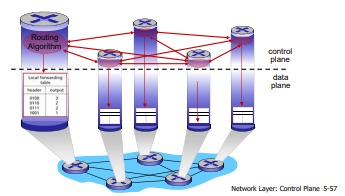
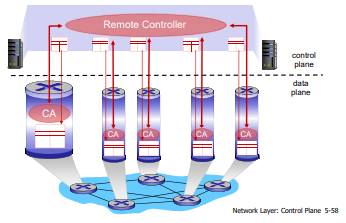
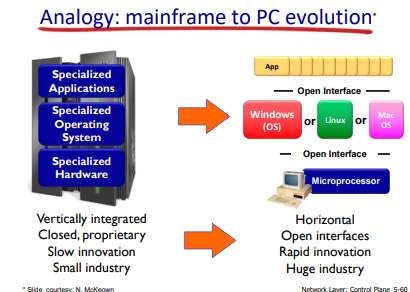
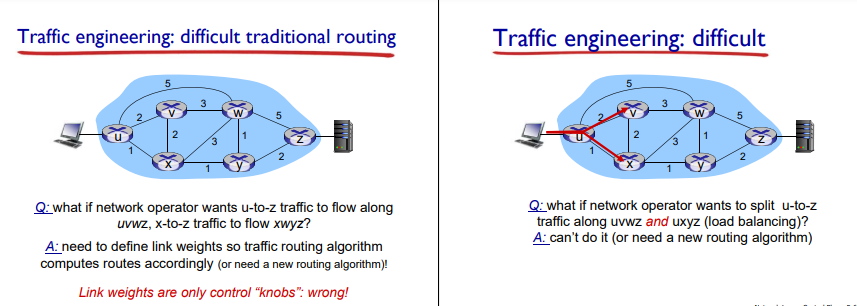
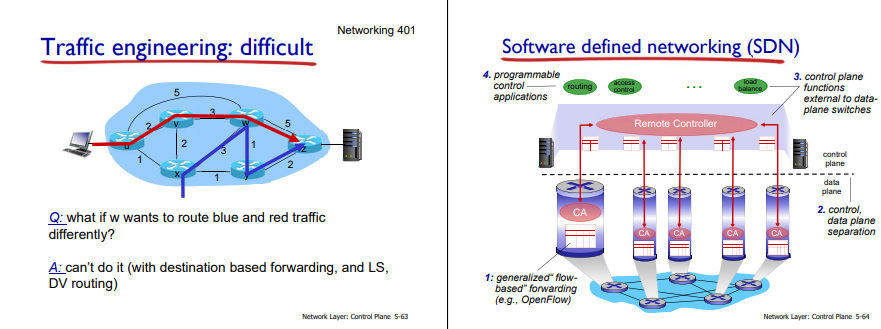
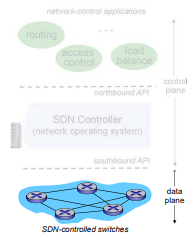
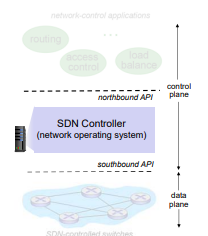
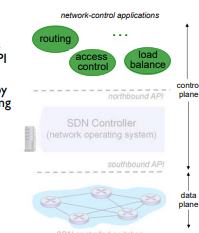
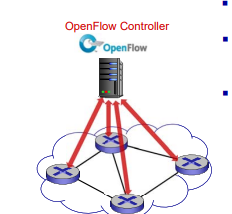
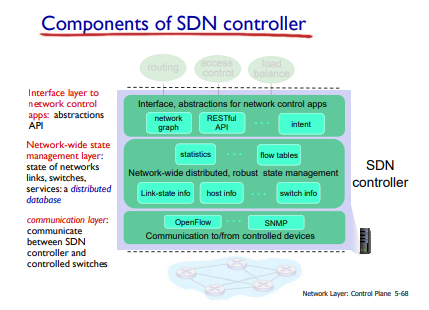
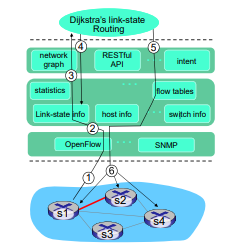
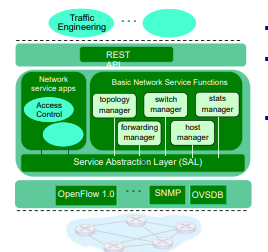
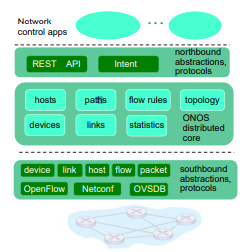
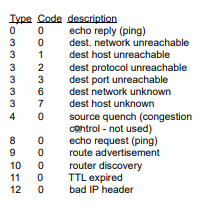
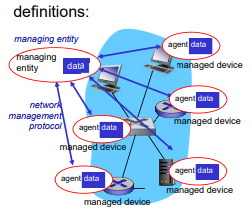
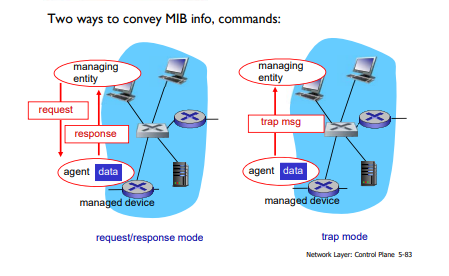
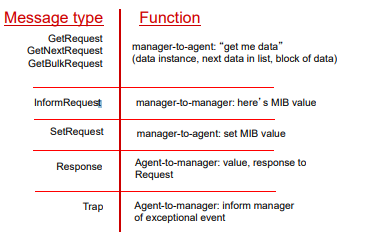
* Software Defined Networking
  + Internet network layer: historically has been implemented via distributed, per-router approach
    - monolithic router contains switching hardware, runs proprietary implementation of Internet standard protocols (IP, RIP, IS-IS, OSPF, BGP) in proprietary router OS (e.g., Cisco IOS)
    - different “middleboxes” for different network layer functions: firewalls, load balancers, NAT boxes, ..
* Recall: Per-router Control Plane
  + Individual routing algorithm components in each and every router interact with each other in control plane to compute forwarding tables
  + 
* Recall: Logically Centralized Control Plane
  + A distinct (typically remote) controller interacts with local control agents (CAs) in routers to compute forwarding tables
  + 
* Software Defined Networking (SDN)
  + Why a logically centralized control plane?
    - easier network management: avoid router misconfigurations, greater flexibility of traffic flows
    - table-based forwarding (recall OpenFlow API) allows “programming” routers
    - • centralized “programming” easier: compute tables centrally and distribute
    - • distributed “programming: more difficult: compute tables as result of distributed algorithm (protocol) implemented in each and every router
  + open (non-proprietary) implementation of control plane
* Analogy: Mainframe to PC Evolution:
  + 
* Traffic Engineering: Difficult Traditional Routing





* SDN Perspective: Data Plane Switches
  + Data plane switches
  + fast, simple, commodity switches implementing generalized data-plane forwarding (Section 4.4) in hardware
  + switch flow table computed, installed by controller
  + API for table-based switch control (e.g., OpenFlow)
    - • defines what is controllable and what is not
  + protocol for communicating with controller (e.g., OpenFlow)
  + 
* SDN Perspective: SDN Controller
  + SDN controller (network OS):
  + maintain network state information
  + interacts with network control applications “above” via northbound API
  + interacts with network switches “below” via southbound API
  + implemented as distributed system for performance, scalability, fault-tolerance, robustness
  + 
* SDN Perspective: Control Applications
  + network-control apps:
    - “brains” of control: implement control functions using lower-level services, API provided by SND controller
    - unbundled: can be provided by 3rd party: distinct from routing vendor, or SDN controller
  + 
* Components of SDN Controller:
  + 
* OpenFlow Protocol
  + operates between controller, switch
  + TCP used to exchange messages
    - • optional encryption
  + three classes of OpenFlow messages:
    - • controller-to-switch
    - • asynchronous (switch to controller)
    - • symmetric (misc)
* OpenFlow: Controller-to-Switch messages
  + Key controller-to-switch messages
  + features: controller queries switch features, switch replies
  + configure: controller queries/sets switch configuration parameters
  + modify-state: add, delete, modify flow entries in the OpenFlow tables
  + packet-out: controller can send this packet out of specific switch port
* OpenFlow; Switch-to-Controller messages
  + Key switch-to-controller messages
  + packet-in: transfer packet (and its control) to controller. See packetout message from controller
  + flow-removed: flow table entry deleted at switch
  + port status: inform controller of a change on a port.
* SDN: Control/Data Plane Interaction Example
  + S1, experiencing link failure using OpenFlow port status message to notify controller
  + SDN controller receives OpenFlow message, updates link status info
  + Dijkstra’s routing algorithm application has previously registered to be called when ever link status changes. It is called.
  + Dijkstra’s routing algorithm access network graph info, link state info in controller, computes new routes
  + link state routing app interacts with flow-table-computation component in SDN controller, which computes new flow tables needed
  + Controller uses OpenFlow to install new tables in switches that need updating
  + 
* OpenDaylight (ODL) Controller
  + ODL Lithium controller
  + network apps may be contained within, or be external to SDN controller
  + Service Abstraction Layer: interconnects internal, external applications and services
  + 
* ONOS Controller (Open Network Operating System)
  + control apps separate from controller
  + intent framework: high-level specification of service: what rather than how
  + considerable emphasis on distributed core: service reliability, replication performance scaling
  + 
* SDN: Selected Challenges
  + hardening the control plane: dependable, reliable, performance-scalable, secure distributed system
  + • robustness to failures: leverage strong theory of reliable distributed system for control plane
  + • dependability, security: “baked in” from day one?
  + networks, protocols meeting mission-specific requirements
  + • e.g., real-time, ultra-reliable, ultra-secure
  + Internet-scaling
* ICMP: Internet Control Message Protocol
  + used by hosts & routers to communicate networklevel information
    - error reporting: unreachable host, network, port, protocol
    - echo request/reply (used by ping)
  + network-layer “above” IP:
    - ICMP msgs carried in IP datagrams
  + ICMP message: type, code plus first 8 bytes of IP datagram causing error
  + 
* Traceroute and ICMP
  + Traceroute - tracing the path an IP packet takes across one or many networks
  + source sends series of UDP segments to destination
    - • first set has TTL =1
    - • second set has TTL=2, etc.
    - • unlikely port number
  + when datagram in nth set arrives to nth router:
    - • router discards datagram and sends source ICMP message (type 11, code 0)
    - • ICMP message include name of router & IP address
  + when ICMP message arrives, source records RTTs
  + stopping criteria:
    - UDP segment eventually arrives at destination host
    - destination returns ICMP “port unreachable” message (type 3, code 3)
    - source stops
* What is Network Management?
  + autonomous systems (aka “network”): 1000s of interacting hardware/software components
  + other complex systems requiring monitoring, control:
    - • jet airplane
    - • nuclear power plant
    - • others?
  + Network management includes the deployment, integration and coordination of the hardware, software, and human elements to monitor, test, poll, configure, analyze, evaluate, and control the network and element resources to meet the real-time, operational performance, and Quality of Service requirements at a reasonable cost.
* Infrastructure for Network Management
  + managed devices contain managed objects whose data is gathered into a Management Information Base (MIB)
  + 
* SNMP Protocol
  + Two ways to convey MIB info, commands:
    - 
* SNMP Protocol: messages types
  + 
* 