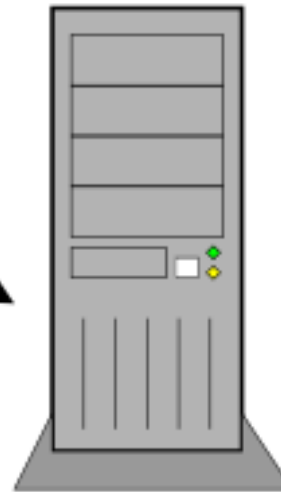
# Why a transport layer?

- IP gives us e2e connectivity doesn't it?
- Why or why not >1 transport layer?
- What does a transport layer typically do?
  - Process identification
  - Reliability
  - Flow control
- Are there times when those are unnecessary?
- What are the security / performance issues?

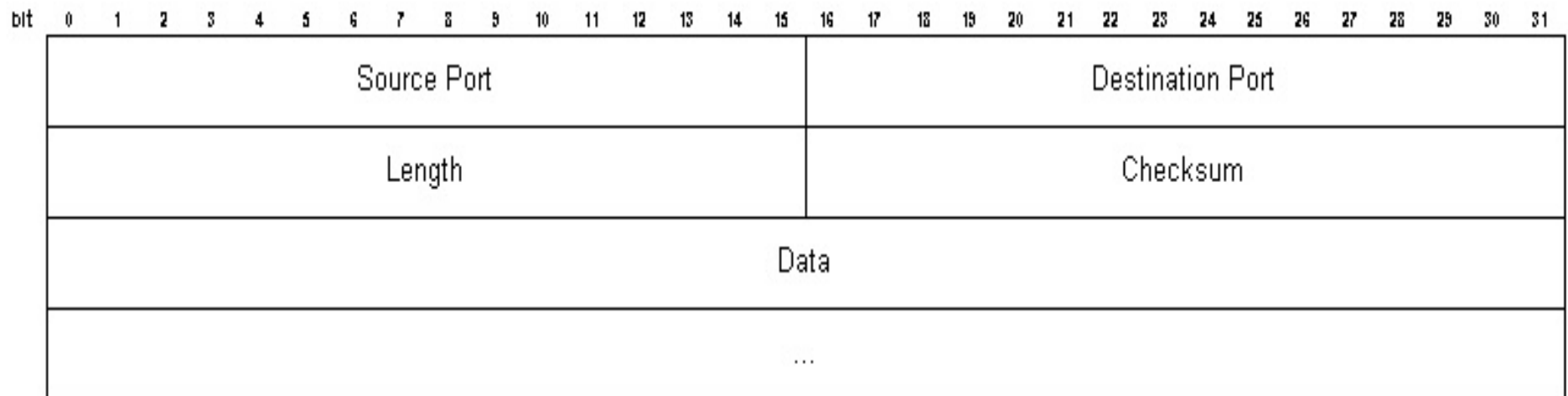
# If no L4, what to do with rx data?



Thanks!!  
But...  
How to get this  
to my to HTTPd?  
Try to parse and interpret!?!?  
Help me out dude!!



# User Datagram Protocol (UDP)



# Application multiplexing

- OS independent identifier for a network app
- Each app assigned a locally unique 16-bit id
  - src or dst “port number”, see /etc/services
- Server (listener) apps
  - Tend to use standard, “well-known” ports
- Client (opener) tends to ephemeral (dynamic) port
  - Usually  $>1023$ , but depends on OS and app
- See <http://www.iana.org/assignments/port-numbers>

# UDP is very simple

- Basically just an application multiplexer
- The length field is practically redundant
  - $\text{IP total length} - 8 = \text{UDP payload}$
- Source port is zero (unused) if no reply expected
- Even the checksum is optional!
  - Though its inexpensive, recommended to use it
- No inherent flow control, reliability
- Why is this good, how could this be bad?

# What uses UDP?

- SNMP, TFTP, DHCP, syslog, Netflow
- Streaming media (VoIP, radio, video)
  - Some use TCP (thank you security monkeys)
- DNS, NTP
  - UDP filtering likely (hi monkeys) if not for these

# Common IP transport protocols

- UDP – very common, but overall low rate of pkts
- TCP – most common, typically most of your pkts
- Some usage, but not widely deployed:
  - UDP-Lite
  - SCTP
  - DCCP
- Note, not all IP protocols considered a “transport”
  - e.g. we don't think of ICMP/IGMP as a transport