第三章

交换机端口配置与生成树协议配置

- 交换机端口配置
- 生成树协议介绍
- 生成树协议配置

交换机端口配置

- 端口速率
- 端口工作模式
- 端口类型
- 端口流量控制
- 端口聚合
- 端口镜像

端口速率配置

- 交换机端口所支持的速率
  - 标准以太网 —— 10Mbps
  - 快速以太网 —— 100Mbps
  - 千兆以太网 —— 1000Mbps
- 自协商的结果

	标准以太网	快速以太网	千兆以太网
标准以太网	10M	10M	10M
快速以太网	10M	100M	100M
千兆以太网	10M	100M	1000M

- 端口速率配置命令（端口为三元组）

[H3C-GigabitEthernet1/0/1] speed { 10 | 100 | 1000 | auto }

端口工作模式配置

- 交换机端口所支持的工作模式
  - 全双工（Full-duplex）
    - 连接计算机时
    - 不使用 CSMA/CD
  - 半双工（Half-duplex）
    - 连接 Hub时
    - 使用 CSMA/CD

- 端口工作模式配置命令：
- [H3C-GigabitEthernet1/0/1] duplex { full | half | auto}

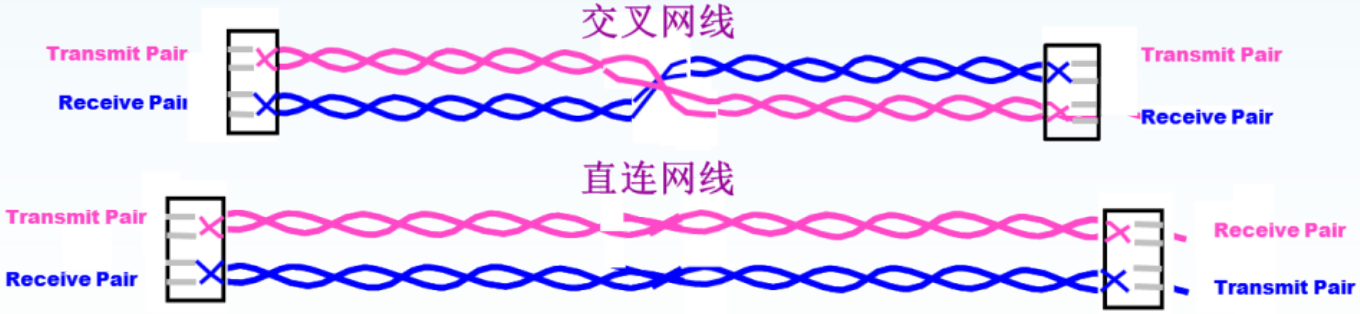
端口类型配置

- 端口类型
  - MDI（Medium Dependent Interface，介质相关接口）
  - MDI-X 或 MII（Medium Independent Interface，介质无关接口） MDI MDI-X

发	1	1	收
发	2	2	收
收	3	3	发
收	6	6	发

端口类型配置（续）

- 路由器和PC机一般都使用MDI接口，以太网交换机一般都使用MDI-X接口
- 同端口类型相连使用交叉网线，异端口类型相连使用直连网线。



端口类型配置（续）

- H3C交换机可以智能识别网线类型和对端MDI / MDI-X 端口类型

- 端口类型配置命令
  - [H3C-GigabitEthernet1/0/1] mdix-mode { normal | cross | automdix }
    - normal ： MDI-X 端口
    - cross ： MDI 端口
    - automdix ： 自适应

流量控制配置

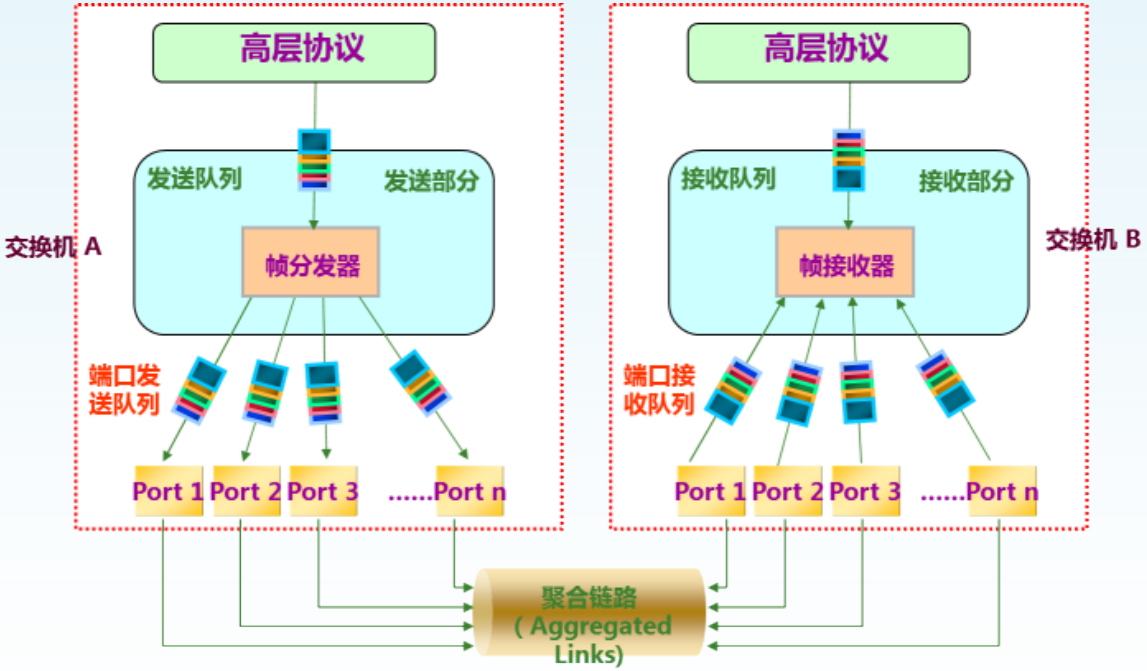
- 流量控制目标
  - 减轻或避免大量以太网帧在交换机端口发生拥塞
- 流量控制原理
  - Half-duplex：使用后退压力（Backpressure）技术，即模拟产生冲突信号（Jam Signal）
  - Full-duplex：向对端设备发送PAUSE帧
- 流量控制配置命令
  - [H3C-GigabitEthernet1/0/1] flow-control
  - [H3C-GigabitEthernet1/0/1] undo flow-control

注：H3C系列交换机所有端口在缺省情况下都禁用了流量控制功能

端口聚合配置 — 概述

- 端口聚合（Port Aggregation），也称为端口捆绑或链路聚合（Link Aggregation）。
- 指两个交换机之间通过两个或多个端口并行连接，以获得更高的带宽。
- 端口聚合是目前很多品牌交换机都支持的一种高级特性。

端口聚合配置 — 实现原理



为了保证帧的按序传送，必须将同一会话的帧分配到同一端口进行发送

端口聚合配置 — 相关命令

- 静态聚合命令

  创建聚合接口，并进入聚合接口视图

  [h3c]interface bridge-aggregation *interface-number*

  - *interface-number*：聚合端口， 系列取值范围为1-1024（不同版本数值有差别，见配置手册）

  将以太网接口加入聚合组（首先进入以太网接口视图）

  – [H3C-GigabitEthernet1/0/1] port link-aggregation group *number（注：此处number与聚合端口的数值一致）*

- 例：将以太网端口GigabitEthernet1/0/1 加入聚合端口22。
  – [H3C] interface bridge-aggregation 22
  – [H3C] interface GigabitEthernet1/0/1
  – [H3C-GigabitEthernet1/0/1] port link-aggregation group 22

端口聚合配置 — 相关命令（续）

- 清除端口聚合(删除聚合端口)
  - [H3C-Bridge-Aggregation1] shutdown
  - [H3C-Bridge-Aggregation1]quit
  - [H3C]undo interface Bridge-Aggregation 1（ 如不能执行，请输入undo link-aggregation group *agg-id* ）
- 显示端口聚合的信息
  - [H3C] display link-aggregation summary

端口镜像配置 — 概述

- 镜像分为两种：端口镜像和流镜像。
- 端口镜像是指将某些指定端口（出或入方向）的数据流量映射到监控端口，以便集中使用数据捕获软件进行数据分析。
- 流镜像是指按照一定的数据流分类规则对数据进行分流，然后将属于指定流的所有数据映射到监控端口，以便进行数据分析。

端口镜像配置 — 相关命令

- 创建本地镜像组
  - [H3C] mirroring-group *group-id* local
- 为本地镜像组配置源端口（被镜像端口）
  - mirroring-group *group-id* mirroring-port *interface-list* { both | inbound | outbound }
  - **inbound** 表示仅对本端口接收的报文进行监控；**outbound** 表示仅对本端口发送的报文进行；监控**both** 表示同时对本端口接收和发送的报文进行监控
- 配置目的端口（镜像端口）
  - [H3C] mirroring-group *group-id* monitor-port *interface-type interface-number*
  - 注意：目的端口上要关闭生成树协议，端口视图下执行“undo stp enable”。

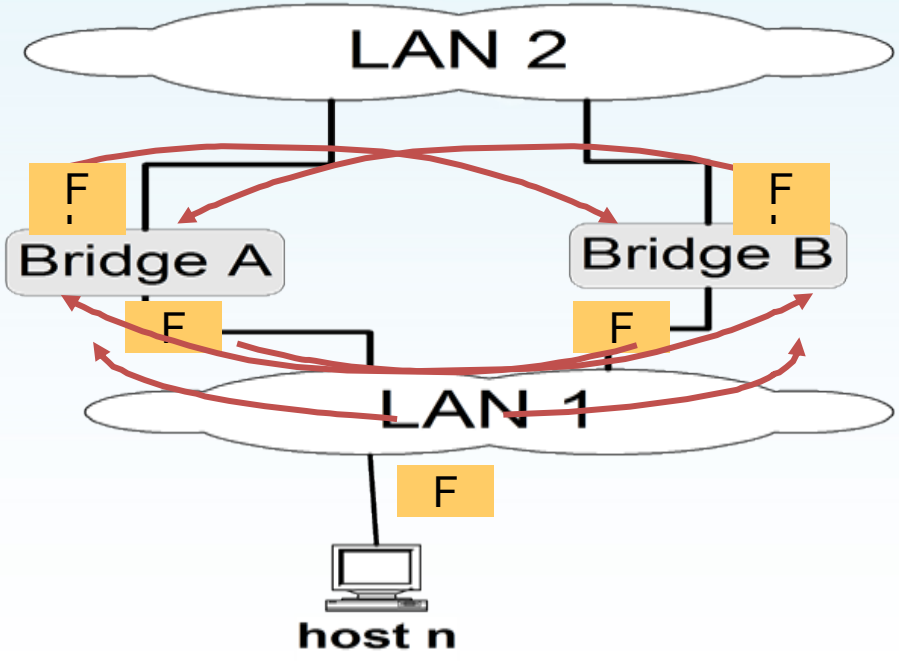
生成树协议（Spanning Tree Protocol,STP）

- 起因和历史
- 概述
- 术语
- BPDU消息与消息交换
- 生成树的构造
- 总结

生成树协议 — 起因

Loop的危险：

- 如右图，考虑被两个Bridges连接起来的两个局域网
- 假设主机n要发送帧F, 并且两个Bridges的MAC地址表中都没有包含F目的地址的表项



生成树协议 — 历史

- In 1980s, Radia Perlman发明了生成树协议来避免在局域网中产生环。
- Radia Perlman 于1988年在MIT获得计算机博士学位。
- 她目前工作于Sun Microsystems, Inc.
- 她有一本很有名的书：《Interconnections, Second Edition: Bridges, Routers, Switches, and Internetworking Protocols》

Algorhyme - Peom of spanning tree algorithm

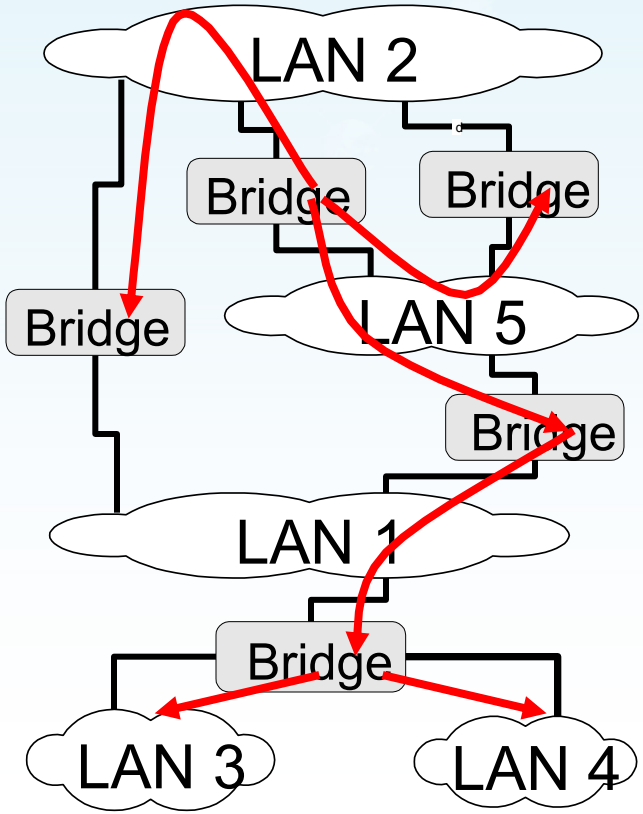
I think that I shall never see,
a graph more lovely than a tree.
A tree whose crucial property,
is loop-free connectivity.
A tree that must be sure to span,
so packet can reach every LAN.
First, the root must be selected.
By ID, it is elected.
Least-cost paths from root are traced.
In the tree, these paths are placed.
A mesh is made by folks like me,
then bridges find a spanning tree.

--- By Radia Perlman

生成树协议 — 概述

- 基本思想：
  - 生成树没有环。
  - Bridge之间通过不断地交换控制帧来动态的构造生成树。

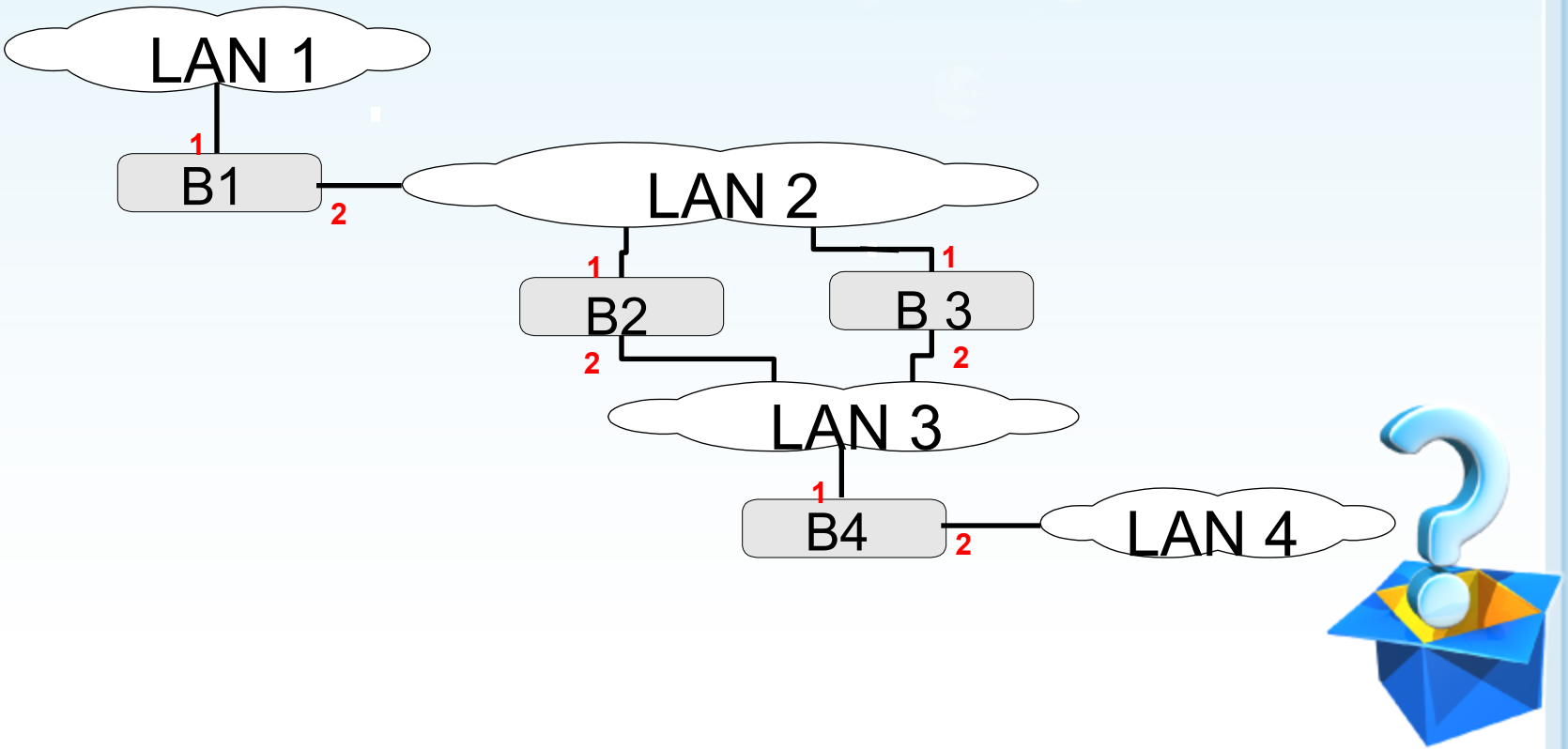
- 这个控制帧被称为：Configuration Bridge Protocol Data Unit (Configuration BPDU)



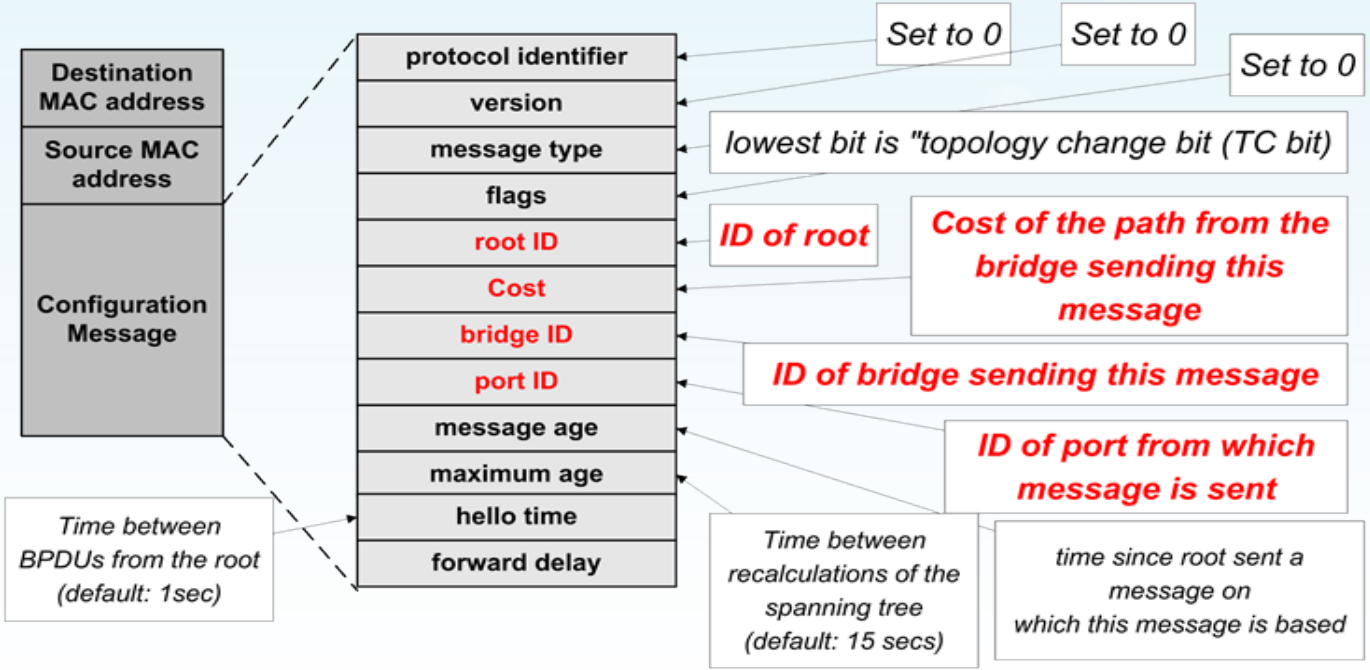
生成树协议 — 术语

- Bridge ID: 每个Bridge的唯一标识（Priority + MAC Address）。
- Port ID: Bridge上每个端口的唯一标识。（Port Priority + Port Index）
- Root Bridge: 具有最小Bridge ID的Bridge。在生成树协议中，将把这个Bridge当作是生成树的Root.
- Root Path Cost: 到达Root Bridge的最短路径的长度，单位一般为hop数。
- Root Port: 到达Root Bridge的最短路径的出发端口。
- Designated Bridge: 对于一个LAN而言，通往Root Bridge最短路径上的所经由的第一个Bridge。如果两个Bridge有相同长度的最短路径，那么取Bridge ID较小的那一个。
- Designated Port: 如果Bridge B是LAN L的Designated Bridge, 那么Bridge B与LAN L相连的端口就称为Bridge B对于LAN L的Designated Port.

生成树协议 — 概述（举例）



生成树协议 — BPDU格式



生成树协议 — BPDU的交换

- 每个Bridge都周期性的向与自己相连的LAN上发送如下的BPDU：

root bridge (what the sender thinks it is) root path cost for sending bridge
Identifies sending bridge
Identifies the sending port

生成树协议 — “better”关系

- 给定两个 BPDU M1与M2：

ID R1	C1	ID B1	ID P1

M1

ID R2	C2	ID B2	ID P2

M2

- 我们说 M1 is better than M2, if

(R1 < R2),

Or (R1 == R2) and (C1 < C2),

Or (R1 == R2) and (C1 == C2) and (B1 < B2),

Or (R1 == R2) and (C1 == C2) and (B1 == B2)

and (P1 < P2)

生成树协议 – 初始化

- 在协议运行之初，每个Bridge都认为自己是Root Bridge.
- 于是每个Bridge都向与它相连的LAN上发送如下形式的BPDU:

B	0	B	P

生成树协议 — 更新BPDU

- 每个Bridge把它所收到的所有BPDU和它自己所发送的BPDU相比较.
- 若一个发送如下BPDU M1的Bridge B1

M1

R1	C1	B1	P1

收到一个Better BPDU M2，满足 R2 < R1:

M2

R2	C2	B2	P2

那么Bridge B1 就把自己的 BPDU 更新为:

R2	C2+1	B1	P1

生成树协议 – 更新BPDU（续）

- 若一个发送如下BPDU M1的Bridge B1

M1

R1	C1	B1	P1

收到一个Better BPDU M2，满足C2 <= C1 – 2 :

M2

R1	C2	B2	P2

那么Bridge B1 就把自己的 BPDU 更新为:

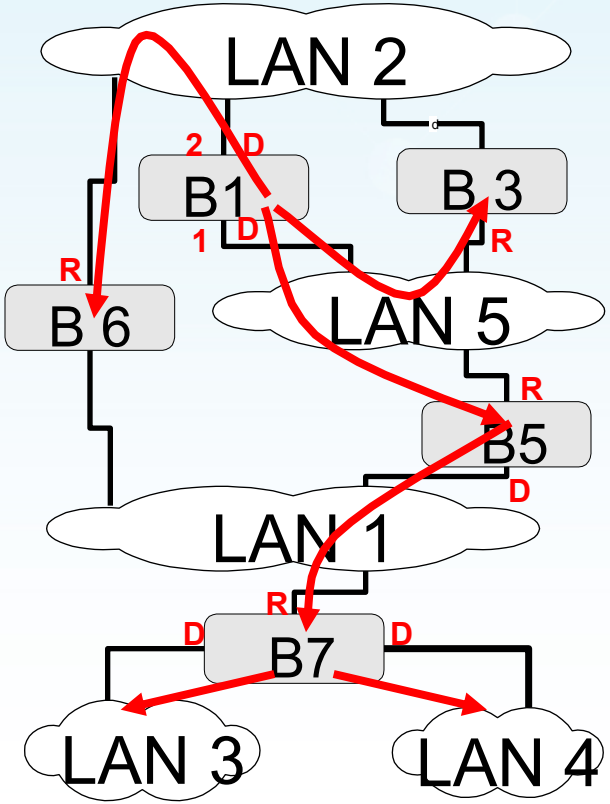
R1	C2+1	B1	P1

生成树协议 — 树的构造

- 对于每一个Bridge：
  - 收到 Better BPDU 的 并且到达Root Bridge路径最短的那个 Port 被认为是该Bridge的 Root Port. (注：如果该Bridge自己的 BPDU为 Best，则无 Root Port.)
  - 对于该Bridge上的一个Port X所连接的LAN L, 如果该Bridge 自己的BPDU比所有从Port X收到的其它的BPDU都好，那么该 Bridge 就认为自己是LAN L的Designated Bridge, 并且认为 Port X是自己对于LAN L的Designated Port.
- 在决定了Root Port和Designated Port后，每个Bridge都认为：
  - 它的Root Port在生成树上
  - 它的所有Designated Ports在生成树上。
  - 它的所有其它Port都不在生成树上。

生成树协议 — 树的构造（续）

- 如右图，每个Bridge都计算出了自己的Root Port和Designated Port
- 作由Designated Port到Root Port的连线和由Designated Port到LAN的连线，就得到了局域网上的生成树。



生成树协议 – 帧转发规则

- 每个Bridge只接受从 Root Port 或 Designated Port 收到的数据帧。
- 每个Bridge只在 Root Port 或者 Designated Port 上转发数据帧
- 这样就避免了环路的出现。

生成树协议 — 端口状态

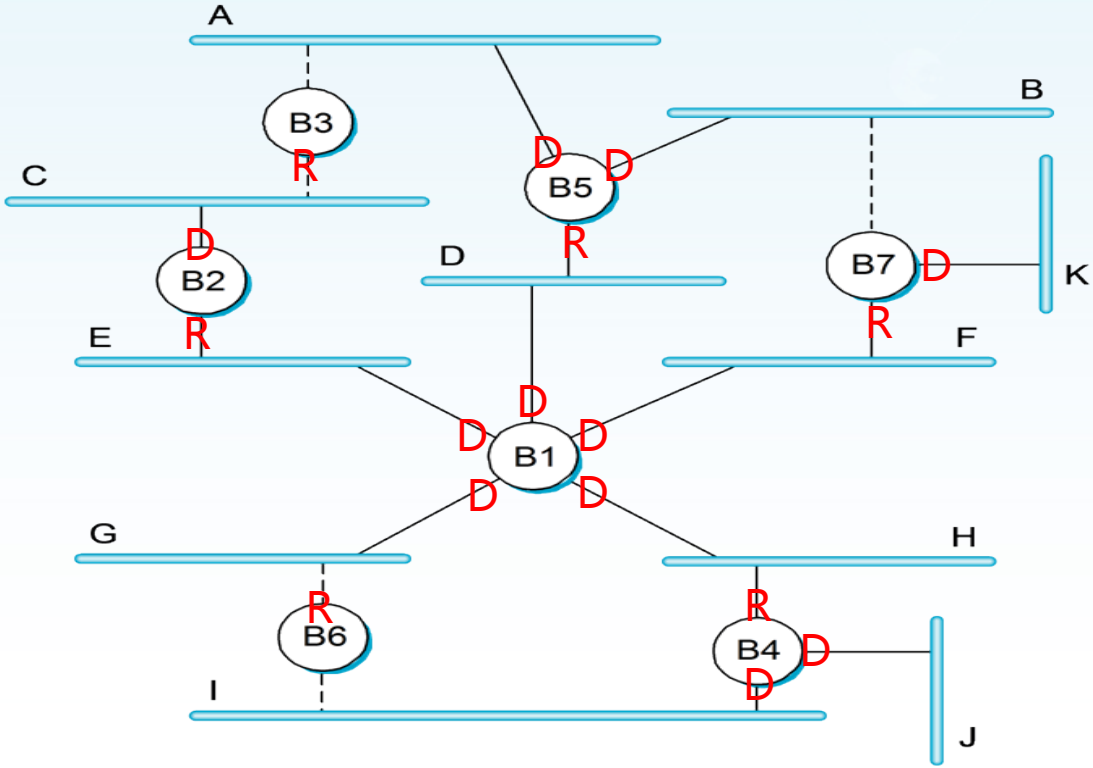
IETF RFC 1493

IEEE 802.1D

端口状态	端口能力	
Disabled	不收发任何报文	
Blocking	不接收或转发数据，接收但不发送BPDUs，不进行地址学习	Discarding
Listening	不接收或转发数据，接收并发送BPDUs，不进行地址学习	
Learning	不接收或转发数据，接收并发送BPDUs，进行地址学习	Learning
Forwarding	接收并转发数据，接收并发送BPDUs，进行地址学习	Forwarding

生成树协议 — 举例

注：某些Bridge可能被闲置，例如B3 和 B6。



生成树协议 — 演变

- 生成树协议（STP）
  - 1990年，Radia Perlman的STP被IEEE标准化为 IEEE Std 802.1D
  - 缺点：需要等待计时器超时，网络恢复连通速度慢
  - 消失于IEEE Std 802.1D 2004 Edition
- 快速生成树协议（RSTP, Rapid STP）
  - 出现于IEEE Std 802.1D 2001 Edition
  - 优点：无需等待计时器超时，增加了用于主动通知的TCN（Topology Change Notification）消息，完全兼容STP
  - 目前的交换机一般都实现RSTP
- 多生成树协议（MSTP, Multiple STP）
  - 出现于IEEE Std 802.1s 2002 Editionzai
  - 用于VLAN的生成树协议
  - 在S3900系列和S3610交换机上实现的是MSTP

STP, RSTP, MSTP的基本思想和过程相同，都遵照了当初Radia Perlman的想法

生成树协议 – 总结

计算机网络领域最优美的协议之一

- 动态算法
  - 能适应网络拓扑结构的改变
- 分布式算法
  - 每个Bridge独立的作出自己对Root Port和Designated Port的决定。
- 消息交换简单
  - 周期性的相邻Bridge之间BPDU交换
- 消息处理简单
  - 仅决定“better”关系即可

生成树协议配置

- 启动/关闭生成树协议
- 设定网桥的优先级
- 设定端口的优先级
- 设定端口的开销
- 设定端口的Forward Delay
- 显示生成树协议信息

启动/关闭生成树协议（V5系统）

- 在H3C系列交换机（Comware V5系统）中，生成树协议缺省为关闭状态。可以用下述命令来改变生成树的状态：
  - stp { enable | disable }
  - undo stp
- 适用视图
  - 系统视图、以太网端口视图
- 参数
  - **enable**：用来开启设备或端口的STP。
  - **disable**：用来关闭设备或端口的STP。
- stp { enable | disable } 命令用来开启/关闭设备或端口上的STP。
- 当在系统视图下用来配置设备STP 时，undo stp 命令用来恢复设备的STP为缺省状态；在以太网端口视图下使用undo stp 命令，配置的效果和系统视图下一样，也是将设备的STP 恢复为缺省状态。

启动/关闭生成树协议（V7系统）

- 在5820V2系列交换机中，出厂配置启动时，全局生成树协议处于开启状态。可以用下述命令来改变生成树的状态：
  - stp global enable
  - undo stp global enable
  - 适用视图-系统视图
- 在以太网端口视图下使能或关闭生成树协议
  - stp enable
  - undo stp enable

设定网桥的优先级

- 网桥ID由两部分组成:

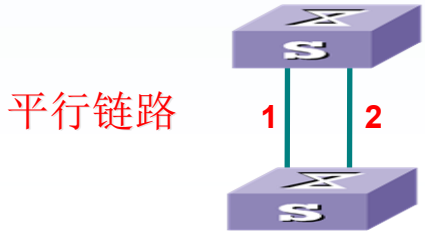
  Bridge Priority + Bridge MacAddress

- 如果网络中的所有交换机都在缺省配置下，根据BPDU比较原则，MAC地址最小的交换机被选为根桥，但是该交换机未必是理想的根桥，可以通过命令配置Bridge Priority将合适的交换机推举为根桥
  - [H3C] stp priority *bridge-priority*
    - *bridge-priority*：用来标识所设定的bridge 优先级，该值是不连续的，范围为0～61440，步长为4096。缺省情况下，交换机的优先级为32768。
  - [H3C] undo stp priority　// 恢复为缺省值

设定端口的优先级

- 根据BPDU比较原则，有时候需要比较端口ID
- 端口ID由两部分组成:
  - Port Priority + Port Index
  - 其中端口优先级部分是可配置的
- 命令格式为
  - [H3C-GigabitEthernet1/0/1] stp port priority *port-priority*

  port-priority：用来标识所设定的优先级，该值是不连续的，范围为0～240，步长为16。缺省情况下，端口优先级为128。
  - [H3C-GigabitEthernet1/0/1] undo stp port priority



设定端口的开销（legacy标准）

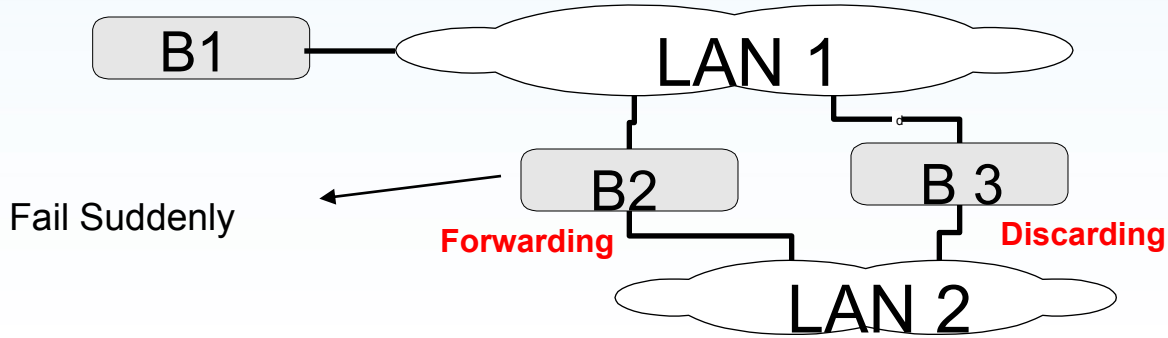
- 从本网桥到根桥的路径上所有经过端口的端口开销之和为"根路径开销"，可以通过命令来改变端口开销的值
  - [H3C-GigabitEthernet1/0/1] stp cost *cost*
    - *cost*：用来标识所设定的路径开销值，范围1～200000。缺省情况下，网桥根据与端口相连的链路速率而直接得到端口的路径开销。
  - [H3C-GigabitEthernet1/0/1] undo stp cost

表1-2 端口路径开销与链路速率对照表

链路速率	推荐值	推荐取值范围	值域
10Mbit/s	2000	200～20000	1～200000
100Mbit/s	200	20～2000	1～200000
1Gbit/s	20	2～200	1～200000
10Gbit/s	2	2～20	1～200000

设定端口的Forward Delay

- Forward Delay：端口转换为 Forwarding 状态所需等待的时间
  - 过长的Forward Delay会导致生成树的收敛太慢；
  - 过短的Forward Delay可能会在拓扑改变的时候，引入暂时的环路。（原来的主链路端口尚未阻塞，备份链路即开始转发）



设定端口的Forward Delay（续）

- [H3C] stp timer forward-delay *centiseconds*
  - *centiseconds:* 厘秒，缺省为1500
- [H3C] undo stp timer forward-delay

Table 17-1—RSTP Timer and Transmit Hold Count parameter values

Parameter	Recommended or Default value	Permitted Range	Compatibility Range
Migrate Time (17.13.9)	3.0	—[a]	—[a]
Bridge Hello Time (17.13.6)	2.0	—[a]	1.0–2.0
Bridge Max Age (17.13.8)	20.0	6.0–40.0	6.0–40.0
Bridge Forward Delay (17.13.5)	15.0	4.0–30.0	4.0–30.0
Transmit Hold Count (17.13.12)	6	1–10	1–10

All times are in seconds. —[1] Not applicable, value is fixed.

显示生成树协议信息

- [任意视图] display stp [interface *interface_list*]
  - interface *interface_list*：以太网端口列表，表示多个以太网端口，表示方式为

    interface _list = { *interface_type interface_num* [**to** *interface_type interface_num*] } & <1-10>

    “&<1-10>” 表示前面的参数最多可以输入10 次。
- 本命令用来显示当前STP 的状态信息或统计信息。根据该命令的输出信息，可以帮助用户确认STP 配置是否正确。
- 【例】显示设备上以太网端口Ethernet1/0/1 的STP 信息。
  - <H3C> display stp interface Ethernet1/0/1

做实验注意事项

- 做实验前，请在用户视图下使用“reset saved-configuration”命令和“reboot”命令将设备的配置清空，以免前一个实验留下的配置对本次实验产生影响。
- 请关闭PC机上的防火墙
- 使用交换机做STP实验时，请首先在系统视图下执行以下命令：

  stp mode rstp

  stp pathcost-standard legacy
- 故障诊断：
  - 检查接线是否正确
  - 检查指示灯是否亮（PC机网卡灯，交换机的端口灯）
  - 检查配置是否正确（PC机：ipconifg, 交换机：display）