* networking cont.

Socket layer (logically at same level as VFS layer):
- abstraction of a "network connection".
- "struct socket" -- similar to "struct inode" in file systems

Protocol Drivers:
- more specific objects, per protocol
- e.g., TCP/UDP keep a "struct sock" for maintaining any specific state
- different "struct sock" implementations for UDP, TCP, etc.
  logical abstraction: same as "struct ext3_inode"
- struct sock for TCP maintains a lot of running state
- UDP doesn't need a struct sock

Firewall chains, aka IPchains, now called IPTables
- pre/post processing on packets, and take action
  pre-process: matching against a pattern
  action: drop, reject (w/ "reset"), forward
  actions: modify packet, or defer processing

drop: filter packets (firewall)
Modify packet: useful for D/SNAT
forward: redirect to a different subnet

defer processing:
- can inspect IP header for src/dst IP addrs
- but port numbers are in UDP/TCP headers
- so may have to assemble several IP packets just to get a full UDP/TCP packet, to inspect its headers.
- sometimes have to wait for processing at the "application" layer

e.g., look for HTTP message, filter based on URLs listed, or
undesired java scripts

e.g., FTP -- uses TCP port 21 as control channel, and a per-download/upload
UDP port number for each file up/download.
- when you say "get foo.zip" over the control channel, FTP sends a text
formatted message on the ctl channel such as "file foo.zip port 12345"
- have to parse string and set up bi-directional port translation (at application layer)

Linux has lot of "net=filter" (NF) hooks in the networking subsystem.
- can inject custom code into any hook, to control pre/post actions, from logging to pkt manipulation of any sort.

packets processed may go up/down layers, and even jump to other parts
(IPtables).

BPF: Berkeley Packet Filter systems (from BSD OSs)
- eBPF: Extended BPF

ICMP: Internet Control Message Protocol
- different control messages
  ICMP RESET: close a connection
  ICMP REDIRECT: divert packets from IP X to IP Y
  ICMP ECHO: ping
- traceroute: using TTL field set to 1, 2, 3, ...

ARP: Address Resolution Protocol
- translates b/t IP addresses and hardware (MAC) addresses
- every host keeps this association, so you can fill in headers of Ethernet frames
- ARP broadcasts of new associations
- ARP requests for "who has IP 1.2.3.4"

* history of locking in Linux

In the beginning, there was just one Big Kernel Lock (BKL)
- hurts concurrency, everyone has to grab BKL for all d-s
- simple to use/program
- minimize deadlocks, but not entirely (self-deadlock)
- doesn't eliminate all races: if not grabbing BKL

Over time, locks at layer N were broken up into smaller locks and pushed down
- many more locks
- not simple any longer, more complex locking semantics
- PERFORMANCE, PERFORMANCE, PERFORMANCE