主要内容

- BGP协议
- 多协议标签交换 MPLS
- 路由器体系结构
- IP路由表查找算法（补充）
  - 最长前缀匹配算法（longest prefix match algorithms）
  - 包分类（Packet classification）算法
- 软件定义网络 SDN（Software-Defined Networking ）
  - SDN概述
  - SDN体系结构

多协议标签交换MPLS

Multiprotocol label switching

链路虚拟化

- 链路： 连接两台通信主机的物理线路
  - 直接连接
  - 互连
- 网络作为链路层
  - 例如：两台主机通过物理线路连接
    - 逻辑上分离的、全球性的电信网络
  - 覆盖网络 overlay network
- 多协议标签交换MPLS
  - 基于分组交换的虚电路网络
  - 为IP设备提供互连服务

问题提出

- 如何有效管理整个网络中端到端的流量？
- 如何对流量进行分类和路径设计？

Multiprotocol Label Switching (MPLS)

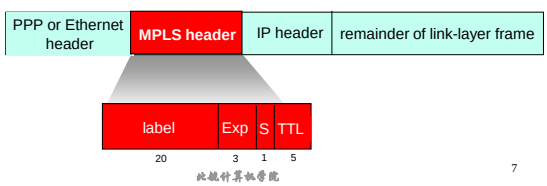
- 多协议标签交换（2001年，RFC3031，3032）
  - 将虚电路和数据报的特点相结合
  - 面向连接的方法，增加快速路由和服务质量
  - 思路：在每个分组前增加标记，根据标记进行路由
- 路由器技术的发展
  - Remove the need to provide both technologies (ATM and IP) in same network
- MPLS 功能
  - QoS support
  - 流量工程（Traffic engineering）
  - Virtual private networks(VPN)
  - Multiprotocol support

MPLS应用

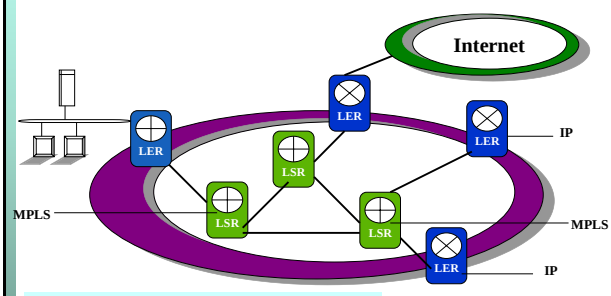
- 电信运营商的IP核心网络：事实标准
- 在广域网等场景已经得到了大量部署
  - ISP/OTT/大企业通过部署LDP、RSVP-TE等协议，为广域网业务提供VPN隔离以及差异化的流量调度方案

Multiprotocol label switching (MPLS)

- 初始目标: 使用固定长度标签（取代IP地址）进行高速IP转发
  - 使用固定长度标签快速查找转发表
  - 采用虚电路方法（Virtual Circuit，VC）
  - IP分组仍使用IP地址!



MPLS 网络模型

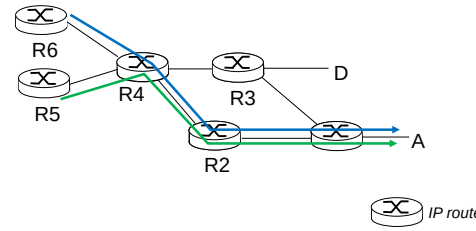


LSR = 标签交换路由器 Label Switched Router
LER = 标签边界路由器 Label Edge Router

支持MPLS的路由器

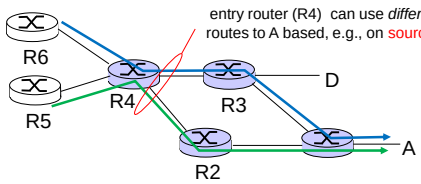
- 标签交换路由器 （LSR, label-switched router)
- 仅根据标签值将分组转发到输出端口
  - MPLS转发表和IP转发表不同
- 控制平面：需要建立信令协议
  - RSVP-TE（Resource Reservation Protocol - Traffic Engineering，基于流量工程扩展的资源预留协议）
  - LDP（Label Distribution Protocol，标签分发协议）
- 和传统支持IP的路由器共存
- *灵活性（flexibility）*：MPLS转发决策与IP转发不同
  - 流量工程：使用目的和源地址将流（flows）路由到各自的目的地址
  - 如果链路失败，快速对流进行重路由（re-route）：预先计算备份路径 (例如，用于VoIP)

MPLS vs. IP paths



- IP 路由:根据目的地址进行路由选择，决定分组转发路径

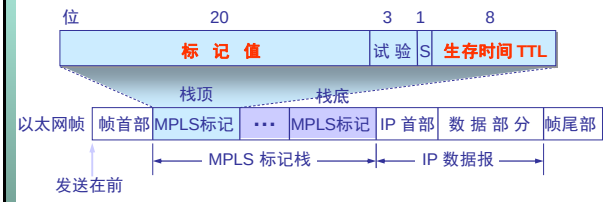
MPLS vs. IP paths



- MPLS 路由: 根据源和目的地址决定转发路径
  - fast reroute: precompute backup routes in case of link failure
  - 负载均衡

MPLS 标记栈

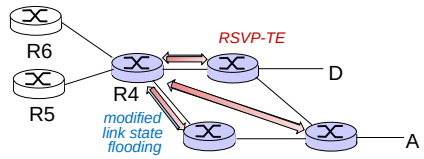
- MPLS 标记的格式以及标记栈：
  - 如何区分IP分组中是否有MPLS标记?
  - 标记栈的操作?
  - TTL字段的处理?



MPLS 信令（signaling）

- 修改链路状态协议（ OSPF, IS-IS）承载 MPLS路由
  - 如，链路带宽，“预留”链路带宽

entry MPLS router uses RSVP-TE signaling protocol to set up MPLS forwarding at downstream routers



MPLS 转发过程：初始化

- MPLS 域中的各 LSR 使用专门的标记分配协议 LDP 交换报文，建立LSP (Labelled switched path)，并构造分组转发表
- 设置QoS 参数
  - Resource commitment
  - Queuing and discard policy at LSR
- 需要的协议
  - 交换可达性和路由信息
    - Interior routing protocol e.g. OSPF used
  - 对转发等价类FEC分配标记
    - Local significance only
    - Manually or using Label distribution protocol (LDP) or enhanced version of RSVP

MPLS转发过程：分组处理

- 分组通过边界路由器LER进入MPLS域
  - Processed to determine QoS
  - LSR assigns packet to FEC and hence LSP
    - May need co-operation with other LSP to set up new LSP
  - Append label
  - Forward packet
- 域内LSR收到分组后
  - Remove incoming label, attach outgoing label and forward
- 出口边界路由器LER去掉标记，转发分组

下一跳标记转发条目NHLFE

- 在LSP上各个LSR都建立了输入/输入标记映射表，称为下一跳标记转发条目NHLFE
  - 处理过程：输入标记映射ILM（Incoming Label Map）
- NHLFE（Next Hop Label Forwarding Entry）包括下列信息
  - 分组的下一跳
  - 处理分组的标记栈，完成下列操作之一：
    - 标记替换：用新标记替换栈顶的标记
    - 出栈操作
    - 入栈操作：标记替换，并将一个或多个新标记压入标记栈
- 若分组的下一跳是LSR本身，必须执行“出栈操作”，并作转发决定
  - 该分组可能仍为一个标记分组或者是一个原始IP分组

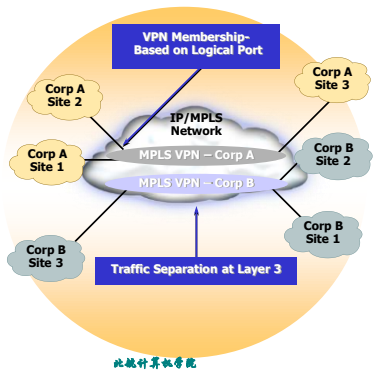
路由选择

- 如何为特定的FEC选择一个LSP?
- Hop-by-hop 逐跳路由
  - LSR independently chooses next hop
  - Ordinary routing protocols e.g. OSPF
  - Doesn't support traffic engineering or policy routing
- Explicit 显式路由选择
  - LSR (usually ingress or egress) specifies some or all LSRs in LSP for given FEC
  - Selected by configuration, or dynamically
  - *source routed*

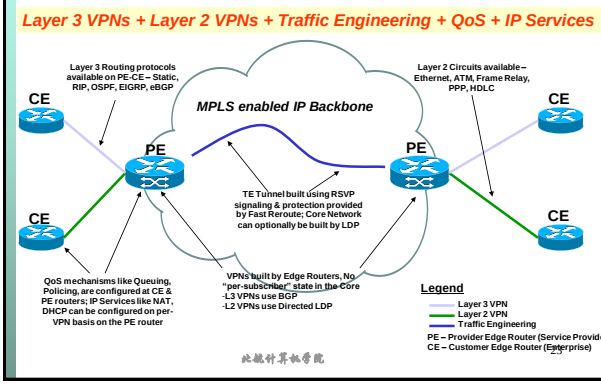
MPLS 流量工程

- MPLS Traffic Engineering (TE) provides high quality IP service.
- 流量工程定义了:
  - LSP Admission Control (LAC)
  - IP traffic (policing or shaping)
  - IP service prioritization
  - Network capacity and growth capacity

MPLS Layer 3 VPNs



MPLS Offerings Today – Putting It Together



路由器的体系结构

什么是路由器(Router)?

- 具有以下功能的计算机
  - Multiple interfaces
  - Implementing routing protocols
  - Packet forwarding
- 路由器的不同类型
  - Small device in a home network
  - Linux-based PC running router software
  - Million-dollar high-end routers with large chassis
- 链路特点
  - Serial line, Ethernet, WiFi, Packet-over-SONET, …

路由器图例

Access routers

Core router

Data center top-of-rack switch
数据中心机柜顶端交换机（ToR）

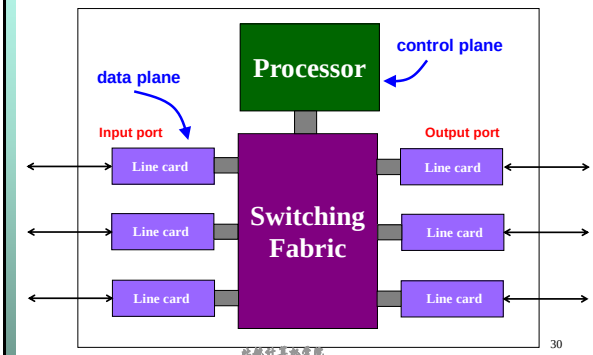
路由器的发展

- 早期
  - Workstation
  - Multiprocessor workstation
  - Line cards + shared bus
- 目前
  - Network controller
  - Line cards
  - Switched backplane

路由器的设计

- 不同类型路由器具有不同体系结构
- 高速路由器需要大量处理器
- 优化问题

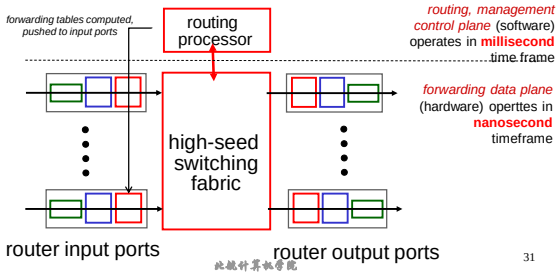
路由器体系结构



路由器体系结构

路由器的主要功能:

- 运行路由协议 (RIP, OSPF, BGP)
- *转发分组：forwarding* datagrams from incoming to outgoing link



路由器基本组成

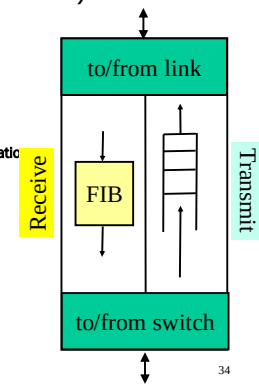
- 线卡（Line Cards）：对每个分组执行“快速”路径处理
  - Network interface cards，Provides parallel processing of packets
  - Forwarding lookup (hardware/ASIC vs. software)
- 网络处理器（Network Processor）
  - 运行路由协议，将转发表（forwarding table）下载到线卡上
    - 执行“慢速”路径处理
- 交换单元（switch）
  - N 个输入，M 个输出
    - 输出竞争：Multiple packets for same output
    - 交换竞争（Switch contention）：switch cannot support arbitrary set of transfers
  - 解决方案：缓存（buffer packets where needed）

数据平面（Data Plane）

- 处理分组：Streaming algorithms that act on packets
  - Matching on some bits, taking a simple action
  - 与控制平面和管理平面交互
- 功能
  - 转发Forwarding
  - 访问控制Access control
  - Mapping header fields
  - Traffic monitoring
  - Buffering and marking
  - Shaping and scheduling
  - Deep packet inspection

线卡（Line Cards）

- 网卡Network interface cards
- 连接：
  - Physical link，Switching fabric
- 并行包处理
  - Packet forwarding (Forwarding Information base，FIB)
  - Packet filtering (ACLs)
  - Buffer management
  - Link scheduling
  - Rate-limiting，Packet marking，Measurement
- 快速路径
  - Forwarding lookup (hardware/ASIC vs. software)



北航计算机学院

34

处理器（Route Processor）

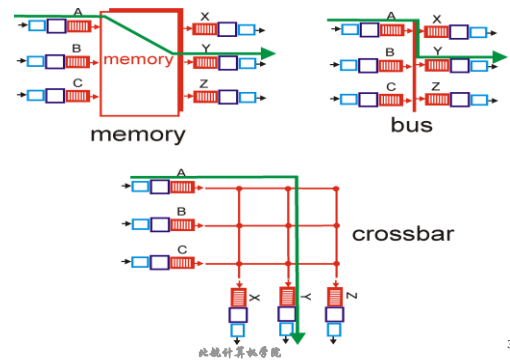
- 本地接口
  - So-called "Loopback" interface
  - IP address of the CPU on the router
- 控制层软件

  Control-plane software
  - Implementation of the routing protocols
  - Downloads forwarding table to the line cards
- 网络管理接口
  - Interface to network administrators
  - Command-line interface for configuration
  - Transmission of measurement statistics
- 慢速路径处理
  - Handles ICMP error messages
  - Handles IP option processing
  - Packets with expired Time-To-Live field（TTL）

北航计算机学院

35

三种交换结构(switching fabrics)



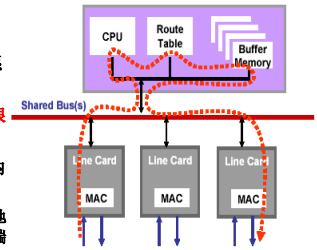
北航计算机学院

37

通过内存交换

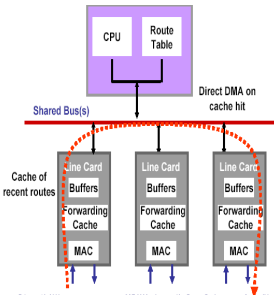
第一代路由器:

- 传统的计算机充当路由器：CPU控制I/O，分组被拷贝到系统内存
- 交换吞吐量受到内存带宽的限制（通常小于0.5Gbps）
- 现代路由器：通过共享系统内存进行交换
  - 输入线路上处理器执行目的地址查找，并缓存分组到输出端口的内存中
- 例：Cisco 的Catalyst 8500，Bay Networks Accelar 1200



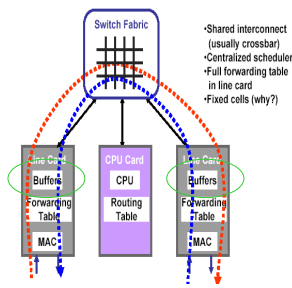
通过总线交换

- 通过共享总线将分组从输入端口内存传送到输出端口内存
  - 若总线忙，被阻塞，并在输入端口排队
- 总线竞争（bus contention）：交换速度受限于总线带宽。
- 32 Gbps bus, Cisco 5600
- 问题
  - 提高cache的命中率

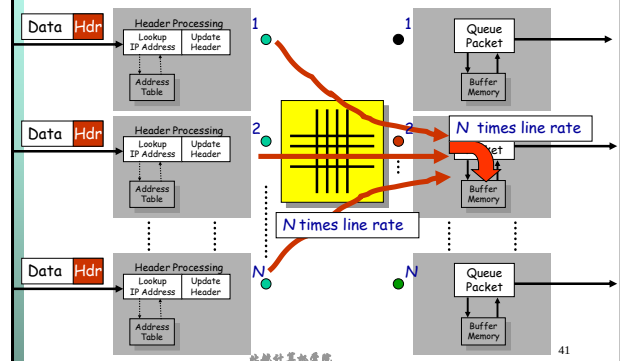


互连网络交换（Interconnection Network）

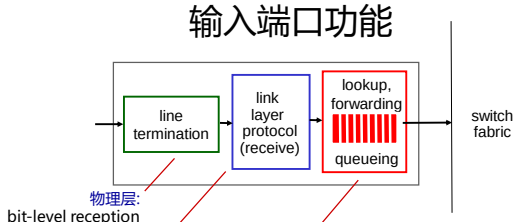
- 克服总线带宽限制
- 连接处理器
  - Crossbar provides full N×N interconnect，纵横式交换机
  - Banyan网络等
- 高级设计:
  - fragmenting datagram into fixed length cells, switch cells through the fabric.
  - Cisco 12000: switches 60 Gbps through the interconnection network



交换结构（Switch Fabric）：例子



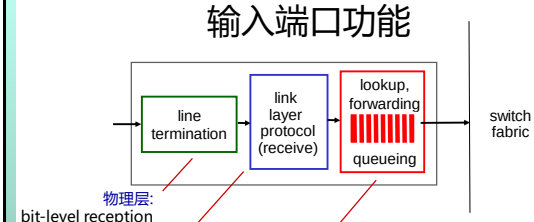
输入端口功能



decentralized switching:

- using header field values, lookup output port using forwarding table in input port memory ("match plus action")
- 目标: complete input port processing at 'line speed' 线速
- 排队: if datagrams arrive faster than forwarding rate into switch fabric

输入端口功能

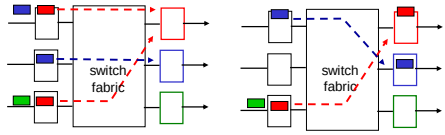


decentralized switching:

- using header field values, lookup output port using forwarding table in input port memory ("match plus action")
- 基于目的IP的转发: forward based only on destination IP address (traditional)
- 通用转发: generalized forwarding: forward based on any set of header field values (SDN)

输入端口排队

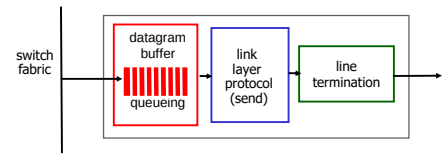
- Fabric slower than input ports combined
  - So, queuing may occur at input queues
- 队头阻塞 Head-of-the-Line (HOL) blocking
  - Queued packet at the front of the queue prevents others in queue from moving forward



输出端口竞争: 红色包
only one red datagram can be transferred.
lower red packet is blocked

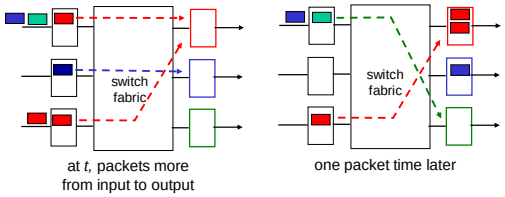
one packet time later:
green packet
experiences HOL
blocking

输出端口



- 需要缓存
  - 分组从交换结构到达速率大于输出端口线速率
- 排队规则
  - 优先级调度
- 拥塞控制

输出端口排队



at *t*, packets more from input to output

one packet time later

- 缓冲区溢出
  - 排队queueing (delay)
  - 丢包loss

交换缓存Switch Buffering

- 输出缓存（Output buffering）
  - Output may receive multiple packets per slot
  - Need speedup proportional to # inputs
- 输入缓存（Input buffering）
  - Which inputs are processed each slot – 调度schedule?
  - Head of line packets destined for busy output blocks other packets
- 内部缓存（Internal buffering）
  - Head of line blocking 队头阻塞：队列的首个分组由于它的目的端口正忙而被延迟转发,导致后面的分组被阻塞
  - 需要增加缓存

缓冲区设置折中: Design Trade-offs

- 增加输出队列长度
  - Pro: work-conserving, so maximizes throughput
  - Con: memory must operate at speed N*R
- 增加输入队列长度
  - Pro: memory can operate at speed R
  - Con: head-of-line blocking for access to output

工作保留模式 Work-conserving: output line is always busy when there is a packet in the switch for it

问题：如何设计路由器缓存的大小

- 目标：提高主干链路的利用率
- 计算缓存大小的方法【RFC3439】
  - 缓存B= RTT（平均往返时延，通常：250ms）×C（链路容量）

  例如, C = 10 Gpbs link: 2.5 Gbit buffer
- 大量TCP 流(N)经过一条链路，TCP非同步传输，所需缓存大小明显减少

  推荐： N代表拥塞链路中长TCP流的数目，需要缓存大小为：

$$\frac{RTT \cdot C}{\sqrt{N}}$$

- 可以降低缓存需求，同时保证良好的时延特性。

分组调度

- Scheduling discipline：在输出端口进行分组调度，从队列中选择一个分组进行传输
  - first-come-first-served (FCFS)，FIFO
  - weighted fair queuing (WFQ)
- 服务质量(QoS)的重要内容之一
  - Configuring packet classifiers
  - Configuring policers
  - Configuring link schedulers
- 类似地，在输入端口也存在缓冲区不足导致分组排队问题：丢弃到达分组
  - 弃尾 drop tail
  - Active queue management (AQM) → 拥塞控制
    - Random Early Detection (RED)

问题

- 路由器软硬件功能划分?
  - Trade-offs in speed vs. flexibility
- 可扩展性方面的限制?
  - Bit rate? Number of IP prefixes? # of line cards?
- 需要多少内存?
  - How much memory space should be available?
  - 是否越大越好?

路由表查找算法

（补充）

IP路由表查找

- 最长前缀匹配算法（longest prefix match algorithms）
- 包分类（Packet classification）算法
- 课外自学：
  - Linux的哈希查找算法
  - Linux的Trie树查找算法

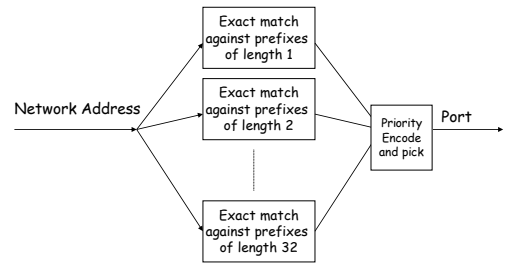
简单线性查找算法的问题

- 每次扫描转发表中的一个表项
  - See if the destination matches the entry
  - If so, check the size of the mask for the prefix
  - Keep track of the entry with longest-matching prefix
- 开销与转发表的大小呈线性关系
  - 200,000 entries!
  - And, the router may have just a few nanoseconds
  - ... before the next packet is arriving
- 提高效率：保持线速率（*line rate*）
  - 算法：Better algorithms
  - 硬件：Hardware implementations

54

最长前缀匹配算法

- 精确匹配：Use 32 exact match algorithms



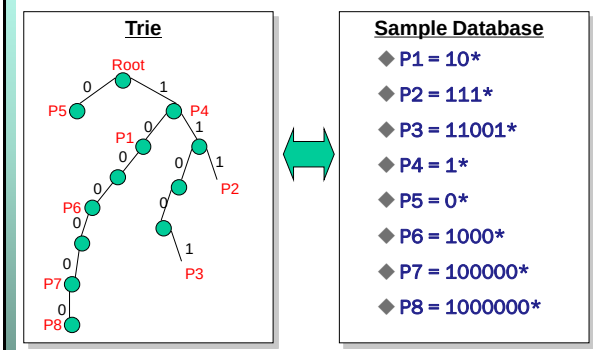
55

查询算法度量

- 速度 (= number of memory accesses)
- 缓存需求 (= amount of memory)
- 最小更新时间 (support >10K updates/s)
- 可扩展性 Scalability
  - With length of prefix: IPv4 unicast (32b), Ethernet (48b), IPv4 multicast (64b), IPv6 unicast (128b)
  - With size of routing table: (sweetspot for today's designs = 1 million)
- 灵活性 Flexibility in implementation
- 处理时间 Low preprocessing time

56

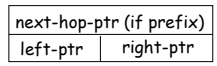
Trie Using Sample Database



57

- W-bit prefixes: O(W) lookup, O(NW) storage and O(W) update complexity
- Advantages
  - Simplicity
  - Extensible to wider fields
- Disadvantages
  - Worst case lookup slow
  - Wastage of storage space in chains

Trie node



Patricia Tree

- 采用树结构存储前缀，压缩路径
  - One bit for each level of the tree
  - Some nodes correspond to valid prefixes
  - ... which have next-hop interfaces in a table
- 当一个分组到达时：
  - Traverse the tree based on the destination address
  - Stop upon reaching the longest matching prefix

其他算法

- 基于多分支 Trie 树(Multibit Trie)的查找算法
  - 在基于二进制 Trie 树和路径压缩 Trie 树的查找中，访问一次存储器，只比较 1个比特位。
  - 如果增加每次访问存储器时所比较的比特位数，就可以减少访问存储器的次数。将步宽大于 1 的 Trie 树称为多分支 Trie 树
- 基于地址区间的二分查找算法
  - 在目的 IP 地址所属的地址区间中，找到范围最小的区间
- 基于硬件的查找算法
  - 使用较多的硬件是内容寻址存储器（Content Addressable Memory，CAM），三态内容可寻址存储器（Ternary Content Addressable Memory，TCAM））

三态内容可寻址存储器TCAM

- 在 TCAM 中，每个比特位有3 种状态，分别表示 0、1、X，当比特位为“X”时，表示该比特位无需比较。
- 这种存储方式十分适合存储地址前缀，因为在查找过程中，只需要比较地址前缀的最开始的位串，其他位串不需要比较，用“X”表示。
- 不需要进行任何扩展，任意长度的地址前缀在 TCAM 中可以直接存储其值和长度。

最长前缀匹配方法实现

- 高端路由设备上采TCAM(TernaryContent-AddressableMemo-ry)，三态内容可寻址存储器
- 根据内容寻址
- TCAM 中每一个表项都是以<地址、掩码>序偶的形式保存
- 在表项的匹配查找中，TCAM 判断(查找关键字“按位与”掩码)是否与(目的地址“按位与”掩码)相等，如果相等表示关键字与该表项匹配；否则不匹配。
- 在一个时钟周期内容进行查找，不考虑表大小
- 可以级联多个TCAM增加查找表容量
- Cisco Catalyst: TCAM中路由表项约 ~1M

62

TCAM查找结构

- 在TCAM 的低地址存储前缀较长的关键字表项，在地址高的区域存储前缀较短的关键字表项。
- 不足
  - 价格昂贵
  - 功耗大
  - 表项更新复杂

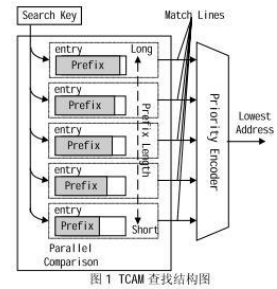


图 1 TCAM 查找结构图

63

包分类 Packet Classification

- 典型应用
  - Identify flows for QoS
  - Firewall filtering
- 需求
  - Match on multiple fields
  - Strict priority among rules
    - E.g 1. no traffic from 128.2.*
      2. ok traffic on port 80

64

复杂度

- 规则数N，数据包头部域个数 k（ k > 2）
  - 时间复杂度O(log N^{k-1})，空间复杂度 O(N)
  - 时间复杂度O(log N)，空间复杂度O(N^k)
  - 特例：k = 2 → source and destination
    - 时间复杂度O(log N)，空间复杂度 O(N)
- 规则数?
  - Largest for firewalls & similar → 1700
  - Diffserv/QoS → much larger → 100k (?)

65

路由查找算法的评价标准

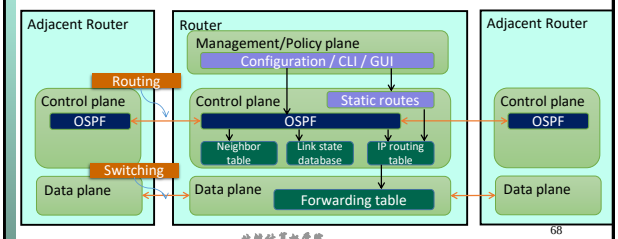
- 查找速度
  - 软件环境下实现，主要由存储器访问次数决定
- 存储容量
- 预处理和更新速度
  - 动态算法
  - 静态算法
- 算法实现的灵活性
  - 软件实现
  - 硬件实现
- 算法的可扩展性
  - IPv4和IPv6

软件定义网络SDN

数据平面和OpenFlow

传统路由器设备

- 传统路由器的体系结构
  - 管理平面 Management plane/ configuration
  - 控制平面 Control plane / Decision: 如OSPF (Open Shortest Path First)
  - 数据平面 Data plane / Forwarding



The Networking "Planes"

- 数据平面Data plane: processing and delivery of packets with local forwarding state
  - Forwarding state + packet header → forwarding decision
  - Filtering, buffering, scheduling
- 控制平面Control plane: computing the forwarding state in routers
  - Determines how and where packets are forwarded
  - Routing, traffic engineering, failure detection/recovery, …
- 管理平面Management plane: configuring and tuning the network
  - Traffic engineering, ACL config, device provisioning, …

网络管理的现状

- 复杂网络环境
  - 网络分层结构：Access接入网, backbone 主干网
  - 数据中心网络Data-center networks, 企业/校园网 enterprise/campus
  - 网络规模： 10-10,000 routers/switches
- 多种网络技术
  - IP转发Longest-prefix routing (IP), label switching (MPLS, ATM), circuit switching (optical, TDM) ,WIFI
- 多种策略
  - Routing, reachability, transit, traffic engineering, robustness
- 管理、控制和转发设备紧耦合

  The control plane software binds these elements together and defines the network

传统网络的挑战

- 难以监控
  - The network is Hard to reason about；Hard to evolve；Expensive
- 难以管理
  - (Too) many task-specific control mechanisms
    - No modularity, limited functionality
  - 间接控制 Indirect control
    - Must invert protocol behavior, "coax" it to do what you want
    - Ex. Changing weights instead of paths for TE
  - 无法协同控制，无法控制路由器的更新
- 复杂性
  - 协议之间相互作用
    - Routing, addressing, access control, QoS

SDN的发展

- 1990s，主动网络Active Networking：存储-计算-转发
- 2006:斯坦福大学的博士生Martin Casado及其团队提出了 clean-slate security architecture (SANE)，定义了集中式控制器管理网络中的安全控制策略；2007年，Ethane 在此基础上进行扩展
  - 4D项目[Princeton, CMU]，SANE/Ethane项目[Stanford/Berkeley]
  - SANE/Ethane项目采用Openflow
- 2008年，斯坦福大学大学教授Nick McKeown 等人提出了OpenFlow 的概念，并于当年在ACM SIGCOMM 发表了题为《OpenFlow: Enabling Innovation in Campus Networks》的论文
  - OpenFlow switch interface [Stanford]，发布第一个开源SDN控制器NOX Network OS [Nicira]
- 2009年12月，OpenFlow规范发布了1.0版本

SDN的发展

- 2009年6月: Martin Casado 成立 Nicira公司
- 2011年3月:开放网络基金会（Open Networking Foundation，ONF）成立
- 2011年10月: 首次开放网络峰会（Open Networking Summit，ONS）
  - 很多企业加入 (Juniper, Cisco等)
  - 在网络行业里产生巨大影响
- 2012 ONF 会议，Google发布了第一个SDN规模应用案例B4
  - Google's G-Scale network is operating using OpenFlow
  - Developed for 2 years (2010~2012.1)
  - Saved CAPEX资本性支出 and OPEX 运营成本
- 2012年7月: VMware 12.6亿美元收购Nicira

SDN的相关机构

- Open Networking Foundation（开放网络基金会），ONF
  - 2011年，由德电、Facebook、Google、微软、Verizon、Yahoo!发起成立了Open Networking Foundation，旨在推进实现基于OpenFlow的开放网络，2012年4月，ONF发布了SDN白皮书（Software Defined Networking：The New Norm for Networks）
- Network Functions Virtualization， NFV
  - 2012年底，AT&T、英国电信(BT)、德国电信、Orange、意大利电信、西班牙电信公司和Verizon联合发起成立了网络功能虚拟化产业联盟(Network Functions Virtualisation，NFV)，旨在将SDN的理念引入电信业
- OpenDaylight
  - 2013年4月，思科和IBM联合微软、惠普、红帽和VMware等18家公司发起成立了Open Daylight，与LINUX基金会合作，开发SDN控制器、南向/北向API等软件，旨在打破大厂商对网络硬件的垄断

SDN发展的主要事件

SDN实验方法

- 利用Mininet网络仿真平台建立OpenFlow实验环境，了解mininet和SDN的基本工作原理
  1、安装以下工具：Virtualbox, Xming, Mininet VM
  2、参考 Mininet Walkthrough (http://mininet.org/walkthrough）
  3、Mininet实验平台分析，定义网络拓扑结构，配置openflow规则。
  - Implement SDN architecture
  - Add/Remove flows
  - Enable/Disable communication
  - Verify Flow table
  - Get statistics/Info northbound API