第 5 篇 配置 IP 路由

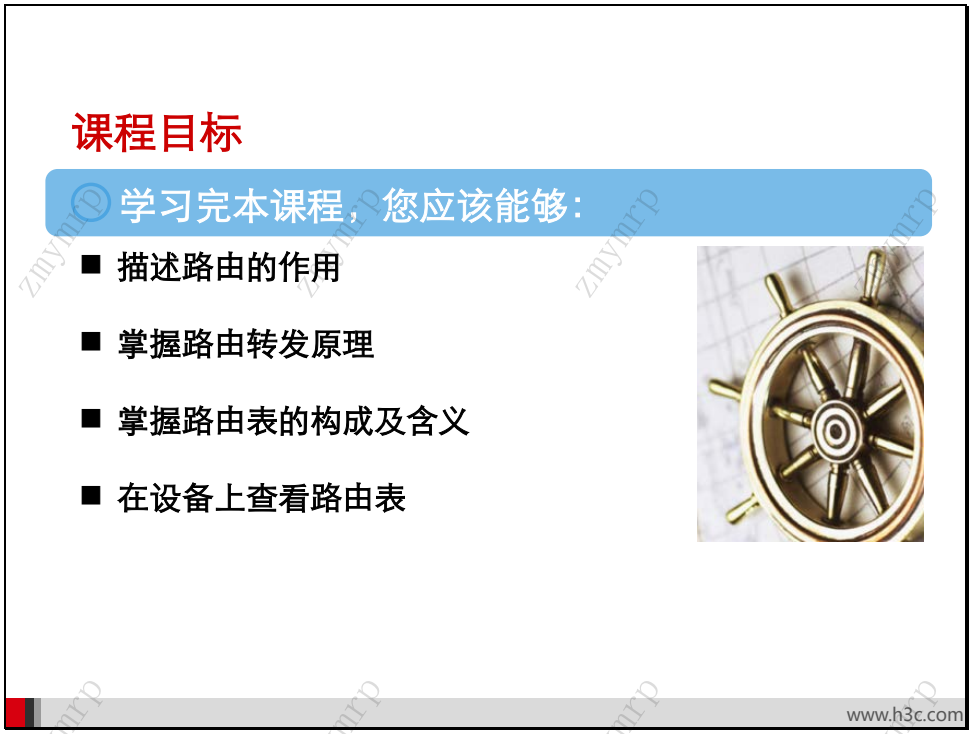
第21章 IP 路由原理

路由器是能够将数据报文在不同逻辑网段间转发的网络设备。路由（Route）是指导路由器如何进行数据报文发送的路径信息。每条路由都包含有目的地址、下一跳、出接口、到目的地的代价等要素，路由器根据自己的路由表对 IP 报文进行转发操作。

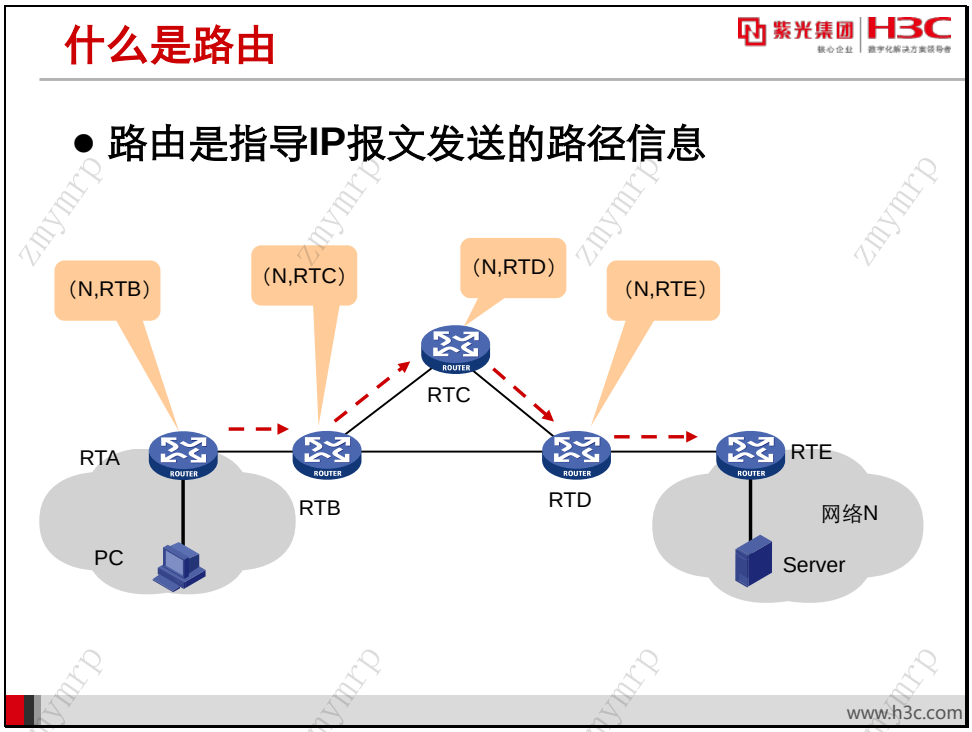
每一台路由器都有路由表（Routing Table），路由便存储在路由表中。

路由环路是由错误的路由导致的，它会造成 IP 报文在网络中循环转发，浪费网络带宽。

21.1 本章目标



21.2 什么是路由

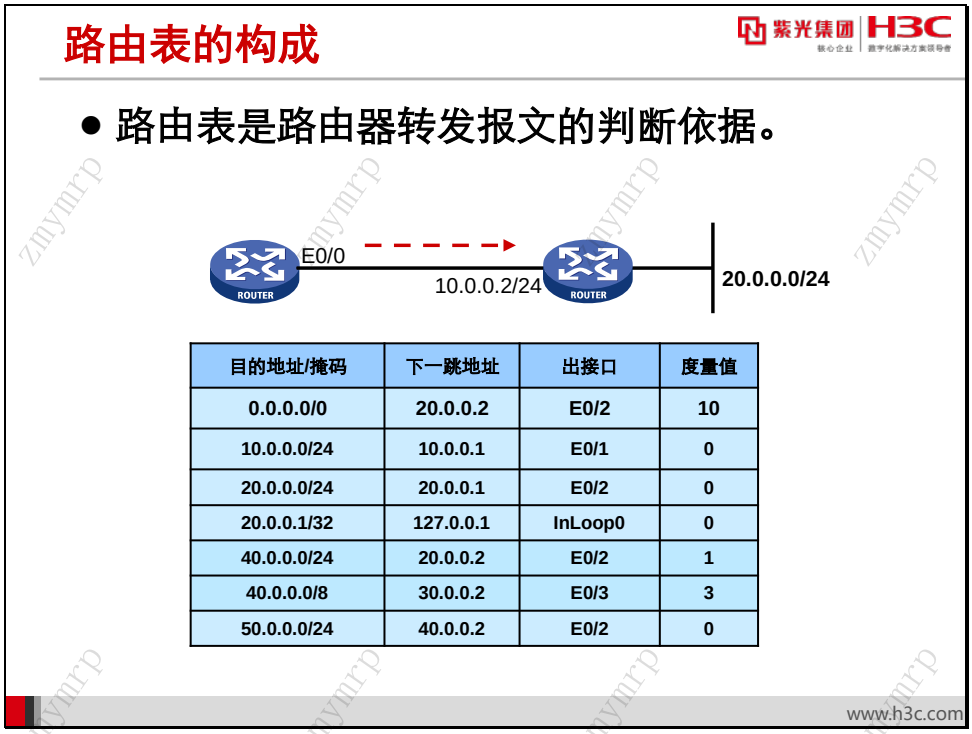


路由器提供了将异构网互联的机制，实现将一个数据包从一个网络发送到另一个网络。路由就是指导 IP 数据包发送的路径信息。

在互联网中进行路由选择要使用路由器，路由器只是根据所收到的数据报头的目的地址选择一个合适的路径（通过某一个网络），将数据包传送到下一个路由器，路径上最后的路由器负责将数据包送交目的主机。数据包在网络上的传输就好像是体育运动中的接力赛一样，每一个路由器只负责将数据包在本站通过最优的路径转发，通过多个路由器一站一站地接力将数据包通过最优路径转发到目的地。当然也有一些例外的情况，由于一些路由策略的实施，数据包通过的路径并不一定是最优的。

路由器的特点是逐跳转发。在上面这个网络中 RTA 收到 PC 发往 Server 的数据包后，将数据包转发给 RTB，RTA 并不负责指导 RTB 如何转发数据包。所以，RTB 必须自己将数据包转发给 RTC，RTC 再转发给 RTD，依次类推。这就是路由的逐跳性，即路由只指导本地转发行为，不会影响其它设备转发行为，设备之间的转发是相互独立的。

21.3 路由表



目的地址/掩码	下一跳地址	出接口	度量值
0.0.0.0/0	20.0.0.2	E0/2	10
10.0.0.0/24	10.0.0.1	E0/1	0
20.0.0.0/24	20.0.0.1	E0/2	0
20.0.0.1/32	127.0.0.1	InLoop0	0
40.0.0.0/24	20.0.0.2	E0/2	1
40.0.0.0/8	30.0.0.2	E0/3	3
50.0.0.0/24	40.0.0.2	E0/2	0

路由器转发数据包的依据是路由表。每个路由器中都保存着一张路由表，表中每条路由项都指明数据包到某子网或某主机应通过路由器的哪个物理端口发送，然后就可到达该路径的下一个路由器，或者不再经过别的路由器而传送到直接相连的网络中的目的主机。

路由表中包含了下列要素：

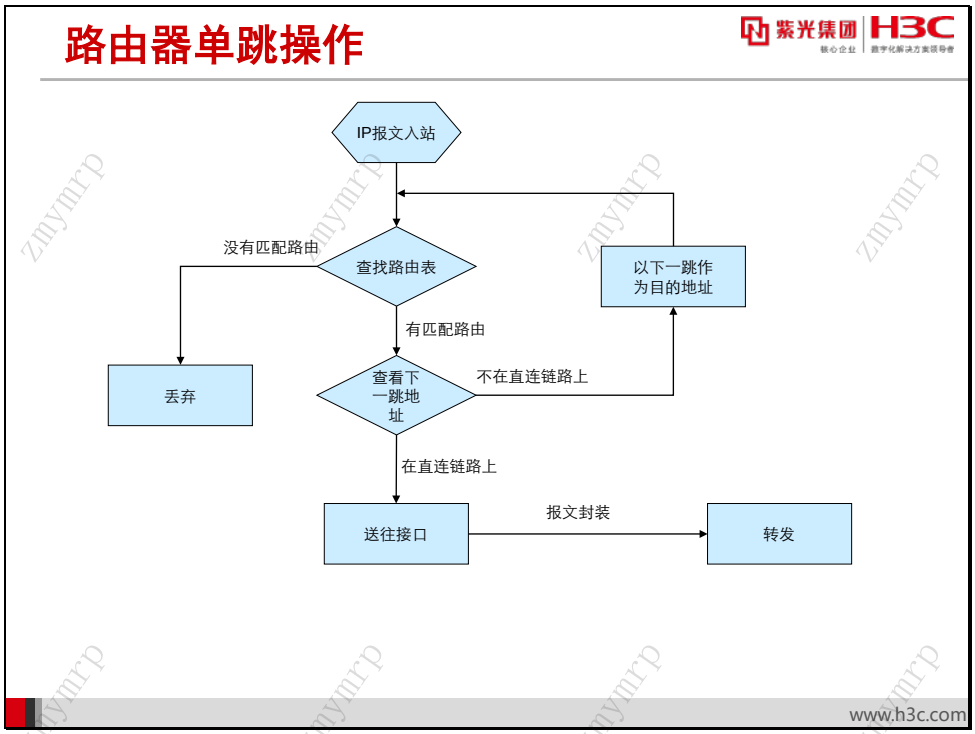
- 目的地址/网络掩码（Destination/Mask）：用来标识 IP 数据报文的目的地址或目的网络。将目的地址和网络掩码“逻辑与”后可得到目的主机或路由器所在网段的地址。例如：目的地址为 8.0.0.0，掩码为 255.0.0.0 的主机或路由器所在网段的地址为 8.0.0.0。掩码由若干个连续“1”构成，既可以用点分十进制表示，也可以用掩码中连续“1”的个数来表示。
- 出接口（Interface）：指明 IP 包将从该路由器哪个接口转发。
- 下一跳地址（Next-hop）：更接近目的网络的下一个路由器地址。如果只配置了出接口，下一跳 IP 地址是出接口的地址。
- 度量值（Metric）：说明 IP 包需要花费多大的代价才能到达目标。主要作用是当网络存在到达目的网络的多个路径时，路由器可依据度量值而选择一条较优的路径发送 IP 报文，从而保证 IP 报文能更快更好的到达目的。

根据掩码长度的不同，我们可以把路由表中路由项分为以下几个类型：

- 主机路由：掩码长度是 32 位的路由，表明此路由匹配单一 IP 地址；

- 子网路由：掩码长度小于 32 但大于 0，表明此路由匹配一个子网；
- 默认路由：掩码长度为 0，表明此路由匹配全部 IP 地址。

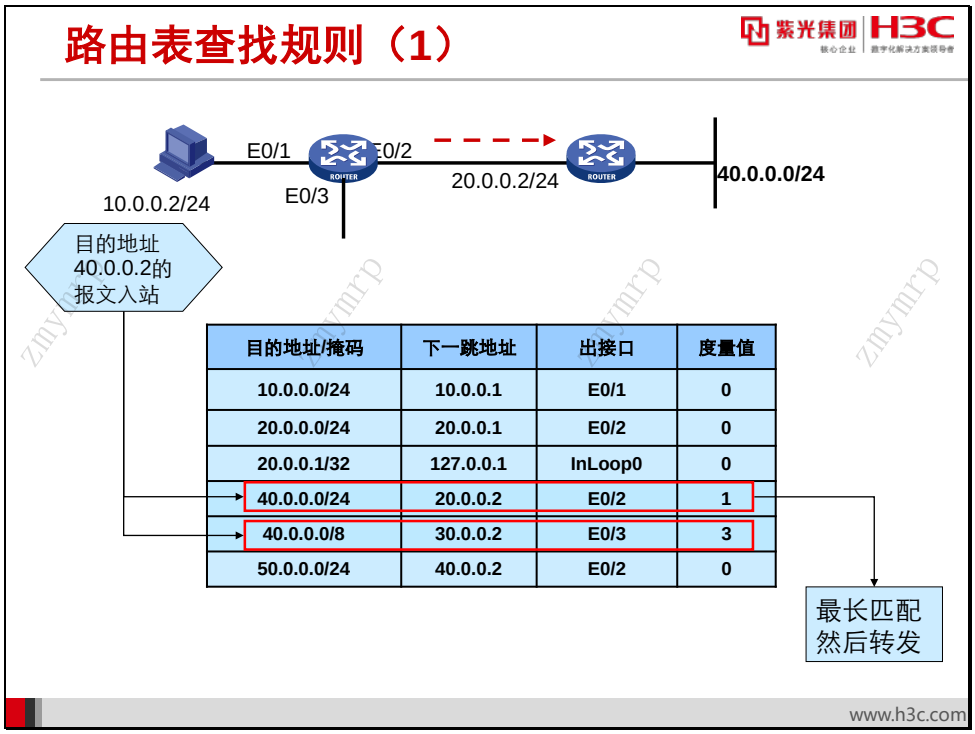
21.4 路由器单跳操作



路由器就是通过匹配路由表里的路由项来实现数据包的转发的。当路由器收到一个数据包的时候，将数据包的目的 IP 地址提取出来，然后与路由表中路由项包含的目的地址进行比较；如果与某路由项中的目的地址相同，则认为与此路由项匹配；如果没有路由项能够匹配，则丢弃该数据包。

路由器查看所匹配的路由项的下一跳地址是否在直连链路上，如果在直连链路上，则路由器根据此下一跳转发；如果不在直连链路上，则路由器还需要在路由表中再查找此下一跳地址所匹配的路由项。

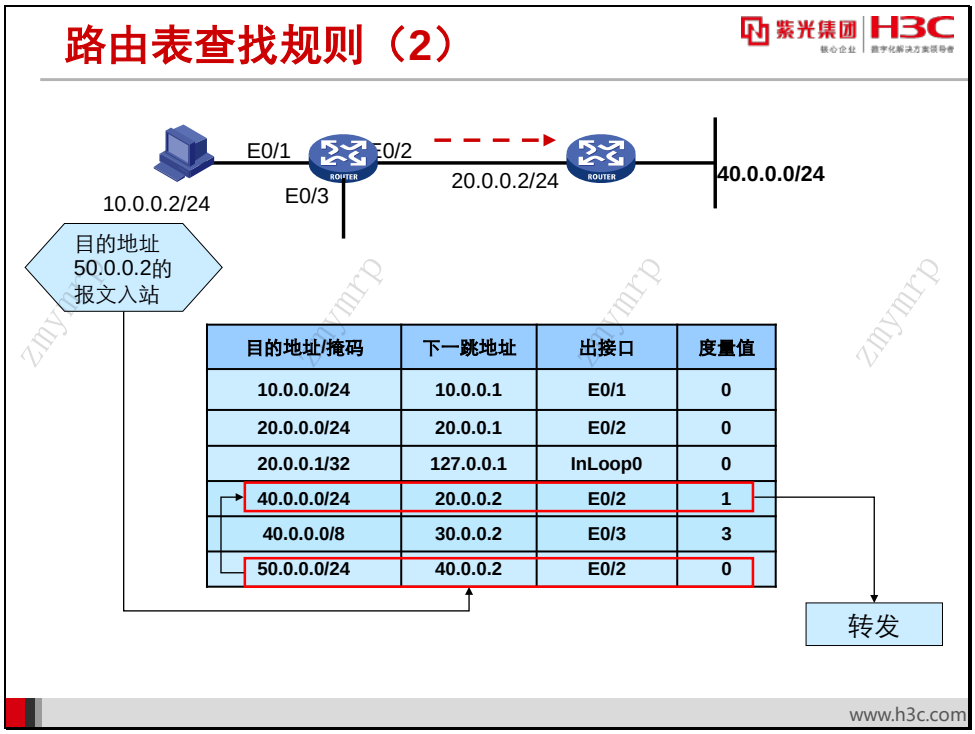
确定了最终的下一跳地址后，路由器将此报文送往对应的接口，接口进行相应的地址解析，解析出此地址所对应的链路层地址，然后对 IP 数据包进行数据封装并转发。



当路由表中存在多个路由项可以同时匹配目的 IP 地址时，路由查找进程会选择其中掩码最长的路由项用于转发，此为最长匹配原则。

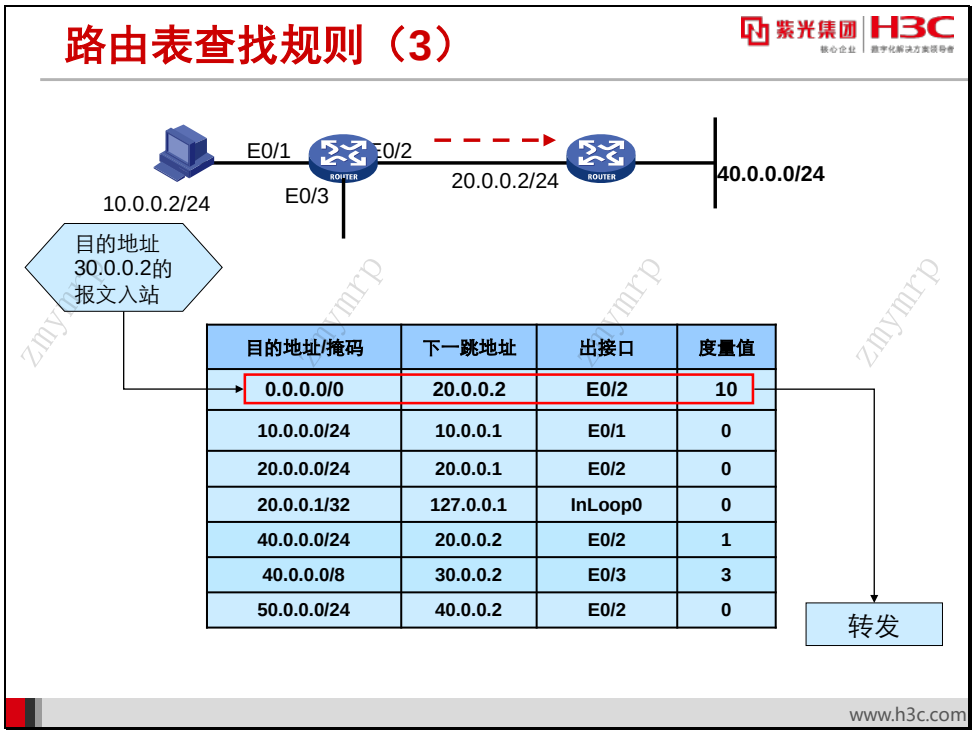
上图中，路由器接收到目的地址为 40.0.0.2 的数据包，经查找整个路由表，发现与路由 40.0.0.0/24 和 40.0.0.0/8 都能匹配。但根据最长匹配的原则，路由器会选择路由项 40.0.0.0/24，根据该路由项转发数据包。

由以上过程可知，路由表中路由项数量越多，所需查找及匹配的次数则越多。所以一般路由器都有相应的算法来优化查找速度，加快转发。



如果所匹配的路由项的下一跳地址不在直连链路上，路由器还需要对路由表进行迭代查找，找出最终的下一跳来。

如上图中，路由器接收到目的地址为 50.0.0.2 的数据包后，经查找路由表，发现与路由表中的路由项 50.0.0.0/24 能匹配。但此路由项的下一跳 40.0.0.2 不在直连链路上，所以路由器还需要在路由表中查找到达 40.0.0.2 的下一跳。经过查找，到达 40.0.0.2 的下一跳是 20.0.0.2，此地址在直连链路上，则路由器按照该路由项转发数据包。

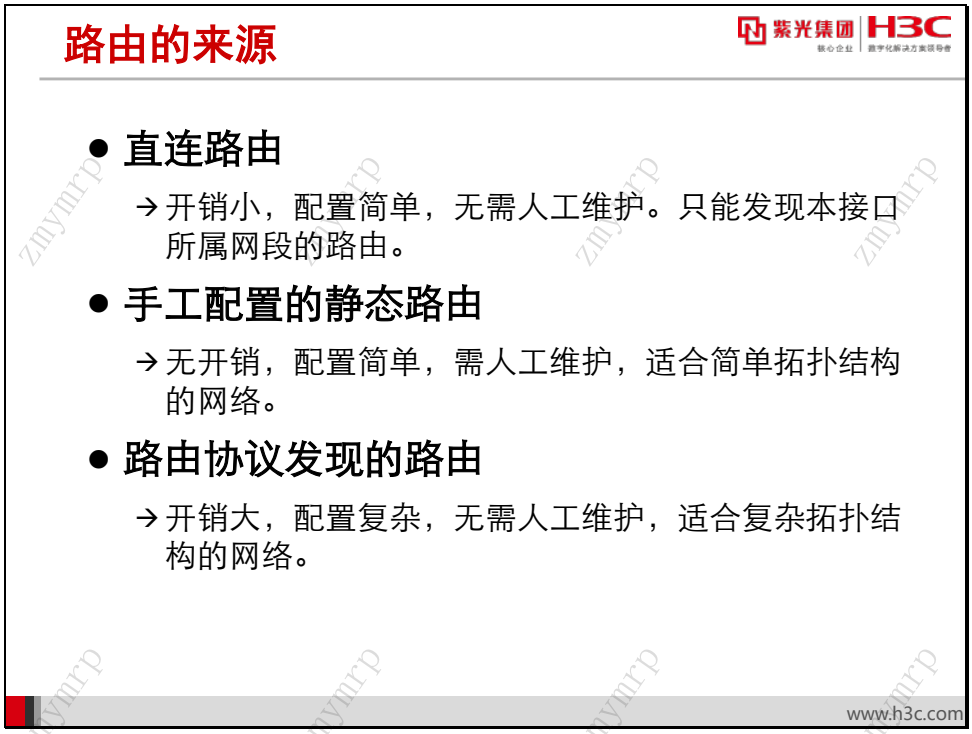


如果路由表中没有路由项能够匹配数据包，则丢弃该数据包。但是，如果在路由表中有默认路由存在，则路由器按照默认路由来转发数据包。默认路由又称为默认路由，其目的地址/掩码为 0.0.0.0/0。

如上图中，路由器收到目的地址为 30.0.0.2 的数据包后，查找路由表，发现没有子网或主机路由匹配此地址，所以按照默认路由转发。

默认路由能够匹配所有 IP 地址。但因为它的掩码最短，所以只有在没有其它路由匹配数据包的情况下，系统才会按照默认路由转发。

21.5 路由的来源



路由的来源主要有 3 种：

- 直连（Direct）路由

直连路由不需要配置，当接口存在 IP 地址并且状态正常时，由路由进程自动生成。它的特点是开销小，配置简单，无需人工维护，但只能发现本接口所属网段的路由。

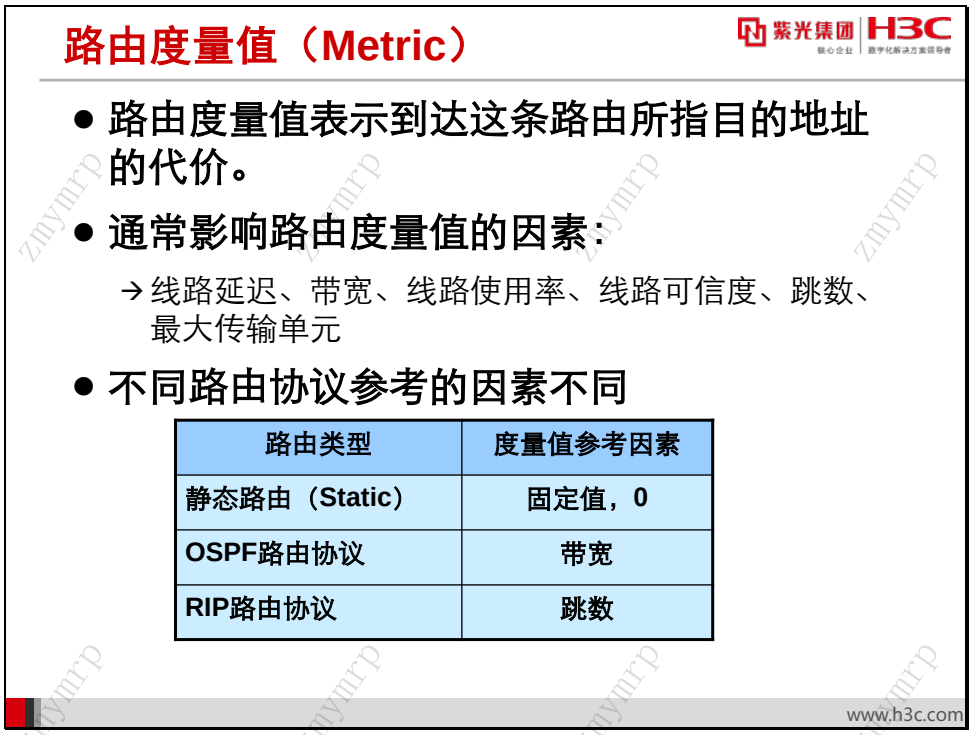
- 手工配置的静态（Static）路由

由管理员手工配置而成的路由称之为静态路由。通过静态路由的配置可建立一个互通的网络，但这种配置问题在于：当一个网络故障发生后，静态路由不会自动修正，必须有管理员的介入。静态路由无开销，配置简单，适合简单拓扑结构的网络。

- 动态路由协议（Routing Protocol）发现的路由

当网络拓扑结构十分复杂时，手工配置静态路由工作量大而且容易出现错误，这时就可用动态路由协议（如 RIP、OSPF 等），让其自动发现和修改路由，避免人工维护。但动态路由协议开销大，配置复杂。

21.6 路由的度量



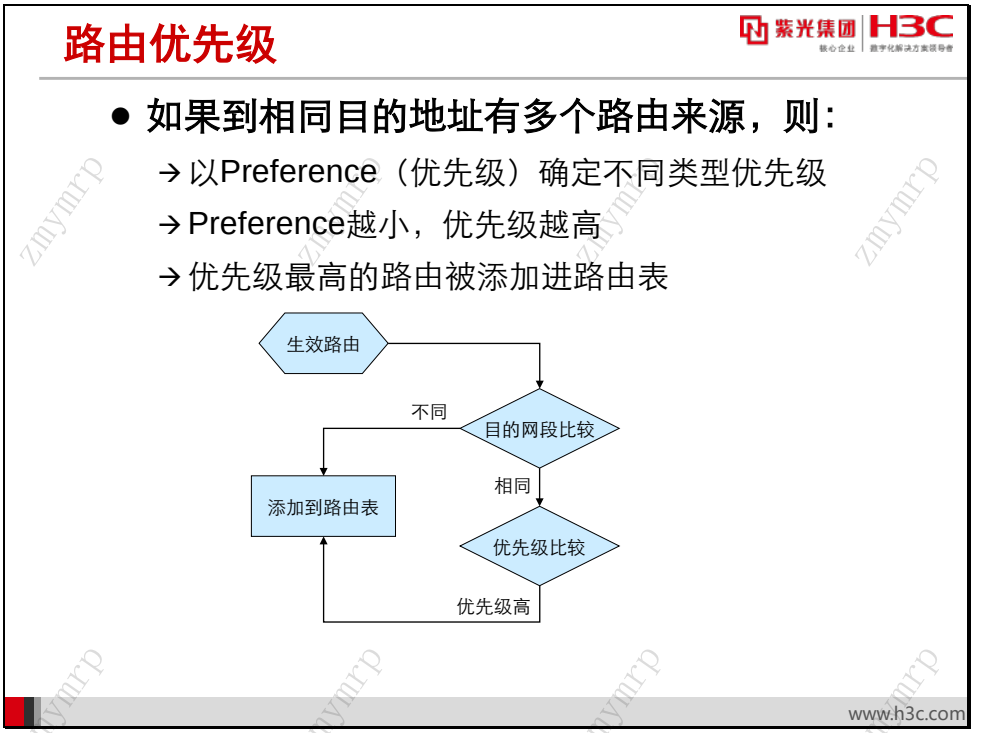
路由度量值（Metric）表示到达这条路由所指目的地址的代价，也称为路由权值。各路由协议定义度量值的方法不同，通常会考虑以下因素：

- 跳数
- 链路带宽
- 链路延时
- 链路使用率
- 链路可信度
- 链路 MTU

不同的动态路由协议会选择其中的一种或几种因素来计算度量值。在常用的路由协议里，RIP 使用“跳数”来计算度量值，跳数越小，其路由度量值也就越小；而 OSPF 使用“链路带宽”来计算度量值，链路带宽越大，路由度量值也就越小。度量值通常只对动态的路由协议有意义，静态路由协议的度量值统一规定为 0。

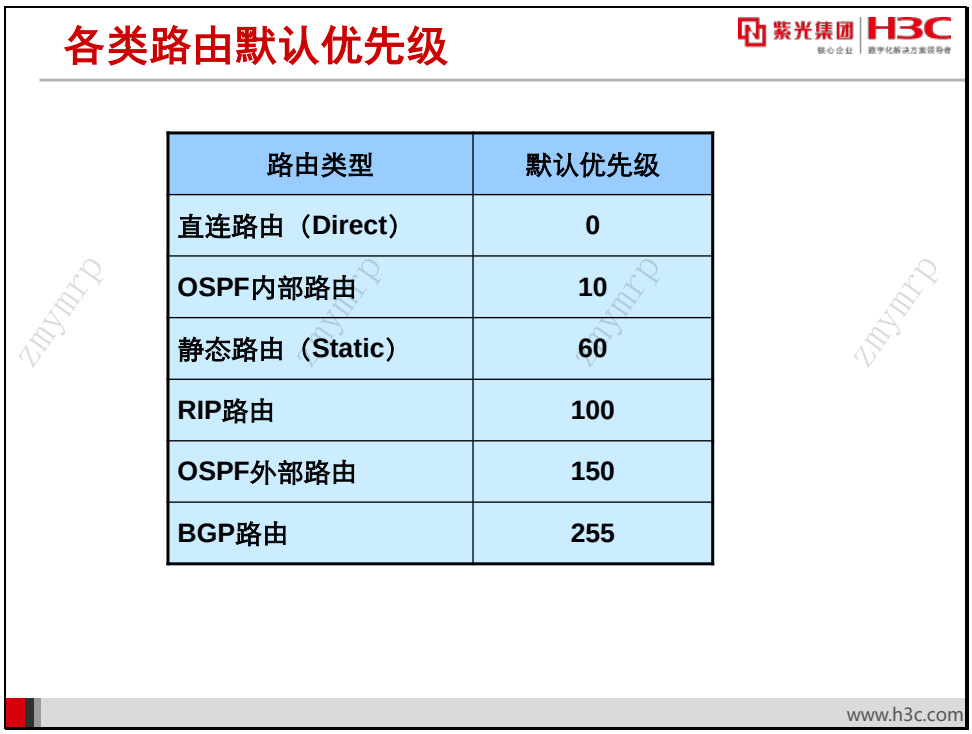
路由度量值只在同一种路由协议内有比较意义，不同的路由协议之间的路由度量值没有可比性，也不存在换算关系。

21.7 路由优先级



路由优先级（Preference）代表了路由协议的可信度。

在计算路由信息的时候，因为不同路由协议所考虑的因素不同，所以计算出的路径也可能会不同。具体表现就是到相同的目的地址，不同的路由协议（包括静态路由）所生成路由的下一跳可能会不同。在这种情况下，路由器会选择哪一条路由作为转发报文的依据呢？此时就取决于路由优先级，具有较高优先级（数值越小表明优先级越高）的路由协议发现的路由将成为最优路由，并被加入路由表中。

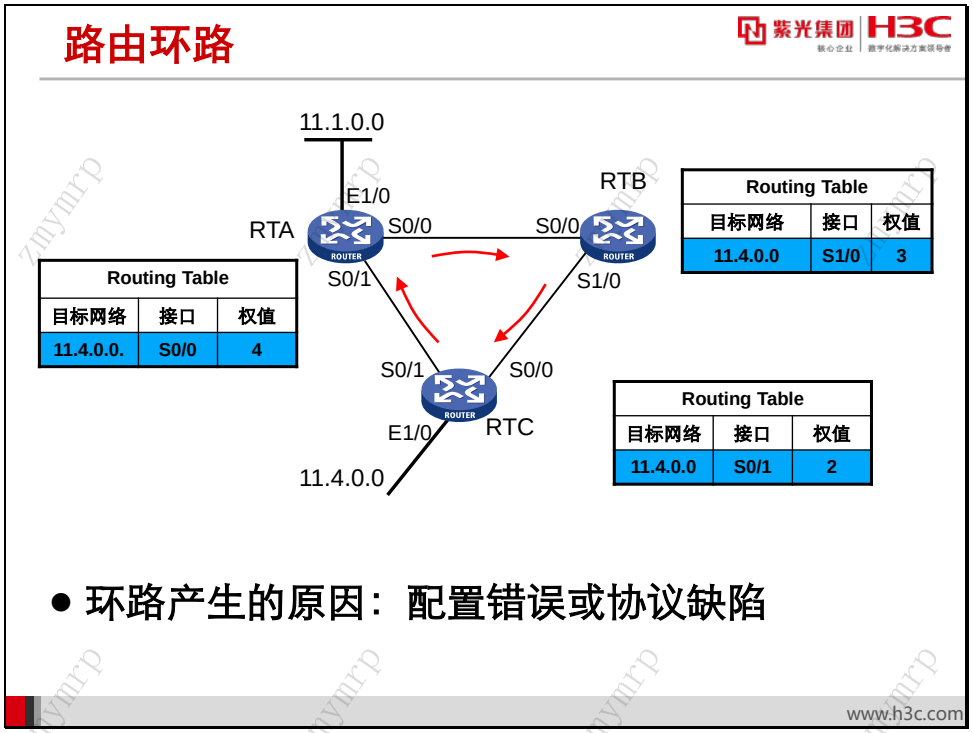


路由类型	默认优先级
直连路由（Direct）	0
OSPF内部路由	10
静态路由（Static）	60
RIP路由	100
OSPF外部路由	150
BGP路由	255

不同厂家的路由器对于各种路由协议优先级的规定各不相同。H3C 路由器的默认优先级如上图所示。

除了直连路由（DIRECT）外，各动态路由协议的优先级都可根据用户需求，手工进行配置。另外，每条静态路由的优先级都可以不相同。

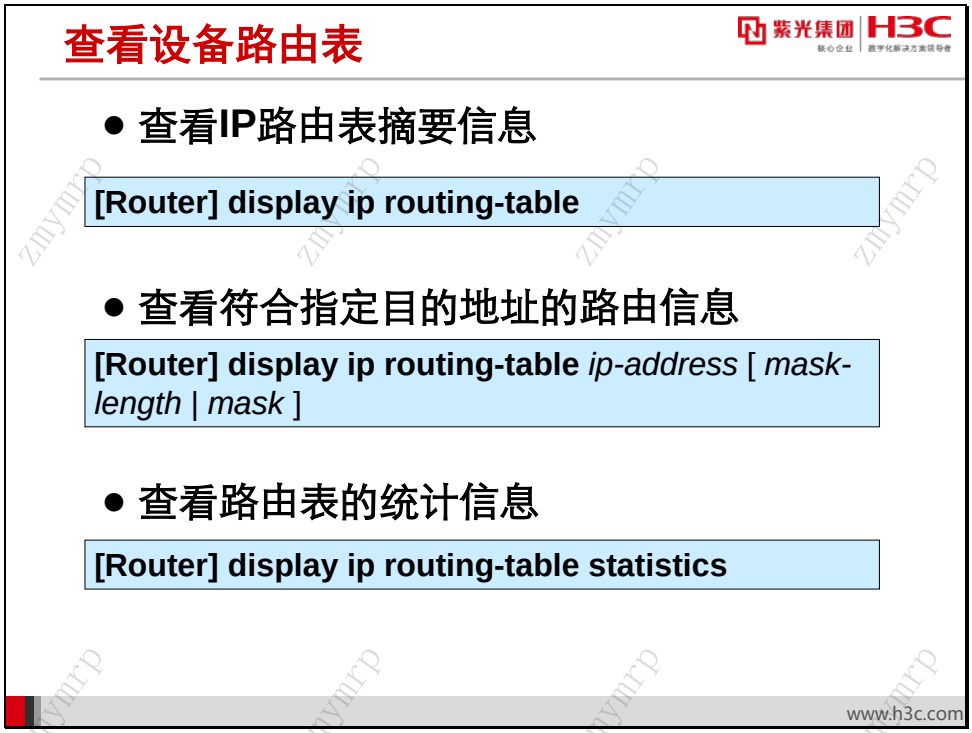
21.8 路由环路



路由环路会使数据转发形成死循环，不能到达目的地。如图中 RTA 收到目的为 11.4.0.0 的数据包后，查看路由表，发现其下一跳是 S0/0 接口，于是转发给 RTB；RTB 发现下一跳是 S1/0，于是又转发给 RTC；RTC 中路由表的下一跳指向 RTA，所以 RTC 又将数据包转发回 RTA。如此在三台路由器间循环转发，直到数据包中 TTL 字段值为 0 后丢弃。这同时也导致了巨大的消耗浪费。

路由环路的主要生成原因是配置了错误的静态路由或网络规划错误。比如，在两台路由器上配置到相同目的地址的路由表项，下一跳互相指向对方，就会造成路由环路。另外，某些动态路由协议在特定环境下或配置不当，也有可能产生环路。

21.9 查看设备的路由表



查看设备路由表的目的是查找所需的路由信息，验证所做的路由配置。

最常用的命令是查看 IP 路由表摘要信息。在任意视图下用如下命令来查看：

display ip routing-table

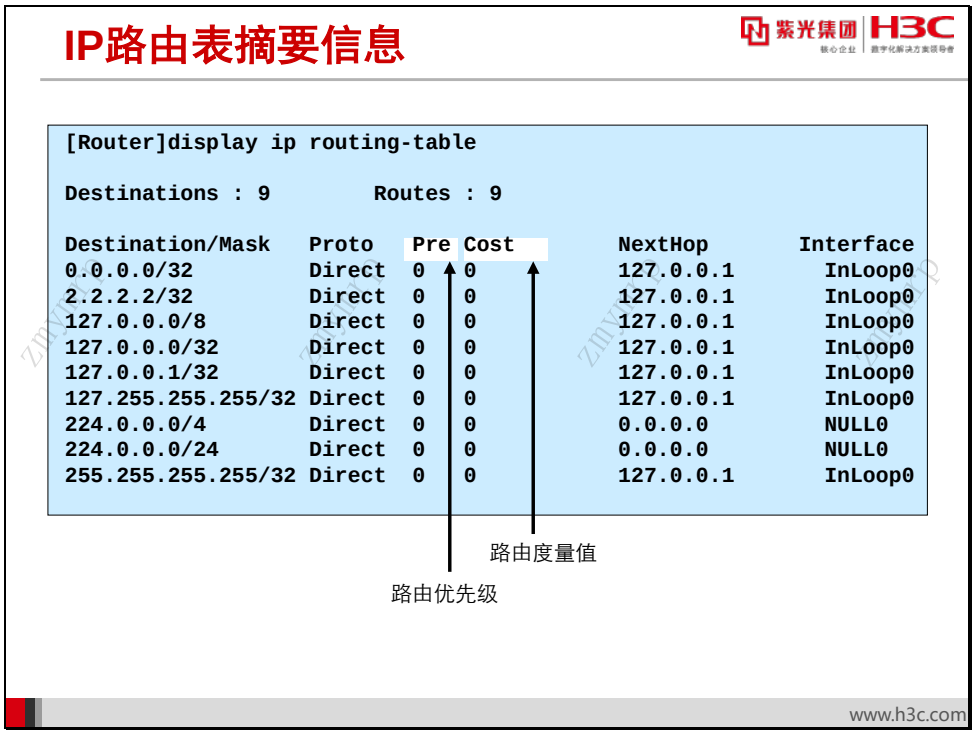
如果想查看某一条具体的路由，可以在任意视图下用如下命令来查看：

display ip routing-table *ip-address* [*mask-length* **|** *mask*]

比如，用命令 display ip routing-table 1.1.1.1 就可以查看匹配目标地址 1.1.1.1 的所有路由项。

有时候，我们想了解路由表的综合统计信息，如总路由数量、RIP 路由数量、OSPF 路由数量、激活路由数量等，可以在任意视图下用如下命令来查看：

display ip routing-table statistics

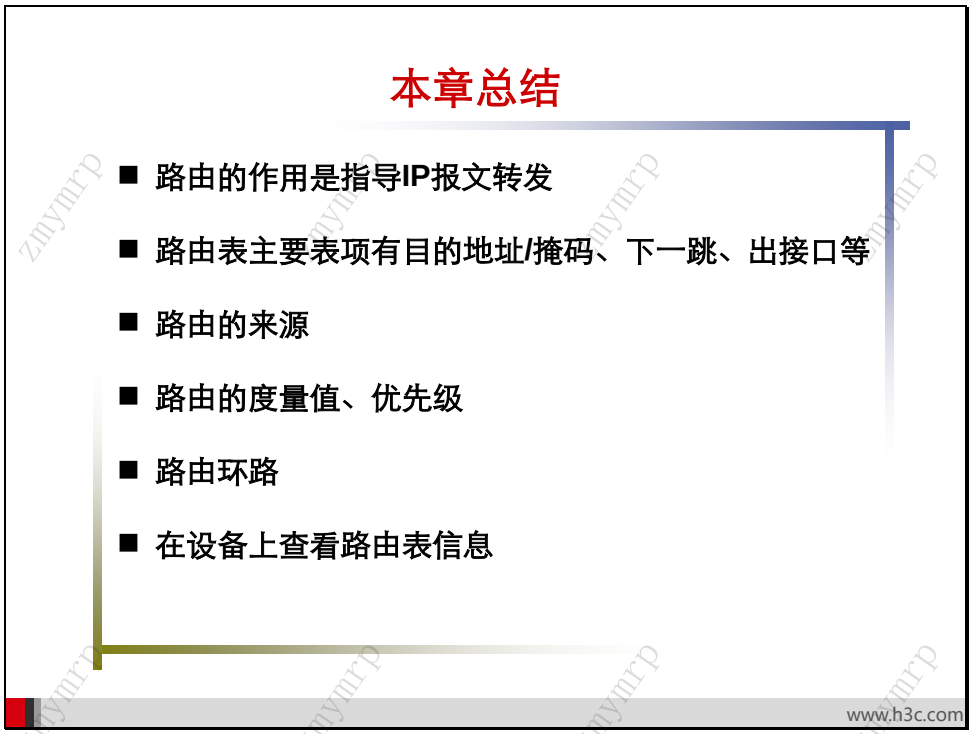


上图中列出了 **display ip routing-table** 命令输出。由上图所示路由表可以得知，该表目前共有 9 条路由。

图中路由信息中的各字段含义如下表：

字段	描述
Destinantions	目的地址个数
Routes	路由条数
Destination/Mask	目的地址/掩码长度
Proto	发现该路由的路由协议
Pre	路由的优先级
Cost	路由的度量值
Nexthop	此路由的下一跳地址
Interface	出接口，即到该目的网段的数据包将从此接口发出

21.10 本章总结

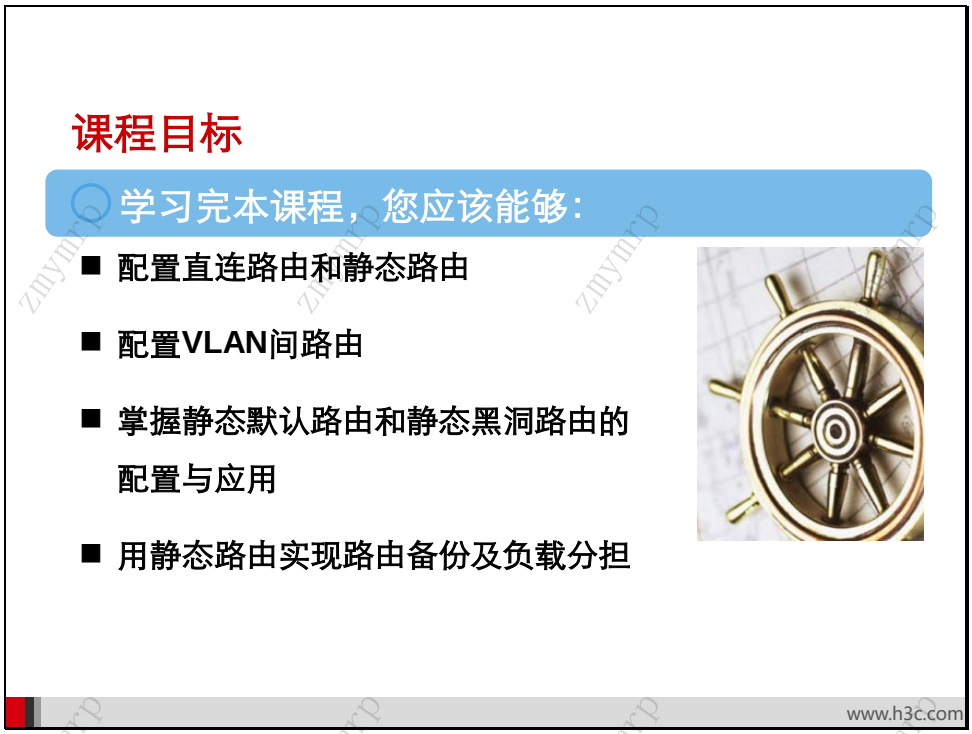

本章总结
路由的作用是指导IP报文转发
路由表主要表项有目的地址/掩码、下一跳、出接口等
路由的来源
路由的度量值、优先级
路由环路
在设备上查看路由表信息
www.h3c.com

第22章 直连路由和静态路由

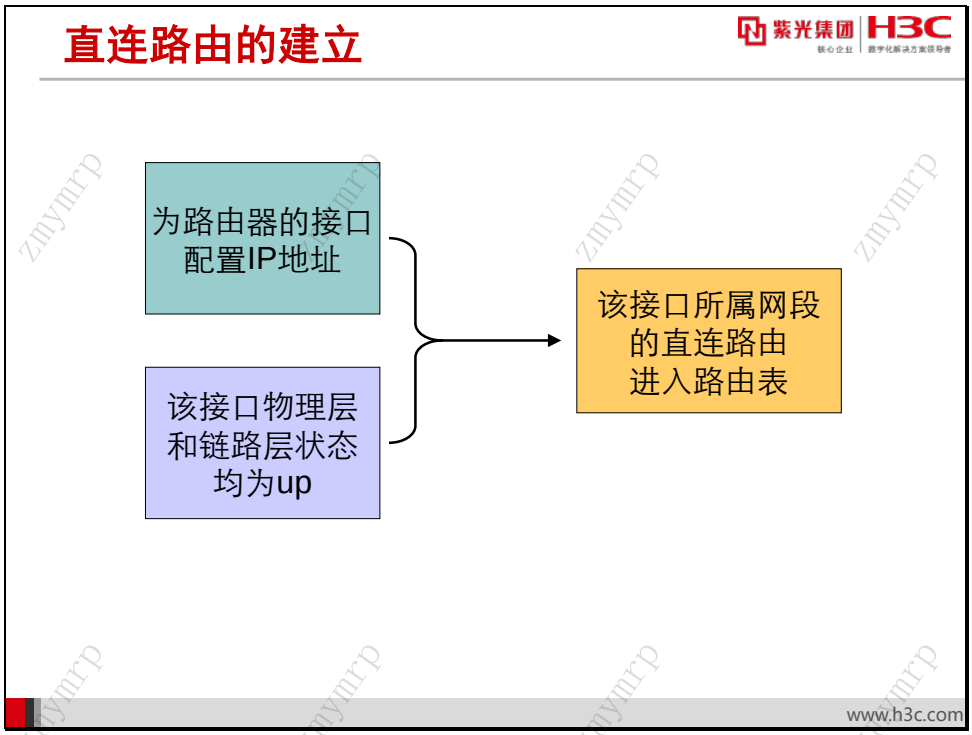
对路由器而言，无需任何路由配置，即可获得其直连网段的路由。路由器最初始的功能就是在若干局域网直接提供路由功能，VLAN 间路由就是这一功能的直接体现。理解直连路由和 VLAN 间路由是理解各种复杂网络路由的基础，也是构建小型网络的基础。

静态路由是一种由管理员手工配置的路由，适用于拓扑简单的网络。恰当地设置和使用静态路由可以有效地改进网络的性能。

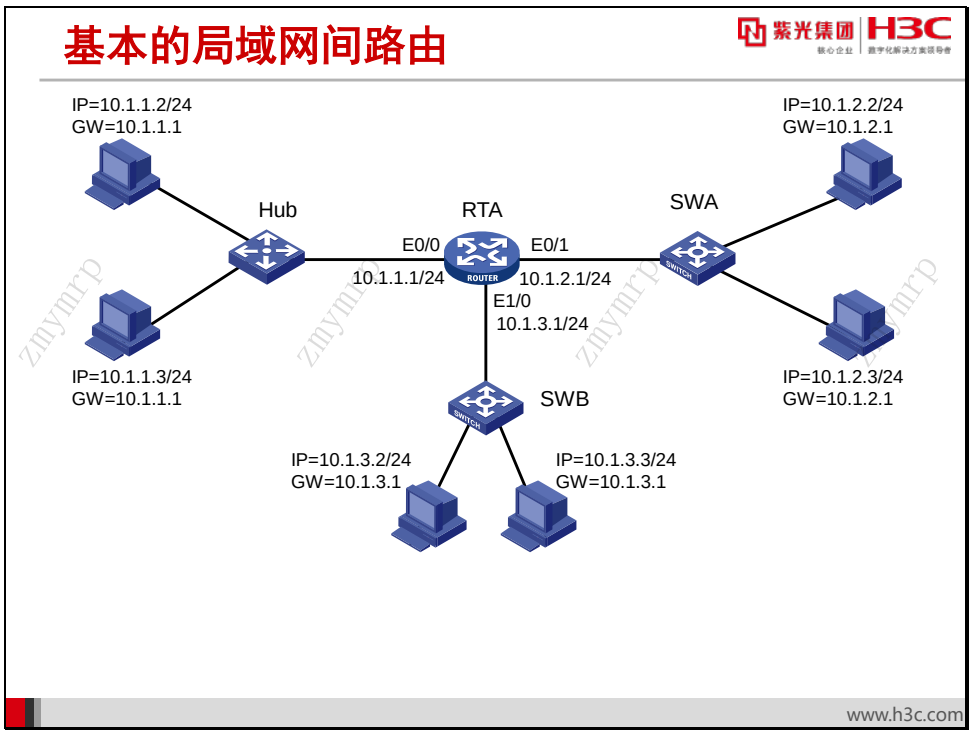
22.1 本章目标



22.2 直连路由



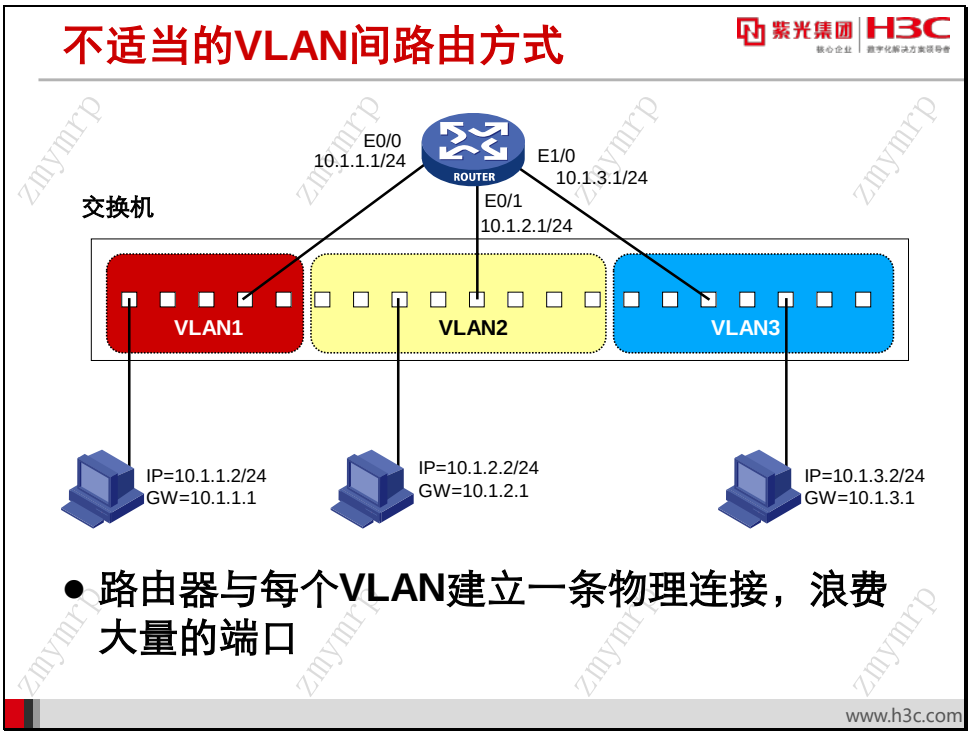
直连路由是指路由器接口直接相连的网段的路由。直连路由不需要特别的配置，只需在路由器的接口上配置 IP 地址即可。但路由器会根据接口的状态决定是否使用此路由。如果接口的物理层和链路层状态均为 up，路由器即认为接口工作正常，该接口所属网段的路由即可生效并以直连路由出现在路由表中；如果接口状态为 down，路由器认为接口工作不正常，不能通过该接口到达其地址所属网段，也就不能以直连路由出现在路由表中。



基本的局域网间路由如图所示。其中路由器 RTA 的三个以太口分别连接三个局域网段，只需在 RTA 上为其三个以太口配置 IP 地址，即可为 10.1.1.0/24、10.1.2.0/24 和 10.1.3.0/24 网段提供路由服务。

22.3 VLAN间路由

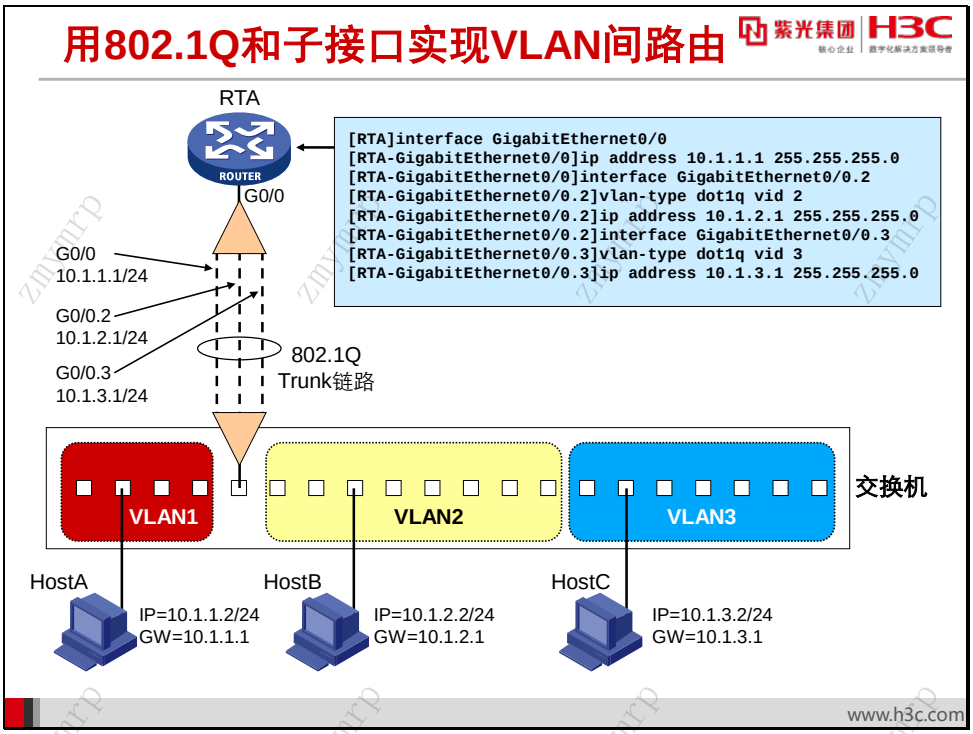
22.3.1 不适当的 VLAN 间路由方式



引入 VLAN 之后，每个交换机被划分成多个 VLAN，而每个 VLAN 对应一个 IP 网段。为了在 VLAN 之间进行路由，路由器到各个 VLAN 就必须各有一个物理接口和一条物理连接。

如图所示，路由器要为三个 VLAN 提供 VLAN 间路由，就必须用三个以太口分别连接到交换机的三个 VLAN 的三个物理接口上。显然，在 VLAN 数量较大时，这种方式要求占用路由器和交换机的大量物理接口，并需要大量的物理连线，因而是难以接受的。

22.3.2 用 802.1Q 和子接口实现 VLAN 间路由



为了避免物理端口和线缆的浪费，简化连接方式，可以使用 802.1Q 封装和子接口，通过一条物理链路实现 VLAN 间路由。这种方式也被形象地称为“单臂路由”。

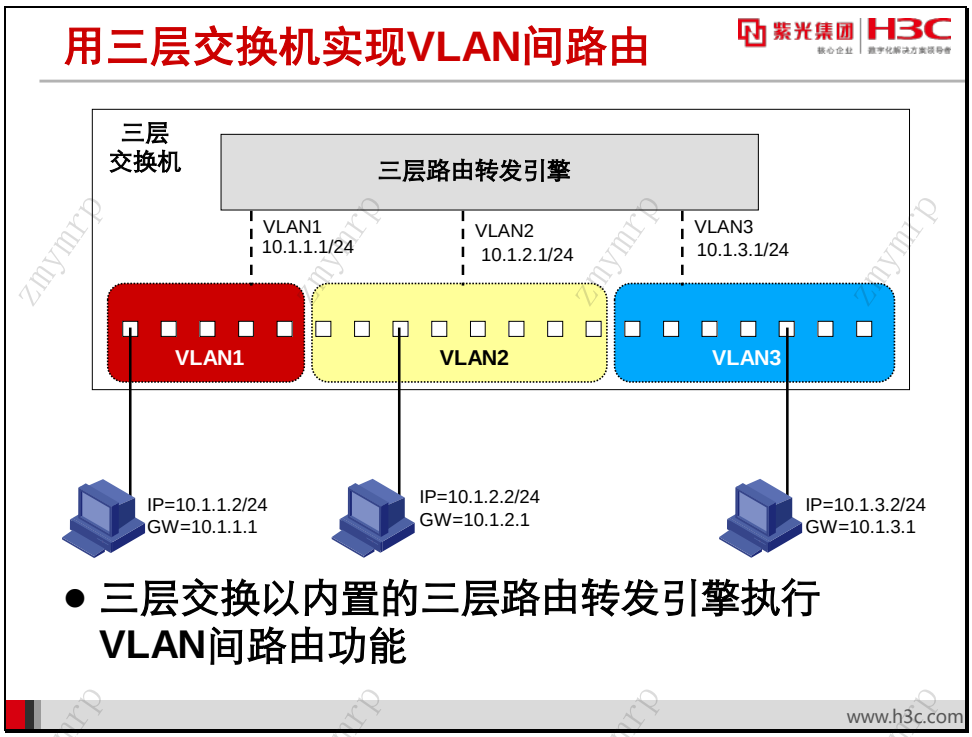
如图所示，交换机通过 802.1Q 封装的 Trunk 链路连接到路由器的千兆以太口 G0/0 上。每一个 VLAN 的数据都可以通过 802.1Q 标记识别出来。在路由器上则为 G0/0 配置了子接口，每个子接口配置了属于相应 VLAN 网段的 IP 地址，并且配置了相应 VLAN 的 802.1Q 标记值。

当 HostB 向 HostA 发送 IP 包时，该 IP 包首先被封装成以太帧，通过 Trunk 链路发送给路由器，在 Trunk 链路上其 802.1Q VLAN ID 为 2。路由器收到此帧后，根据 VLAN ID 将其交给子接口 G0/0.2 处理。

路由器查找路由表，发现 HostA 处于接口 G0/0 所在网段，因而将此数据包封装成帧从接口 G0/0 发出，发送时不加 802.1Q 标记。由于交换机默认 PVID 值为 1，此帧到达交换机后，交换机认为此为 VLAN1 数据，即可将其转发给 HostA。

这种 VLAN 间路由方式节省了物理端口和线缆。但应注意 Trunk 链路需承载所有 VLAN 间路由数据，因此通常应选择带宽较高的链路。

22.3.3 用三层交换机实现 VLAN 间路由



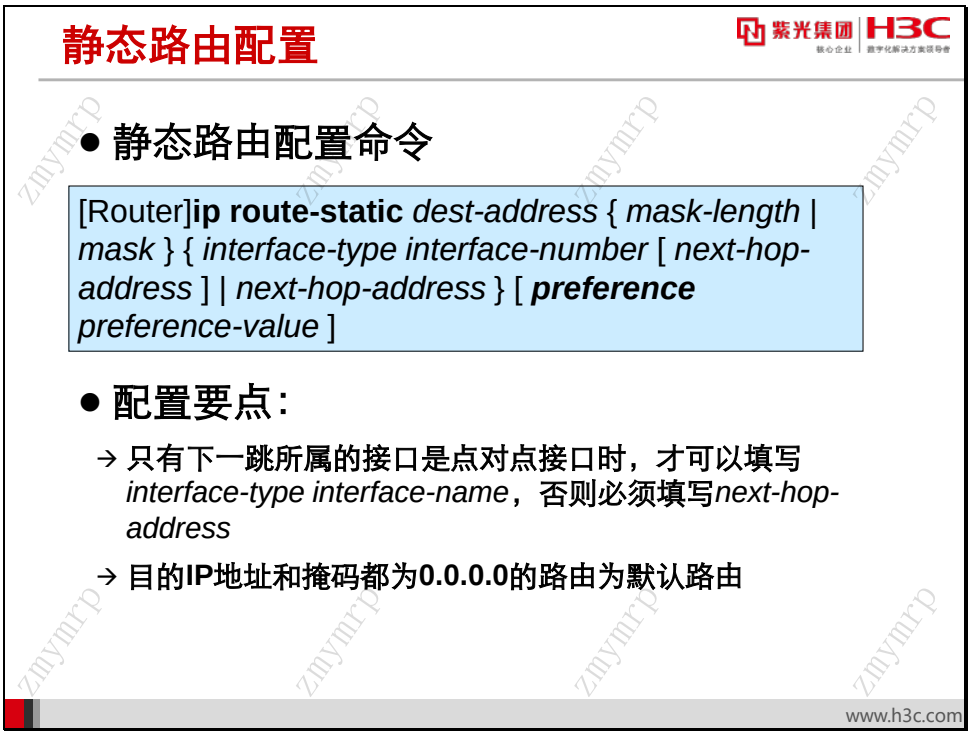
采用“单臂路由”方式进行 VLAN 间路由时，数据在 Trunk 链路上往返发送引入了一定的延迟，VLAN 间路由的大量数据对软件实现的路由器也会造成较大压力。解决的方法是使用三层交换机。

三层交换机为每个 VLAN 创建一个虚拟的三层 VLAN 接口，这个接口像路由器接口一样工作。只需为 VLAN 接口配置相应的 IP 地址，即可实现 VLAN 间路由功能。

三层交换机通过内置的三层路由转发引擎在 VLAN 间进行路由转发。由于硬件实现的三层路由转发引擎速度高，吞吐量大，而且避免了外部物理连接带来的延迟和不稳定性，因此三层交换机的路由转发性能高于路由器实现的 VLAN 间路由。

22.4 静态路由

22.4.1 静态路由配置



静态路由是一种特殊的路由，它由管理员手工配置。当网络结构比较简单时，只需配置静态路由就可以使网络正常工作。恰当地设置和使用静态路由可以改进网络的性能，并可为重要的网络应用保证带宽。

静态路由的缺点在于：当网络发生故障或者拓扑发生变化后，可能会出现路由不可达，导致网络中断，此时必须由网络管理员手工修改静态路由的配置。

静态路由的配置在系统视图下进行，命令为：

ip route-static *dest-address* { *mask-length* | *mask* } { *interface-type interface-number* [*next-hop-address*] | *next-hop-address* } [**preference** *preference-value*]

其中各参数的解释如下：

- *dest-address*：静态路由的目的 IP 地址，点分十进制格式。
- *mask-length*：掩码长度，取值范围为 0～32。
- *mask*：IP 地址的掩码，点分十进制格式。
- *interface-type interface-number*：指定静态路由的出接口类型和接口号。
- *next-hop-address*：指定路由的下一跳的 IP 地址，点分十进制格式。

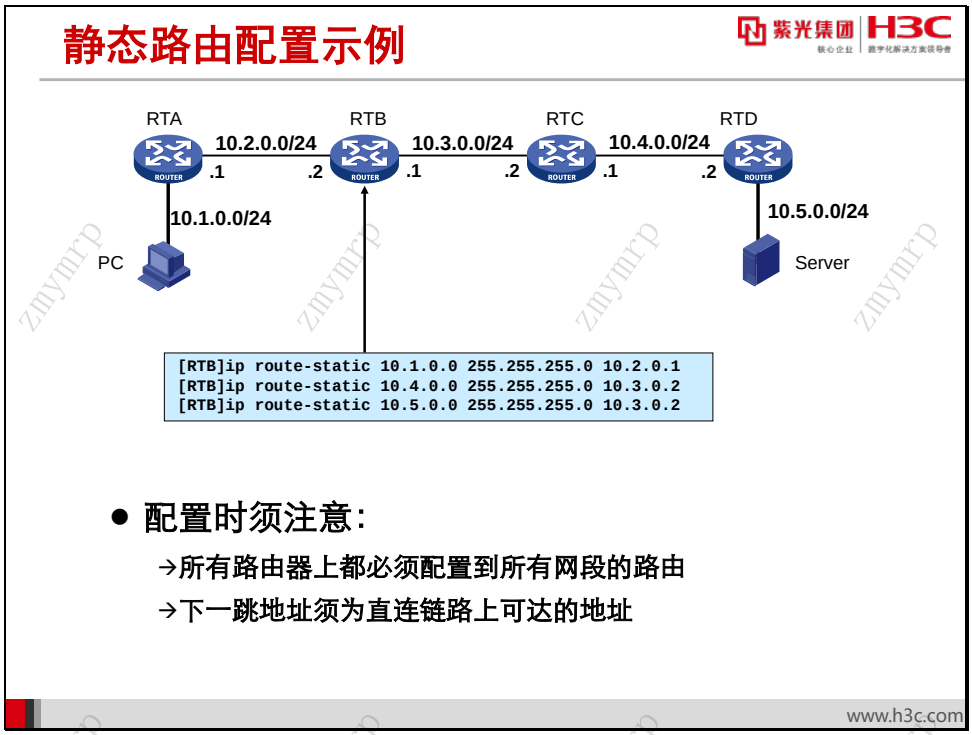
- **preference** *preference-value* ：指定静态路由的优先级，取值范围 1～255，默认值为 60。

在配置静态路由时，可指定发送接口 *interface-type interfacce-name*，如 Serial 2/0；也可指定下一跳地址 *next-hop-address*，如 10.0.0.2。如果出接口是广播类型接口（如以太网接口、VLAN 接口等），则必须指定下一跳地址；而如下情况则可以指定发送接口：

- 当目标地址是一个主机地址，而且该目的地址就在该接口的直连网络中时；
- 当到达目标地址的出接口是点到点接口时。

比如串口封装 PPP 协议，系统能够通过 PPP 协商而获取对端设备的 IP 地址。这时就可以不用指定下一跳地址，只需指定发送接口即可。

22.4.2 静态路由配置示例



例图中，在 PC 与 Server 之间路由器上配置静态路由，以使 PC 能够与 Server 通信。

配置 RTA：

```
[RTA] ip route-static 10.3.0.0 255.255.255.0 10.2.0.2
[RTA] ip route-static 10.4.0.0 255.255.255.0 10.2.0.2
[RTA] ip route-static 10.5.0.0 255.255.255.0 10.2.0.2
```

配置 RTB：

```
[RTB] ip route-static 10.1.0.0 255.255.255.0 10.2.0.1
[RTB] ip route-static 10.4.0.0 255.255.255.0 10.3.0.2
[RTB] ip route-static 10.5.0.0 255.255.255.0 10.3.0.2
```

配置 RTC：

```
[RTC] ip route-static 10.1.0.0 255.255.255.0 10.3.0.1
[RTC] ip route-static 10.2.0.0 255.255.255.0 10.3.0.1
[RTC] ip route-static 10.5.0.0 255.255.255.0 10.4.0.2
```

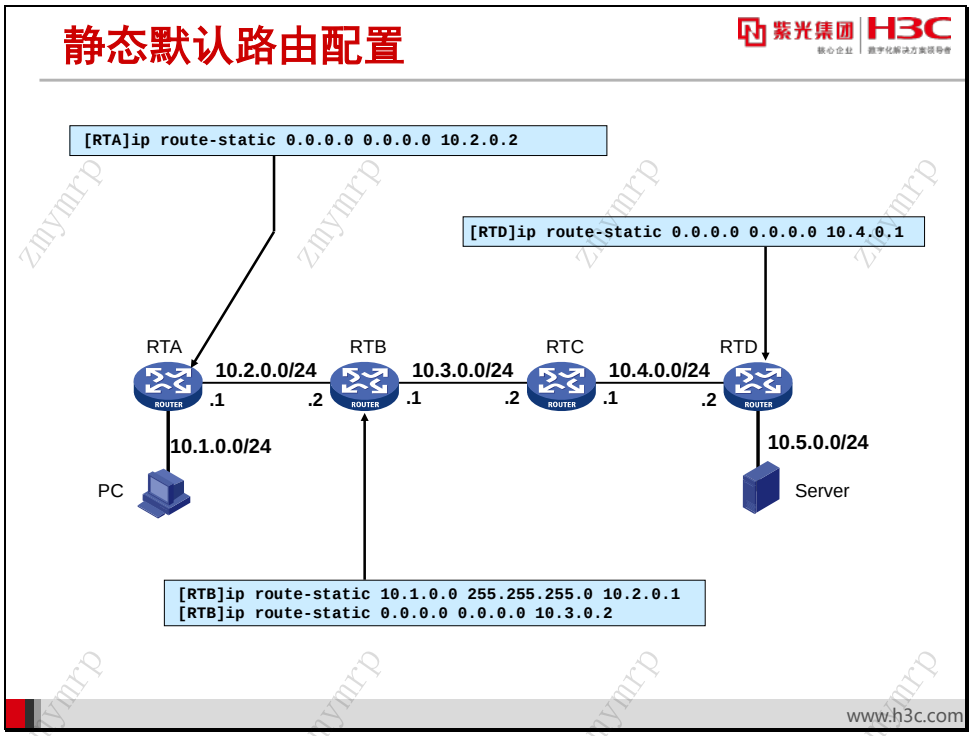
配置 RTD：

```
[RTD] ip route-static 10.1.0.0 255.255.255.0 10.4.0.1
[RTD] ip route-static 10.2.0.0 255.255.255.0 10.4.0.1
[RTD] ip route-static 10.3.0.0 255.255.255.0 10.4.0.1
```

因为路由器是逐跳转发的，所以在配置静态路由时，需要注意在所有路由器上配置到达所有网段的路由。

在 IP 转发过程中，路由器通过下一跳 IP 地址找到对应的链路层地址，然后在出接口上对 IP 报文进行链路层封装。所以在配置静态路由时，需要注意下一跳地址应该是直连链路上可达的地址，否则路由器无法解析出对应的链路层地址。

22.5 静态默认路由的配置



在路由器上合理配置默认路由能够减少路由表中表项数量，节省路由表空间，加快路由匹配速度。

默认路由可以手工配置，也可以由某些动态路由协议生成，如 OSPF、IS-IS 和 RIP。

默认路由经常应用在末端（Stub）网络中。末端网络是指仅有一个出口连接外部的网络，如例图中 PC 和 Server 所在的网络。图中 PC 通过 RTA 来到达外部网络，所有的数据包由 RTA 进行转发。在上一节中，在 RTA 上配置了 3 条静态路由，其下一跳都是 10.2.0.2；所以可以配置 1 条默认路由来代替这 3 条静态路由。

配置 RTA：

```
[RTA] ip route-static 0.0.0.0 0.0.0.0 10.2.0.2
```

这样就达到了减少路由表中表项数量的目的。

同理，在其它路由器上也可以配置默认路由。

配置 RTB：

```
[RTB] ip route-static 10.1.0.0 255.255.255.0 10.2.0.1
[RTB] ip route-static 0.0.0.0 0.0.0.0 10.3.0.2
```

配置 RTC：

```
[RTC] ip route-static 0.0.0.0 0.0.0.0 10.3.0.1
[RTC] ip route-static 10.5.0.0 255.255.255.0 10.4.0.2
```

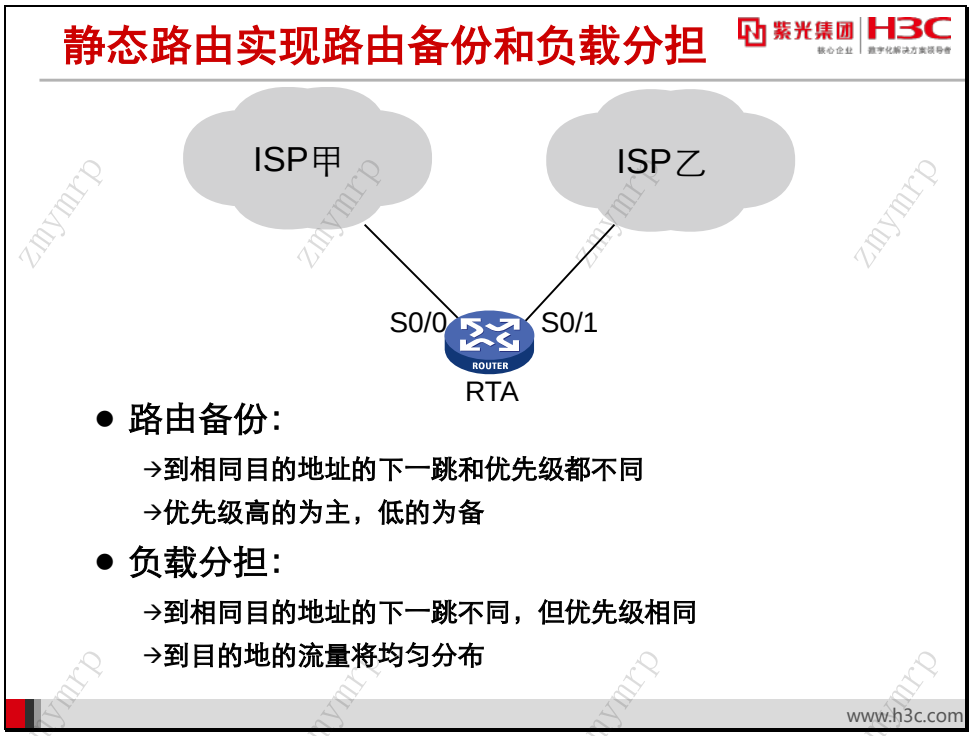
配置 RTD：

```
[RTD] ip route-static 0.0.0.0 0.0.0.0 10.4.0.1
```

所以，可以看到，默认路由在网络中是非常有用的。所以 Internet 上大约 99.99%的路由器上都存在一条默认路由！

zmymrp

22.6 用静态路由实现路由备份和负载分担



通过对静态路由优先级（preference）进行配置，可以灵活应用路由管理策略。如在配置到达网络目的地的多条路由时，若指定相同优先级，可实现负载分担；若指定不同优先级，则可实现路由备份。

如客户使用一台路由器连接到不同的 ISP，如想实现负载分担，则可配置 2 条默认静态路由，下一跳指向 2 个不同接口，使用默认的优先级，如下：

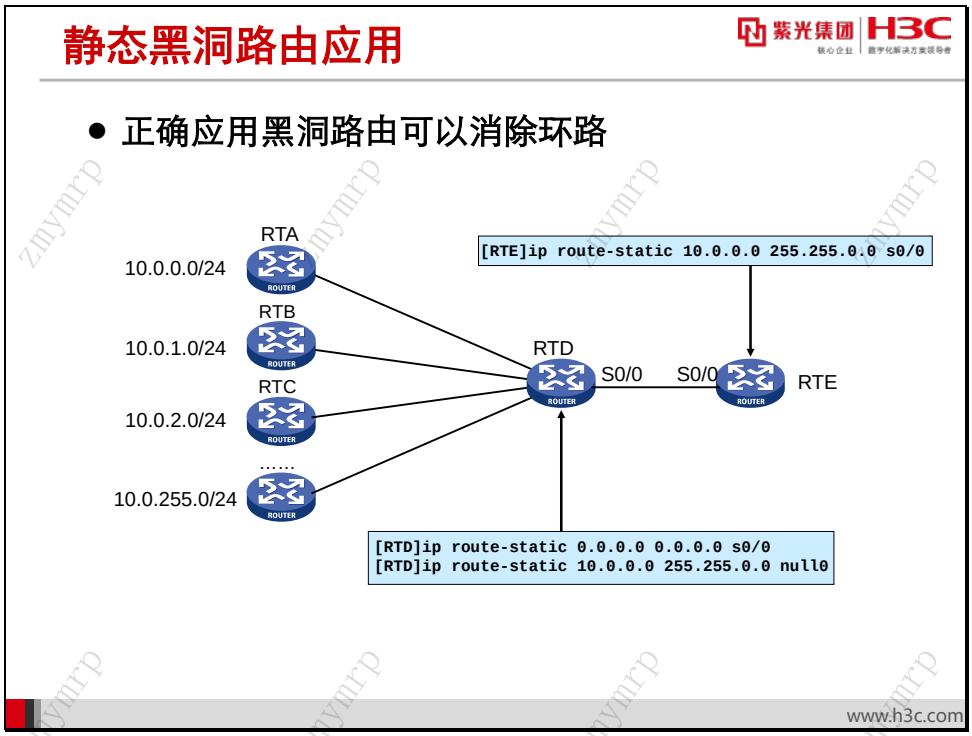
```
[RTA] ip route-static 0.0.0.0 0.0.0.0 serial 0/0
[RTA] ip route-static 0.0.0.0 0.0.0.0 serial 0/1
```

如想实现路由备份，则将其中一条路由的优先级改变。如想让连接到 ISP 甲的线路为主线路，则可以降低到达 ISP 甲的静态路由优先级，如下：

```
[RTA] ip route-static 0.0.0.0 0.0.0.0 serial 0/0 preference 10
[RTA] ip route-static 0.0.0.0 0.0.0.0 serial 0/1
```

因为到 ISP 甲的路由优先级为 10，低于到 ISP 乙的路由优先级，所以数据包被优先转发到 ISP 甲。如果网络产生故障，如 Serial 0/0 物理接口断开，即意味着路由表中下一跳失效，路由器会自动选择下一跳为 Serial 0/1 的路由，数据包被转发到 ISP 乙。

22.7 静态黑洞路由的应用



在配置静态路由时，对应接口可以配置为 NULL0。NULL 接口是一个特别的接口，我们无法在 NULL 接口上配置 IP 地址，路由器会提示配置非法。一个没有 IP 地址的接口能够做什么用呢？此接口单独使用没有意义，但是在一些网络中正确使用能够避免路由环路。

上图是一种常见的网络规划方案。RTD 连接有很多台小路由器，RTA、RTB、RTC 等，每台小路由器配置有默认路由，指向 RTD；相应 RTD 配置有到 10.0.0.0/24、10.0.1.0/24、10.0.2.0/24 等静态路由，回指到 RTA、RTB、RTC 等；同时为了节省路由表空间，RTD 上配置有一条默认路由指向 RTE。由于这些小路由器所连接的网段很有规律，恰好可以聚合成一条 10.0.0.0/16 的路由，于是路由器 RTE 上配置到 10.0.0.0/16 的静态路由，指向 RTD。

上述网络在正常情况下可以很好的运行，但如果出现如下情况时：

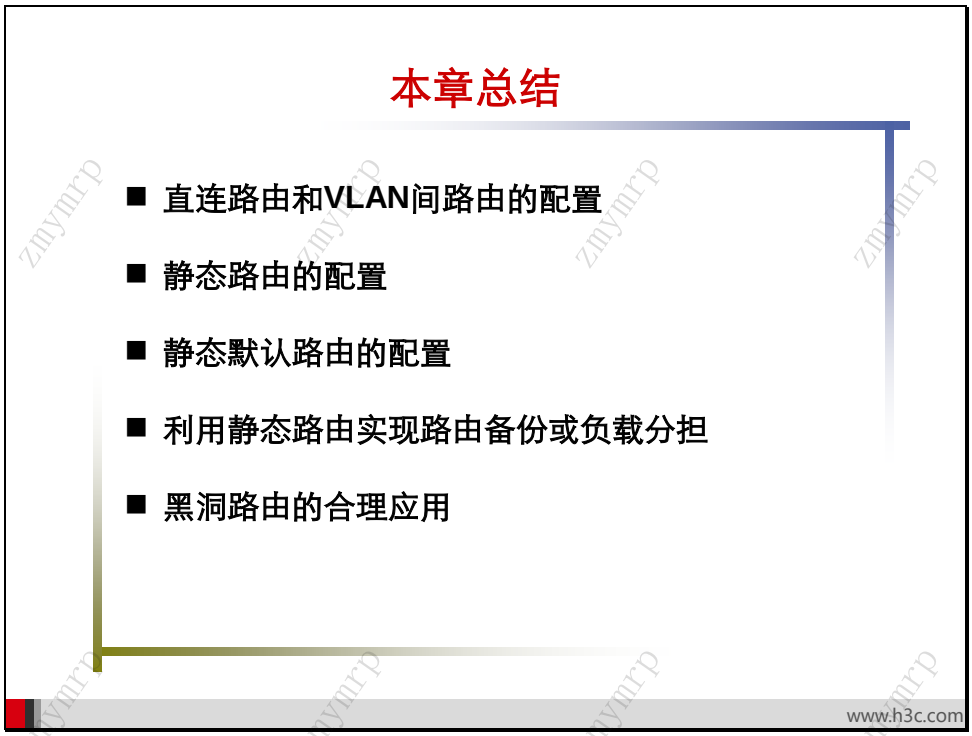
RTC 到 RTD 之间的链路由于故障中断，所以在 RTD 上去往 10.0.2.0/24 的指向 RTC 的路由失效。此时，如果 RTA 所连接网络中的一个用户发送报文，目的地址为 10.0.2.1，则 RTA 将此报文发送到 RTD，由于 RTD 上 10.0.2.0/24 的路由失效，所以选择默认路由，将报文发送给 RTE，RTE 查询路由表后发现该条路由匹配 10.0.0.0/16，于是又将该报文发送给 RTD。同理，RTD 会再次将报文发给 RTE，此时，在 RTD 和 RTE 上就会产生路由自环。

解决上述问题的最佳方案就是，在 RTD 上配置一条黑洞路由：

```
ip route-static 10.0.0.0 255.255.255.0 null 0，
```

这样，如果再发生上述情况，RTD 就会查找路由表，并将报文发送到 NULL0 接口（实际上就是丢弃此报文），从而避免环路的产生。

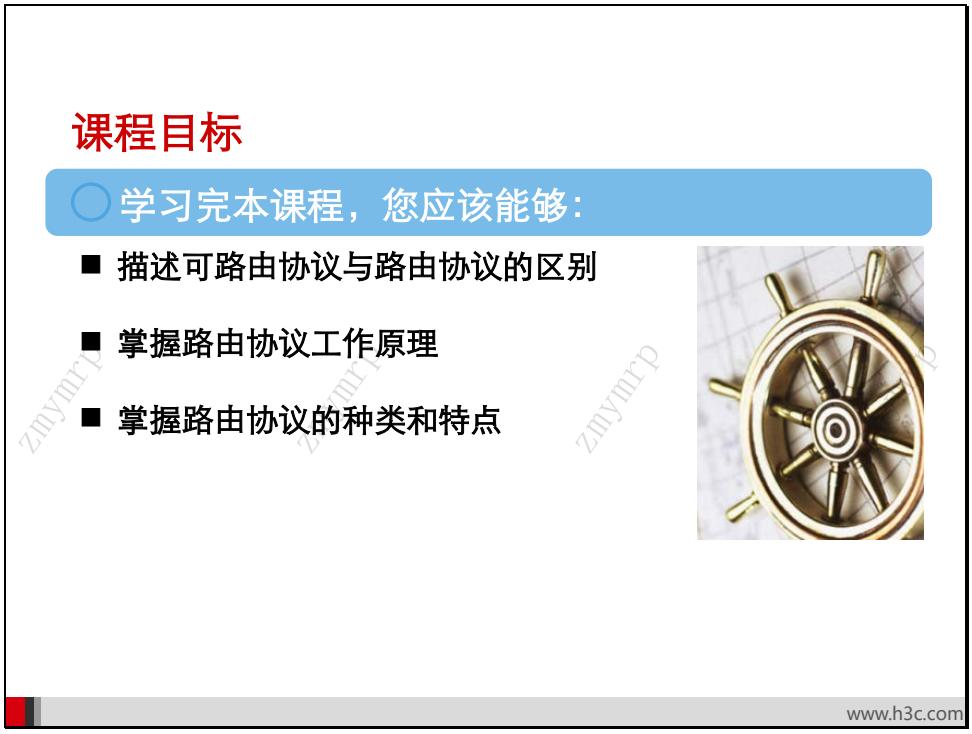
22.8 本章总结


本章总结
直连路由和VLAN间路由的配置
静态路由的配置
静态默认路由的配置
利用静态路由实现路由备份或负载分担
黑洞路由的合理应用
www.h3c.com

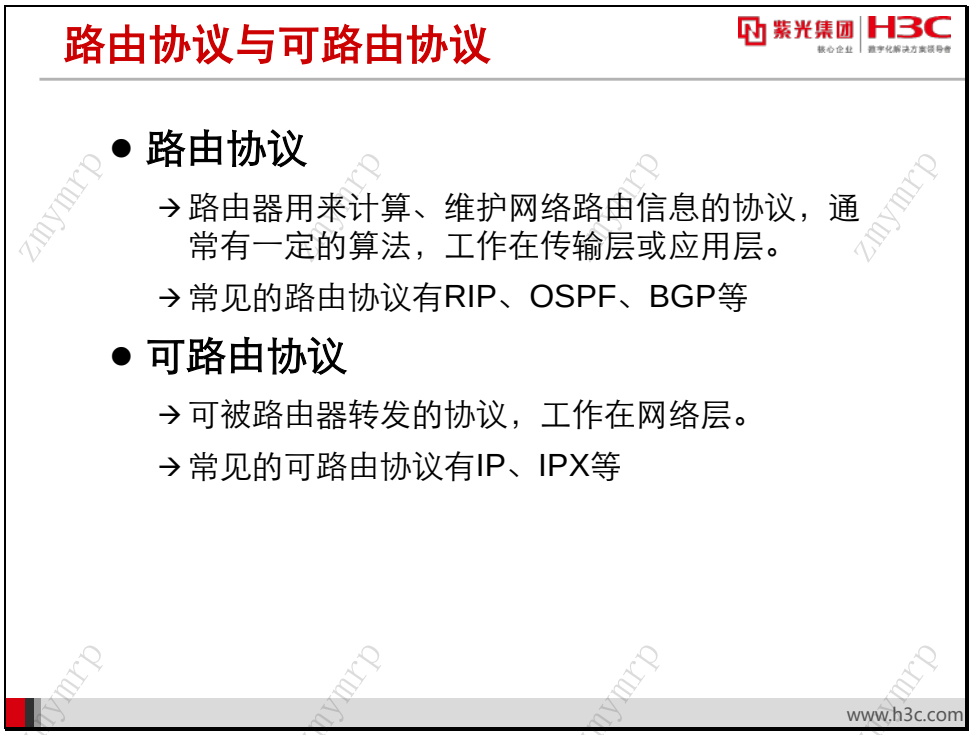
第23章 路由协议概述

路由可以静态配置，也可以通过路由协议来自动生成。路由协议能够自动发现和计算路由，并在拓扑变化时自动更新，无需人工维护，所以适用于复杂的网络中。

23.1 本章目标



23.2 可路由协议与路由协议

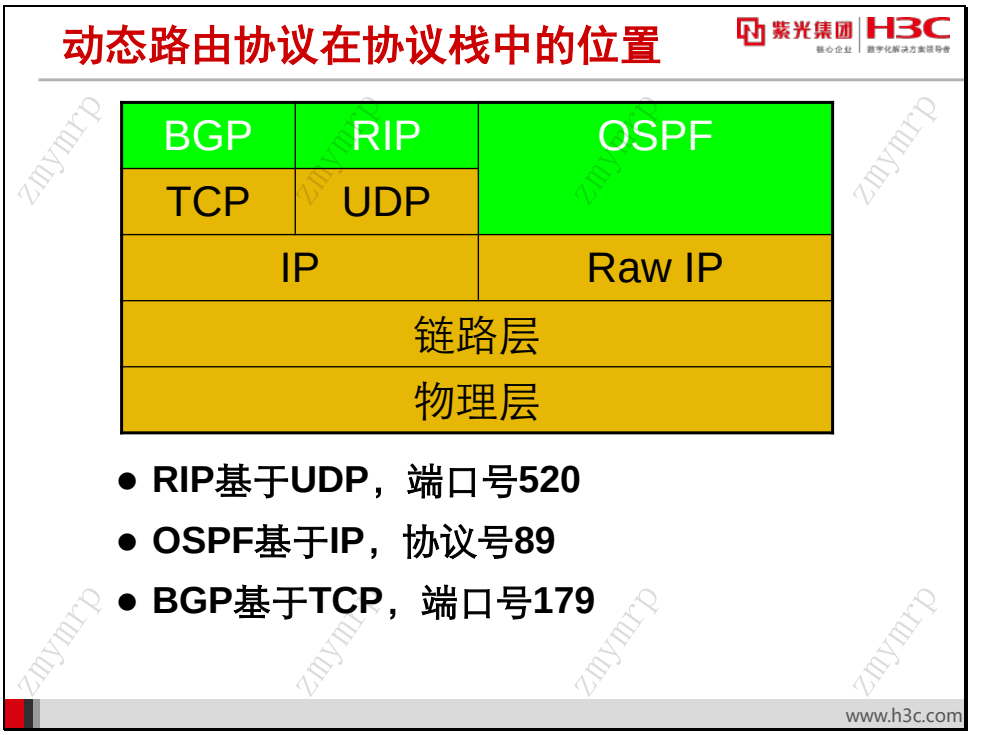


路由协议（routing protocol），简单来说就是用来计算、维护路由信息的协议。路由协议通常采用一定的算法，以产生路由；并有一定的方法确定路由的有效性，来维护路由。

可路由协议（routed protocol）又称为被路由协议，指可以被路由器在不同逻辑网段间路由的协议。比如 IP 协议、IPX/SPX 协议等。可路由协议通常工作在 OSI 模型的网络层，定义了数据包内各字段的格式和用途，其中包括网络地址，路由器可根据数据包内的网络地址对数据包进行转发。

23.3 路由协议概述

23.3.1 路由协议在协议栈中的位置



各种路由协议使用的底层协议各有不同。

OSPF 将协议报文直接封装在 IP 报文中，协议号 89，由于 IP 协议本身是不可靠传输协议，所以 OSPF 传输的可靠性需要协议本身来保证。

BGP 使用 TCP 作为传输协议，提高了协议的可靠性，TCP 的端口号是 179。

RIP 使用 UDP 作为传输协议，端口号 520。

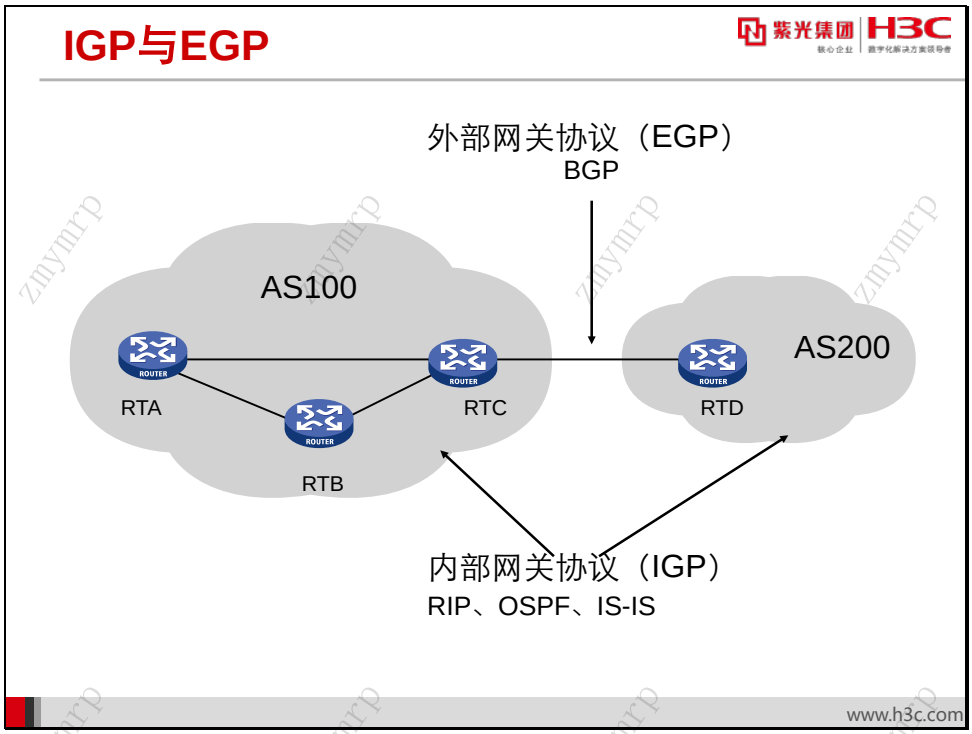
23.3.2 路由协议的基本原理

为了能在路由器之间交换路由信息，需要路由器运行相同的路由协议。每种路由协议都有自己的语言(相应的路由协议报文),如果两台路由器都实现了某种路由协议并已经启动该协议，则具备了相互通信的基础。

各种路由协议所共同的目的是计算与维护路由。通常，各种动态路由协议的工作过程包含以下几个阶段：

- 邻居发现：运行了某种路由协议的路由器会主动把自己介绍给网段内的其它路由器。具体方式既可以是广播发送路由协议消息，也可以是单播将路由协议报文发送给指定邻居路由器。
- 交换路由信息：发现邻居后，每台路由器将自己已知的路由相关信息发给相邻的路由器，相邻路由器又发送给下一台路由器。这样经过一段时间，最终每台路由器都会收到网络中所有的路由信息。
- 计算路由：每一台路由器都会运行某种算法，计算出最终的路由来。(实际上需要计算的是该条路由的下一跳和度量值)
- 维护路由：为了能够感知突然发生的网络故障（设备故障或线路中断等），路由协议规定两台路由器之间的协议报文应该周期性地发送。如果路由器有一段时间收不到邻居发来的协议报文，则认为邻居失效了。

23.3.3 路由协议的分类



按照工作范围的不同，路由协议可以分为 IGP 和 EGP：

- IGP（Interior Gateway Protocols）内部网关协议

IGP 是指在同一个自治系统内交换路由信息的路由协议。RIP、OSPF 和 IS-IS 属于 IGP。IGP 的主要目的是发现和计算自治系统内的路由信息。

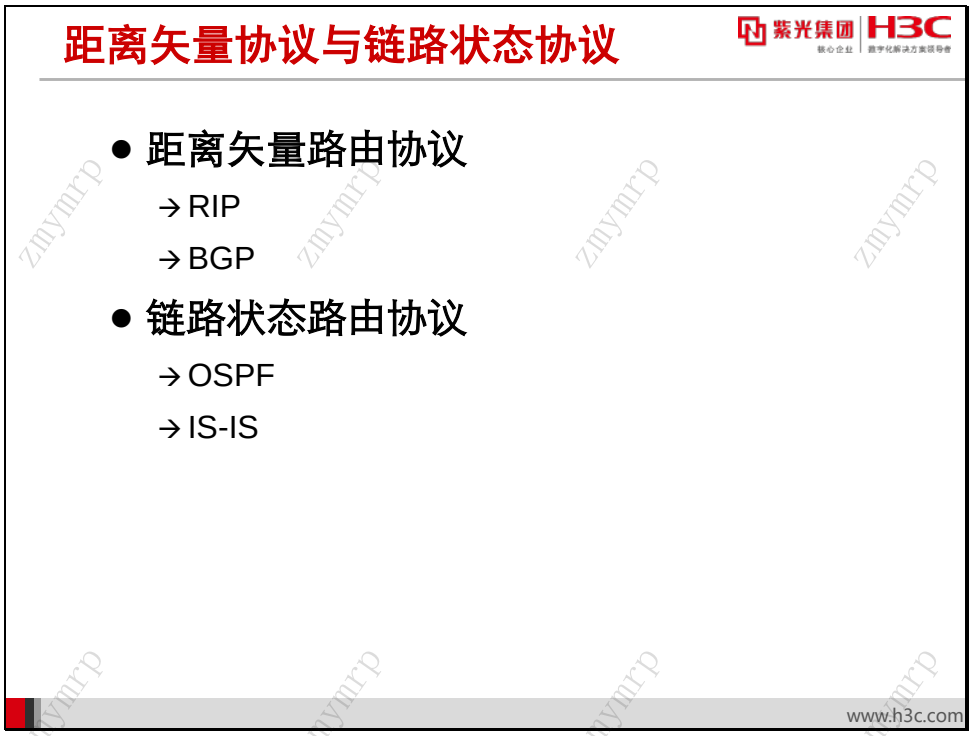
- EGP（Exterior Gateway Protocols）外部网关协议

与 IGP 不同，EGP 用于连接不同的自治系统，并在不同自治系统间交换路由信息。EGP 的主要目的是使用路由策略和路由过滤等手段控制路由信息在自治系统间的传播。BGP（Border Gateway Protocols，边界网关协议）属于 EGP。

自治系统（AS，Autonomous System）是一组共享相似的路由策略并在单一管理域中运行的路由器的集合。一个 AS 可以是一些运行单一 IGP 协议的路由器集合，也可以是一些运行不同路由选择协议但都属于同一个组织机构的路由器集合。不管是哪种情况，外部世界都将整个 AS 看作是一个实体。

每个自治系统都有一个唯一的自治系统编号，这个编号是由因特网授权的管理机构 IANA 分配的。它的基本思想就是希望通过不同的编号来区分不同的自治系统。这样，当网络管理员不希望自己的通信数据通过某个自治系统时，这种编号方式就十分有用了。例如，该网络管理员的网络完全可以访问某个自治系统，但由于它可能是由竞争对手在管理，或是缺乏足够的安全机制，因此可能需要回避它。通过采用路由协议和自治系统编号，路由器就可以确定彼此间的路径和路由信息的交换方法。

自治系统的编号范围是 1～65535，其中 1～64511 是注册的因特网编号，64512～65535 是专用网络编号。



按照路由的寻径算法和交换路由信息的方式，路由协议可以分为距离矢量（Distance-Vector，D-V）路由协议和链路状态（Link-State）路由协议。典型的距离矢量协议如 RIP，典型的链路状态协议如 OSPF。

距离矢量路由协议基于贝尔曼-福特算法。采用这种算法的路由器通常以一定的时间间隔向相邻的路由器发送路由更新。邻居路由器根据收到的路由更新来更新自己的路由，然后再继续向外发送更新后的路由。D-V 算法关心的是到目的网段的距离（Hops）和方向（从哪个接口转发数据）。

RIP 协议是一种典型的距离矢量路由协议。它的优点是配置简单，算法占用较少的内存和 CPU 处理时间。它的缺点是算法本身不能完全杜绝路由自环，收敛相对较慢，周期性广播路由更新占用网络带宽较大，扩展性较差，最大跳数不能超过 16 跳。

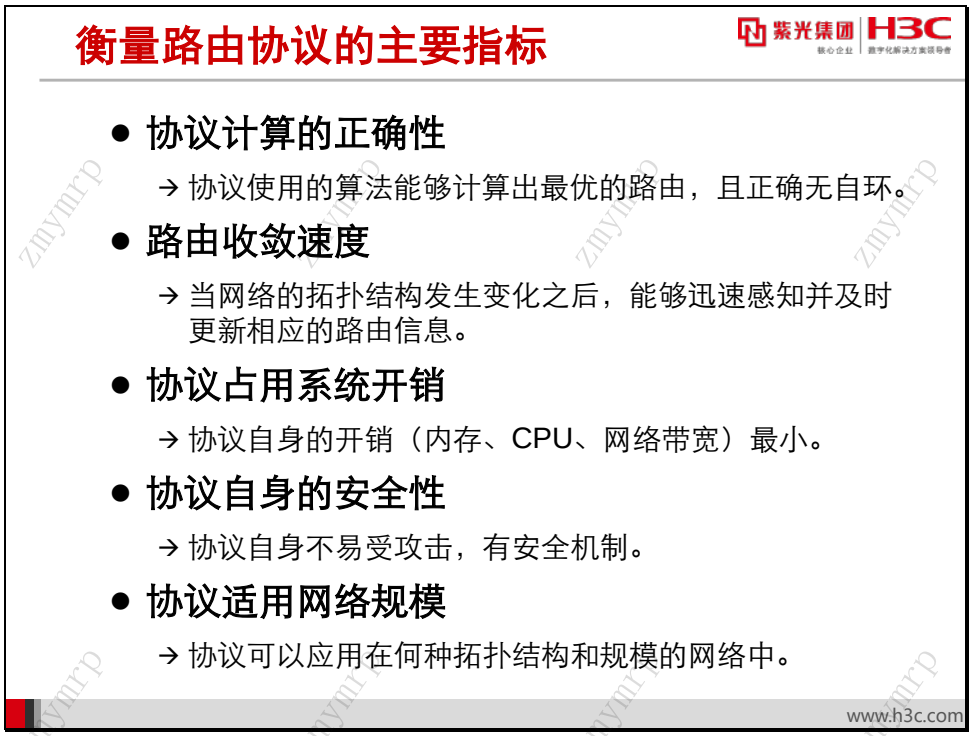
BGP 协议也是一种距离矢量路由协议，与 RIP 不同，BGP 采用一些方法能够防止路由环路，且采用增量更新机制来发送路由更新，只有当路由表变化时才发送路由更新信息，节省了相邻路由器之间的链路带宽。

链路状态路由协议基于 Dijkstra 算法，也称为最短路径优先算法。最短路径优先算法提供比 RIP 等 D-V 算法更大的扩展性和更快的收敛速度，但是它的算法耗费更多的路由器内存和 CPU 处理能力。Dijkstra 算法关心网络中链路或接口的状态（up 或 down、IP 地址、掩码），每个路由器将自己已知的链路状态向该区域的其他路由器通告，这些通告称为链路状态通告。通过这种方式区域内的每台路由器都建立了一个本区域的完整的链路状态数据库。然后路由器

根据收集到的链路状态信息来创建它自己的网络拓扑图，形成一个到各个目的网段的加权有向图。

链路状态算法使用增量更新的机制，只有当链路的状态发生了变化时才发送路由更新信息。

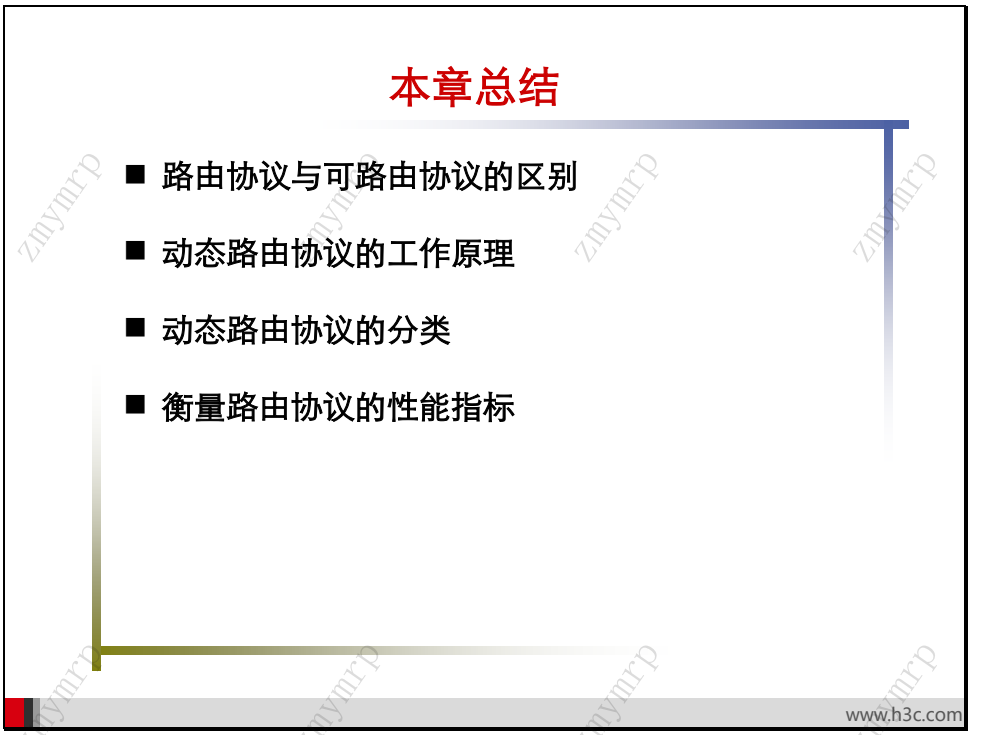
23.4 衡量路由协议的主要指标



路由协议的性能指标主要体现在以下几个方面：

- 协议计算的正确性：主要指路由协议所采用的算法会不会可能产生错误的路由而导致自环。不同路由协议所采用的算法不同，所以其正确性也不相同。总体来说，链路状态算法协议如 OSPF 在算法上杜绝了产生路由环的可能性，所以此项指标上占优。
- 路由收敛速度：路由收敛是指全网中路由器的路由表达到一致。收敛速度快，意味着在网络拓扑发生变化时，路由器能够更快的感知并及时更新相应的路由信息。OSPF、BGP 等协议的收敛速度要快于 RIP。
- 协议所占用的系统开销：路由器在运行路由协议时，需要消耗系统资源，如 CPU、内存等。因为工作原理的不同，各路由协议对系统资源的需求也不同。例如 OSPF 路由计算所需系统资源要大于 RIP 协议。
- 协议自身的安全性：协议安全性是指协议设计时有没有考虑防止攻击。OSPF、RIPv2 有相应的防止协议攻击的认证方法，而 RIPv1 没有。
- 协议适用网络规模：不同路由协议所适用的网络规模、拓扑不同。因为 RIP 协议在设计时有 16 跳的限制，所以应该应用在较小规模网络中；而 OSPF 可以应用在多达几百台路由器的大规模网络中；BGP 能够管理全世界所有的路由器，其所能管理的网络规模大小只受系统资源的限制。

23.5 本章总结

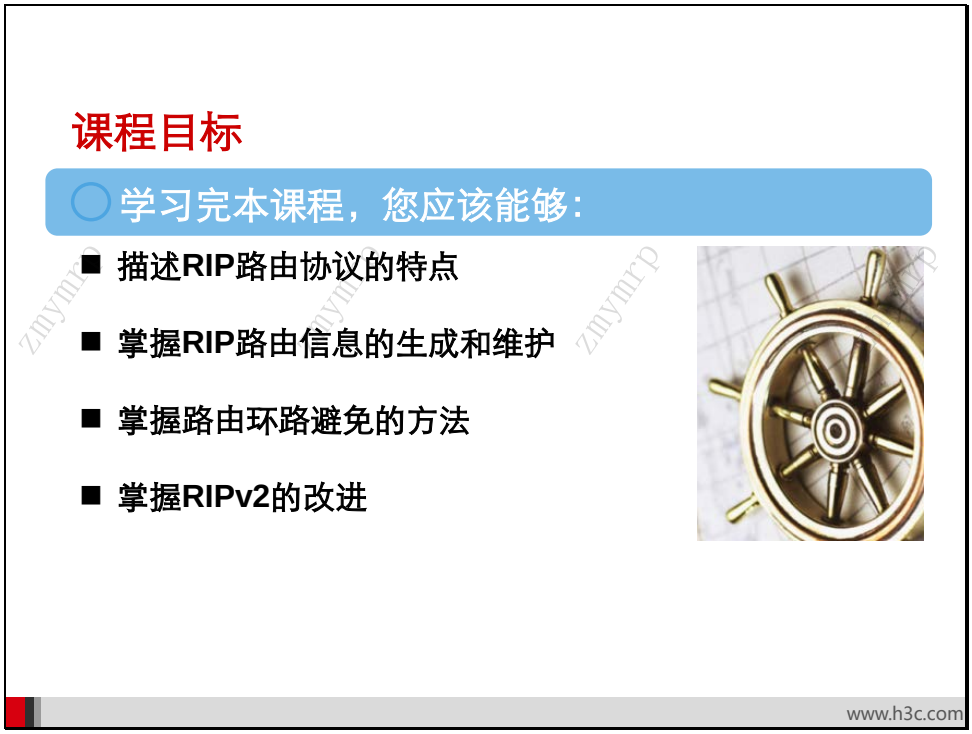

本章总结
路由协议与可路由协议的区别
动态路由协议的工作原理
动态路由协议的分类
衡量路由协议的性能指标
www.h3c.com

第24章 RIP 原理

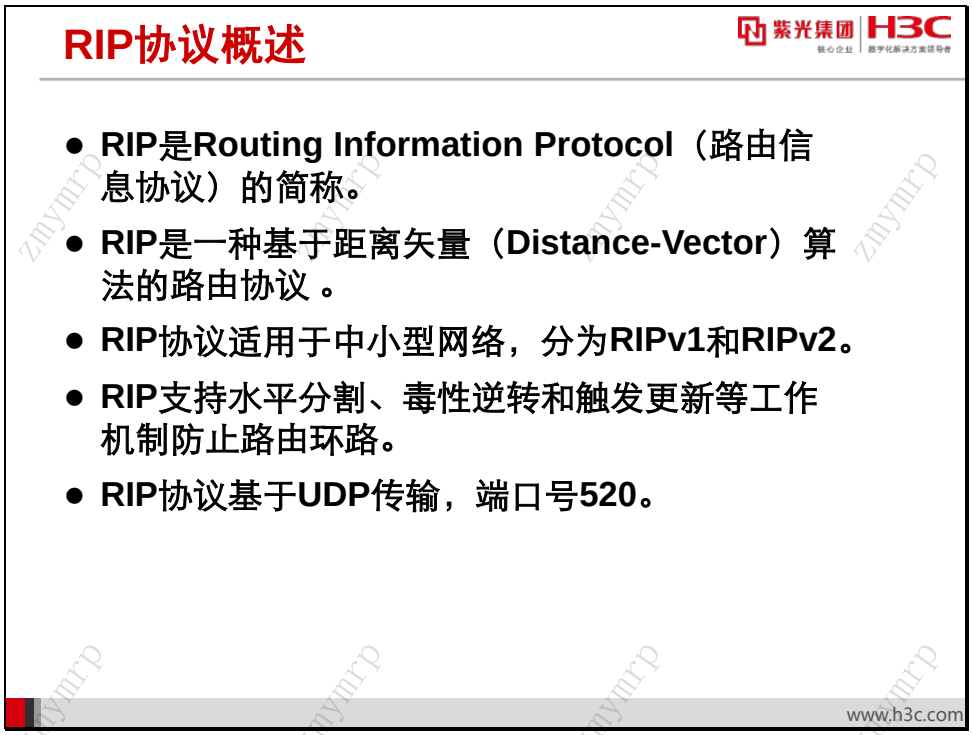
动态路由协议能够自动发现路由，计算路由。最早的动态路由协议是 RIP（Routing Information Protocol，路由信息协议），其原理简单，配置容易。

本章首先介绍了 RIP 路由协议的特点和分类，然后介绍 RIP 协议产生和维护路由信息的工作原理，并重点介绍 RIP 路由环路产生的原因和避免的方法。

24.1 本章目标



24.2 RIP路由协议概述



RIP（Routing Information Protocol，路由信息协议）是一种较为简单的内部网关协议，主要用于规模较小的网络中，比如校园网以及结构较简单的地区性网络。由于 RIP 的实现较为简单，在配置和维护管理方面也远比 OSPF 和 IS-IS 容易，因此在实际组网中有广泛地应用。

RIP 是一种基于距离矢量（Distance-Vector）算法的路由协议。RIP 使用跳数（Hop Count）来衡量到达目的网络的距离。在 RIP 中，路由器到与它直接相连网络的跳数为 0，通过与其直接相连的路由器到达下一个紧邻的网络的跳数为 1，其余依此类推，每多经过一个网络，跳数加 1。为限制收敛时间，RIP 规定度量值取 0～15 之间的整数，大于或等于 16 的跳数被定义为无穷大，即目的网络或主机不可达。由于这个限制，使得 RIP 不适合应用于大型网络。

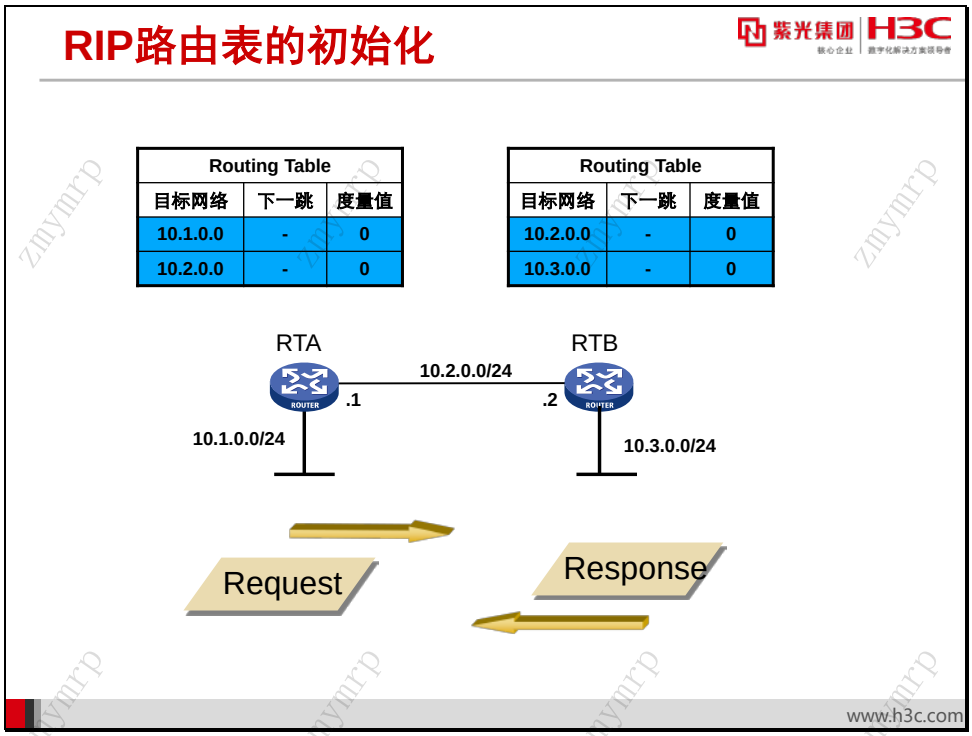
RIP 包括两个版本：RIPv1 和 RIPv2。RIPv1 是有类别路由协议，协议报文中不携带掩码信息，不支持 VLSM。RIPv1 只支持以广播方式发布协议报文。

RIPv2 支持 VLSM，同时 RIPv2 支持明文认证和 MD5 密文认证。

为防止产生路由环路，RIP 支持水平分割（Split Horizon）与毒性逆转（Poison Reverse），并在网络拓扑变化时采用触发更新（Triggered Update）来加快网络收敛时间。另外，RIP 协议还允许引入其它路由协议所得到的路由。

RIP 协议处于 UDP 协议的上层，通过 UDP 报文进行路由信息的交换，使用的端口号为 520。

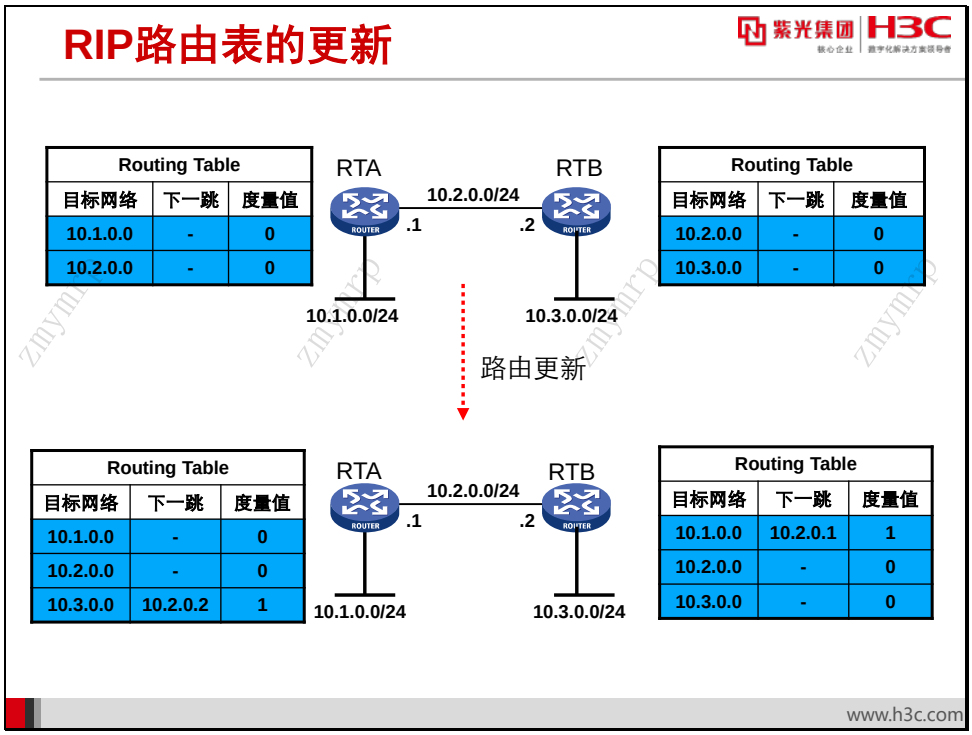
24.3 RIP协议的实现



未启动 RIP 的初始状态下，路由表中仅包含本路由器的一些直连路由。RIP 启动后，为了尽快从邻居获得 RIP 路由信息，RIP 协议使用广播方式向各接口发送请求报文（Request message），其目的是向 RIP 邻居请求路由信息。

相邻的 RIP 路由器收到请求报文后，响应该请求，回送包含本地路由表信息的响应报文（Response message）。

路由器收到响应报文后，查看响应报文中的路由，并更新本地路由表。



RIP 路由器收到响应报文后，更新本地路由表。路由表的更新原则是：

- 对本路由表中已有的路由项，当发送响应报文的 RIP 邻居相同时，不论响应报文中携带的路由项度量值增大或是减少，都更新该路由项（度量值相同时只将其老化定时器清零）；
- 对本路由表中已有的路由项，当发送响应报文的 RIP 邻居不同时，只在路由项度量值减少时，更新该路由项；
- 对本路由表中不存在的路由项，在度量值小于协议规定最大值（16）时，在路由表中增加该路由项。

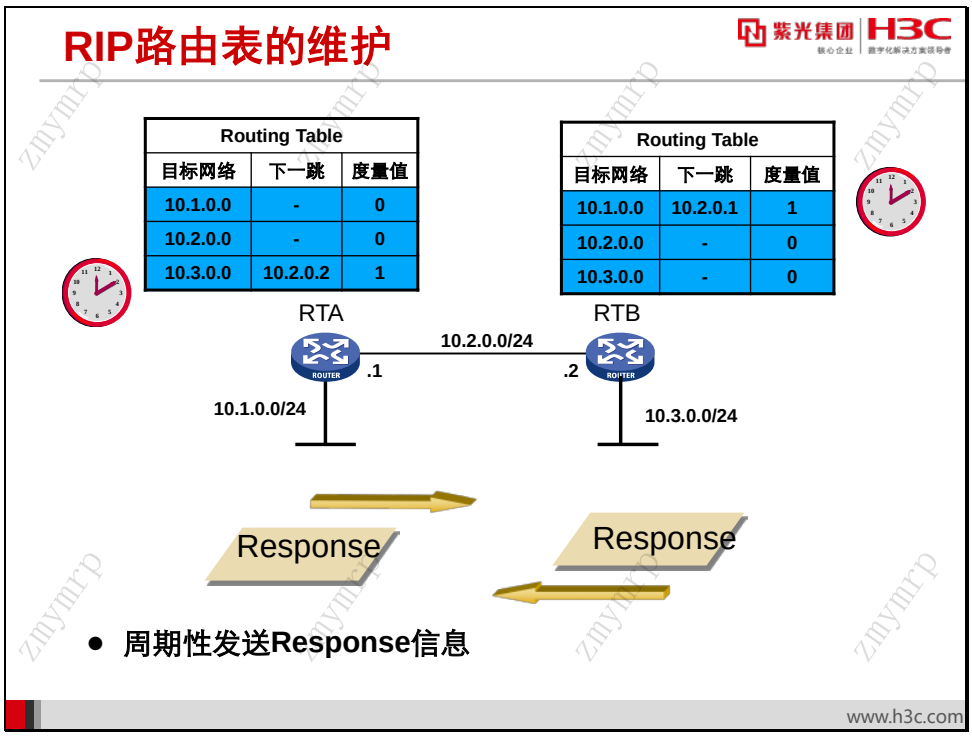
RIP 响应报文中携带有度量值（Metric），其值为路由表中的路由度量值加上发送附加度量值。

附加度量值是附加在 RIP 路由上的输入输出度量值，包括发送附加度量值和接收附加度量值。发送附加度量值不会改变路由表中的路由度量值，仅当接口发送 RIP 路由信息时才会添加到发送路由上，其默认值为 1；接收附加度量值会影响接收到的路由度量值，接口接收到一条 RIP 路由时，在将其加入路由表前会把度量值附加到该路由上，其默认值为 0。

根据以上规则，例图中 RTB 向 RTA 发送响应报文时，包含路由项 10.2.0.0 和 10.3.0.0，并计算出度量值为 1（原度量值 0 加上发送附加度量值 1）。RTA 从 RTB（10.2.0.2）接收到响应报文后，将响应报文中携带的路由项与本路由表中路由项比较，发现路由项 10.3.0.0 是本路

由表没有的，就把它增加到路由表中。添加时需要计算度量值，计算结果为 1（原度量值 1 加上接收附加度量值 0），并设置下一跳为 RTB（10.2.0.2）。

例中 RTB 响应报文中的路由项 10.2.0.0，因 RTA 路由表中路由项 10.2.0.0 是直连路由，所以 RTA 并不对其进行路由更新。



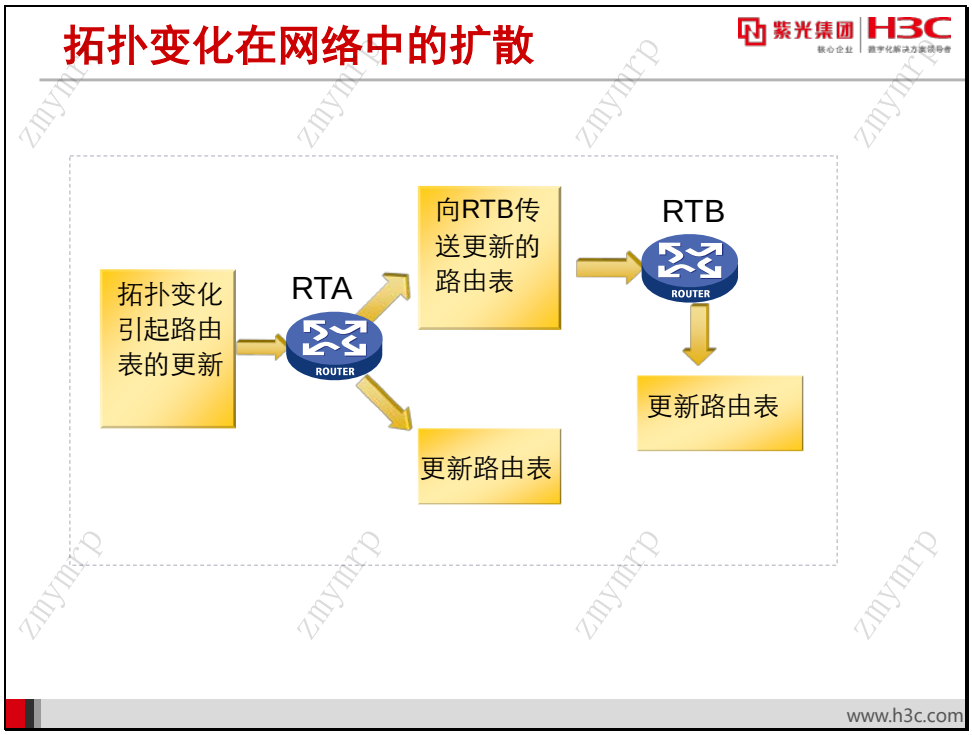
RIP 路由信息维护是由定时器来完成的：

- Update 定时器，定义了发送路由更新的时间间隔。默认值为 30 秒。
- Timeout 定时器，定义了路由老化时间。如果在老化时间内没有收到关于某条路由的更新报文，则该条路由的度量值将会被设置为无穷大（16），并从 IP 路由表中撤销。定时器默认值为 180 秒。
- Garbage-Collect 定时器，定义了一条路由从度量值变为 16 开始，直到它从路由表里被删除所经过的时间。如果 Garbage-Collect 超时，该路由仍没有得到更新，则该路由将被彻底删除。默认值为 120 秒。

例图中，路由器以 30 秒为周期用 Response 报文广播自己的路由表，称为周期性发送路由更新。如果路由器 RTA 经过 180 秒没有收到来自 RTB 的路由更新信息，则将路由表中的路由项 10.3.0.0 的度量值设为无穷大（16），并从 IP 路由表中撤销；若在其后 120 秒内仍未收到路由更新信息，就将路由 10.3.0.0 彻底删除。

注意：

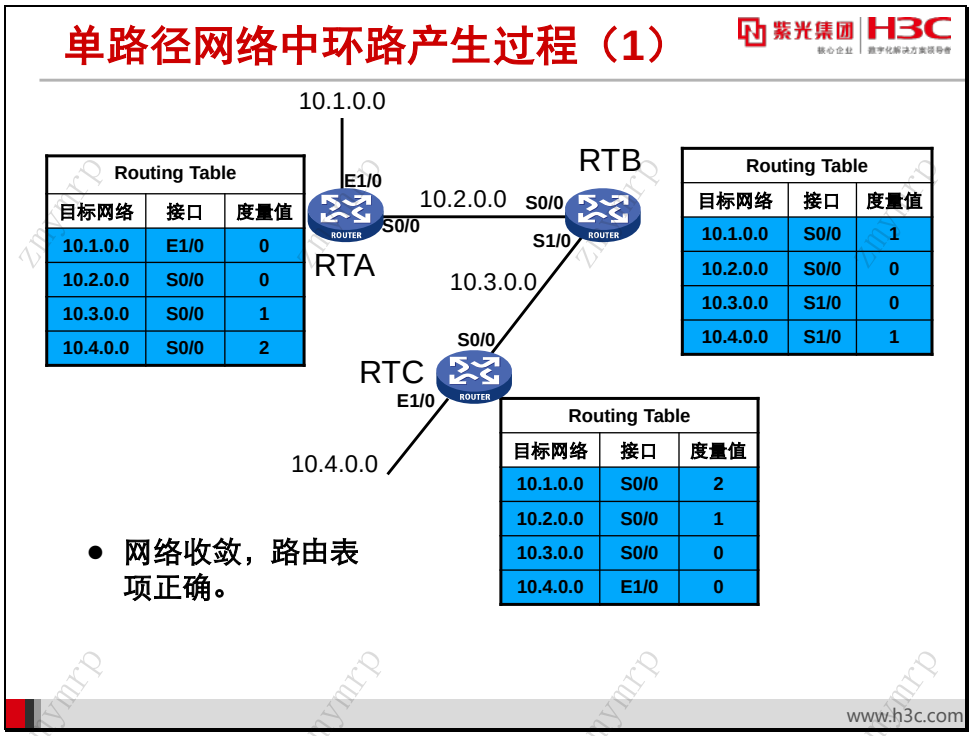
路由器对 RIP 协议维护一个单独的路由表，也称为 RIP 路由表。这个表中的有效路由会被添加到 IP 路由表中，作为转发的依据。从 IP 路由表中撤销的路由，可能仍然存在于 RIP 路由表中。



RIP 路由在网络中的扩散是逐跳进行的。当网络拓扑发生变化，如链路故障、新增加子网等，与变化所在地直连的路由器首先感知到变化，于是更新自己的路由表。在更新周期到来后，向邻居路由器以广播形式发送路由更新。邻居路由器收到更新后，根据更新规则更新本地路由表；然后在自身更新周期期满，再发送路由更新给自己的邻居路由器。以上拓扑的扩散过程是逐跳进行的，每台路由器仅负责通知自己的邻居。所以，路由器仅是通过传闻的方式获知路由，并不知道路由的确切来源。

拓扑发生变化的扩散过程需要一定的时间。图中 RTA 收到路由更新后，等待更新周期到来后向 RTB 发送路由更新；RTB 再向下一个邻居路由器发送，扩散时间取决于网络中路由器的数量和更新周期的长短。如果网络较大，更新周期长，则拓扑变化需要较长的时间才能通告到全网路由器。

24.4 单路径网络中路由环路产生与避免

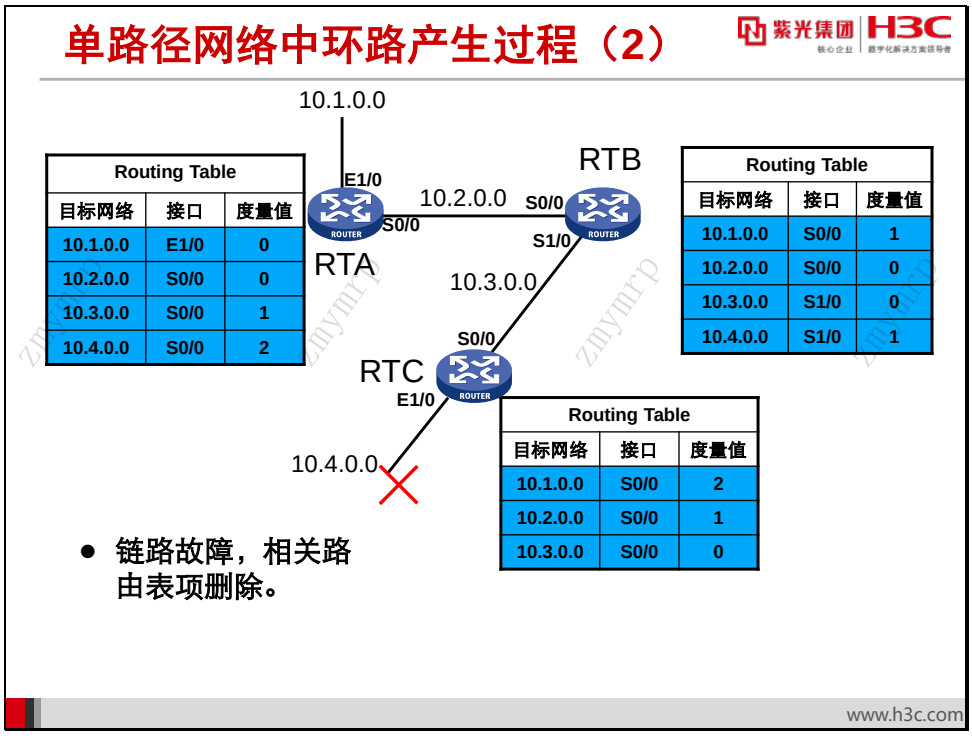


RIP 协议通过周期性地发送路由更新信息来维护路由信息，其基本维护原则如下：

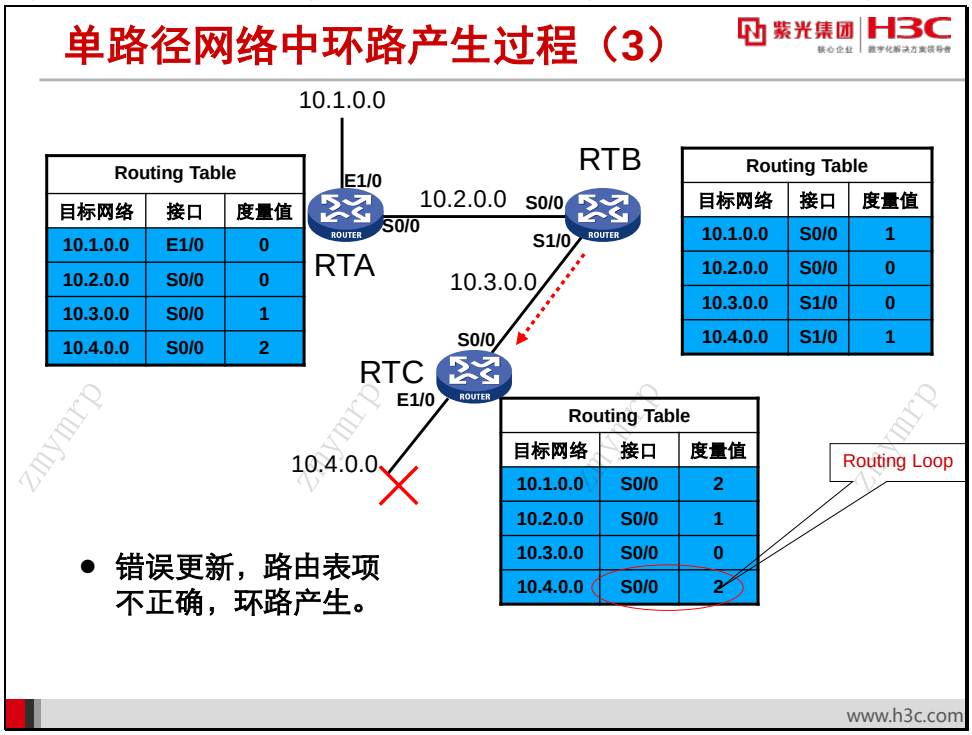
- 对本路由表中已有的路由项，当发送路由更新的 RIP 邻居相同时，不论路由更新中携带的路由项度量值增大或是减少，都更新该路由项；
- 对本路由表中不存在的路由项，在度量值小于无穷大（16）时，在路由表中增加该路由项。

由于 RIP 具有以上特点，在网络故障时可能会引起路由表信息与实际网络拓扑结构不一致，而发生路由环路现象。图中用示例来说明单路径网络中路由环路的产生过程。

如上图所示，在网络 10.4.0.0 发生故障之前，所有的路由器都具有正确一致的路由表，网络是收敛的。RIP 协议路由度量值是用跳数来计算。RTC 与网络 10.4.0.0 直连，所以 RTC 路由表中表项 10.4.0.0 的跳数是 0；RTB 通过 RTC 学习到路由项 10.4.0.0，其跳数为 1，接口为 S1/0。RTA 通过 RTB 学习到路由项 10.4.0.0，所以跳数为 2。



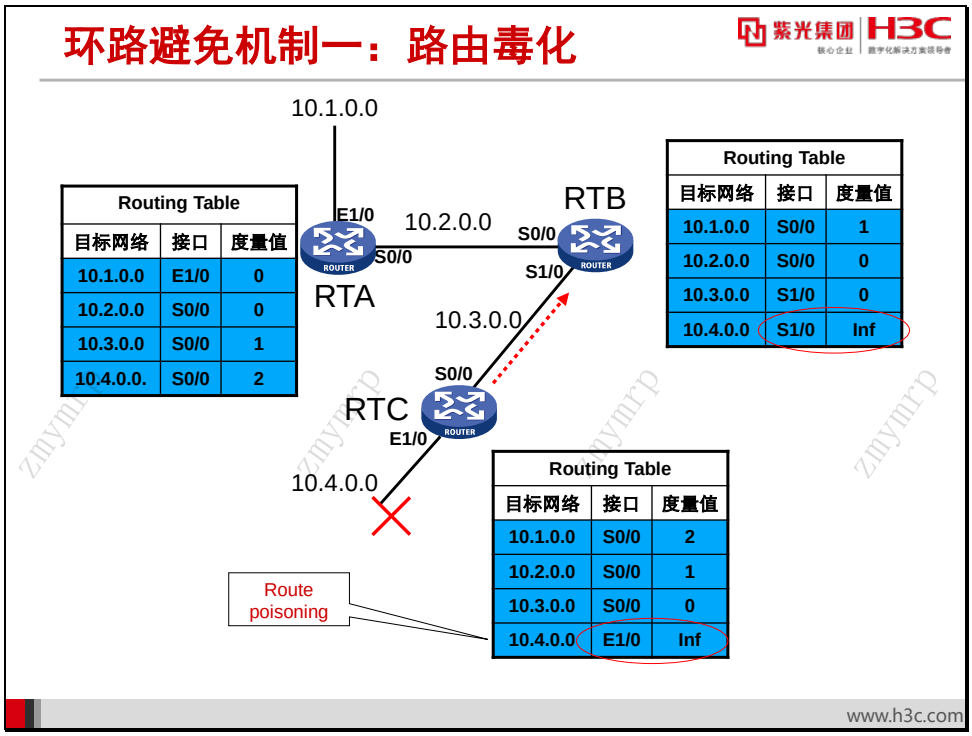
当网络 10.4.0.0 发生故障，直连路由器 RTC 最先收到故障信息，RTC 把网络 10.4.0.0 从路由表中删除，并等待更新周期到来后发送路由更新给相邻路由器。



根据 RIP 协议的工作原理，所有路由器都要周期性发送路由更新信息。所以，在 RTB 的路由更新周期到来后，RTB 发送路由更新，更新中包含了自己的所有路由。

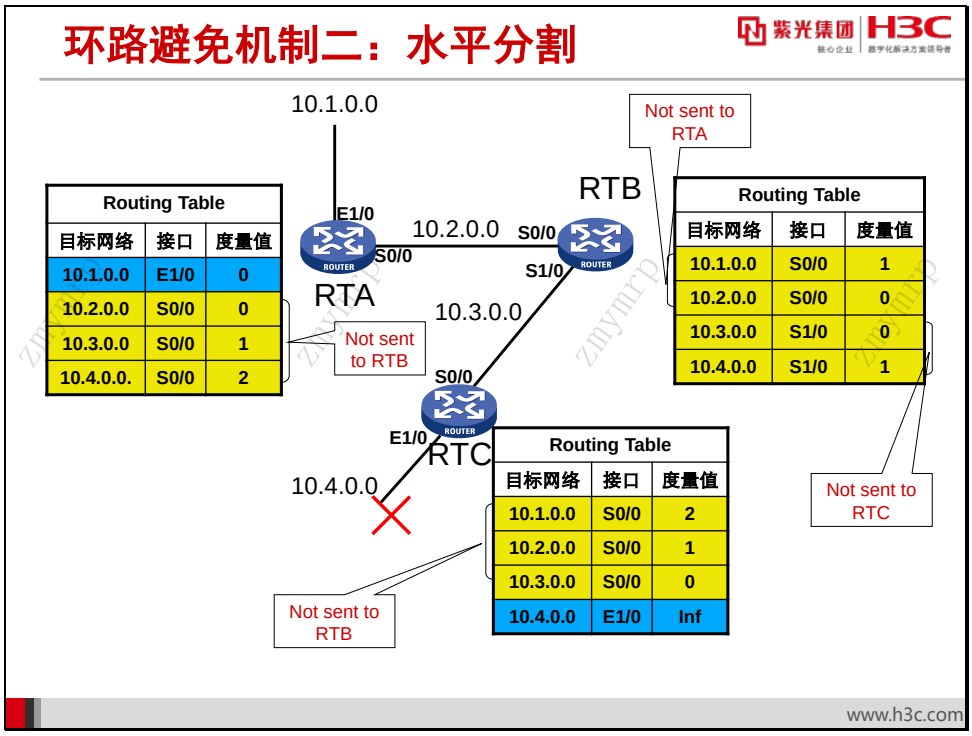
RTC 接收到 RTB 发出的路由更新后，发现路由更新中有路由项 10.4.0.0，而自己路由表中没有 10.4.0.0，就把这条路由项增加到路由表中，并修改其接口为 S0/0（因为是从 S0/0 收到更新消息），跳数为 2。这样，RTC 的路由表中就记录了一条错误路由（经过 RTB，可去往网络 10.4.0.0，跳数为 2 ）。

这样，RTB 认为可以通过 RTC 去往网络 10.4.0.0，RTC 认为可以通过 RTB 去往网络 10.4.0.0，就形成了环路。



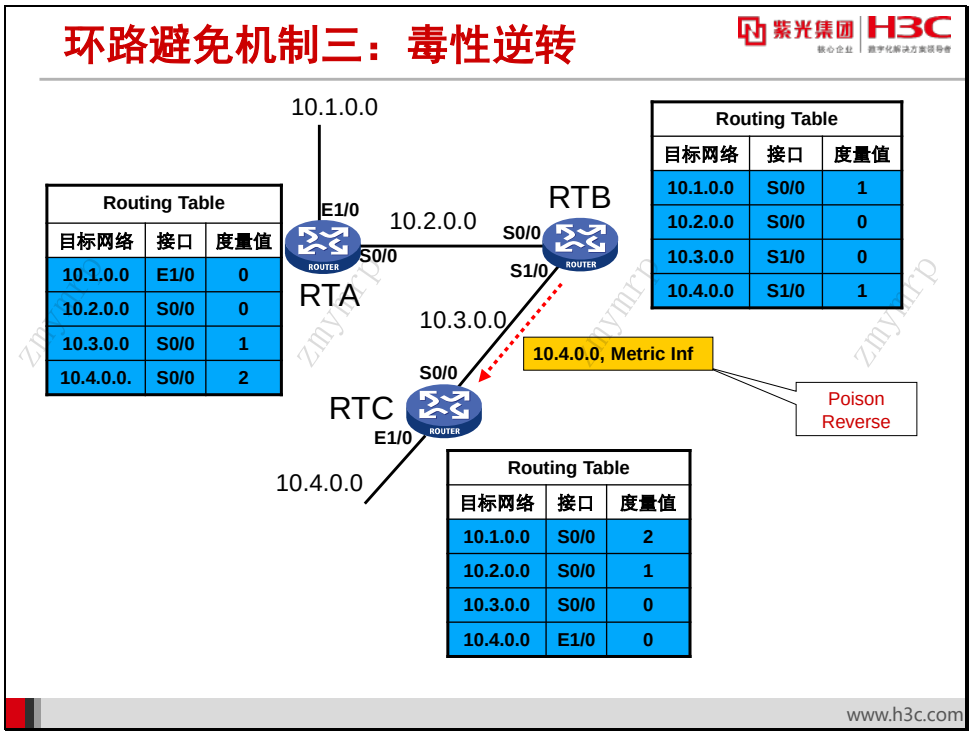
在前面路由环路生成的例子中，当网络 10.4.0.0 发生故障时，RTC 把路由项 10.4.0.0 从路由表中删除，以至于 RTB 不知道网络 10.4.0.0 发生故障。所谓路由毒化（Route Poisoning）就是路由器主动把路由表中发生故障的路由项以度量值无穷大（16）的形式通告给 RIP 邻居，以使邻居能够及时得知网络发生故障。

例图中，RTC 在路由更新信息里把路由项 10.4.0.0 的度量值置为无穷大，通告给 RTB。RTB 接收路由更新信息后，更新自己路由表，路由项 10.4.0.0 的度量值也置为无穷大。如此将网络 10.4.0.0 不可达的信息向全网扩散。



分析前述产生路由环路的原因，另外最重要的一条就是因为路由器将从某个邻居学到的路由信息又告诉了这个邻居。

水平分割（Split Horizon）是在距离矢量路由协议中最常用的避免环路发生的解决方案之一。水平分割的思想就是 RIP 路由器从某个接口学到的路由，不会再从该接口发回给邻居路由器。上图中，RTC 把它的直连路由 10.4.0.0 通告给 RTB，也就是 RTB 从 RTC 那里学习到了路由项 10.4.0.0，接口为 S1/0。在应用水平分割后，RTB 在接口 S1/0 上发送路由更新时，就不能包含路由项 10.4.0.0。

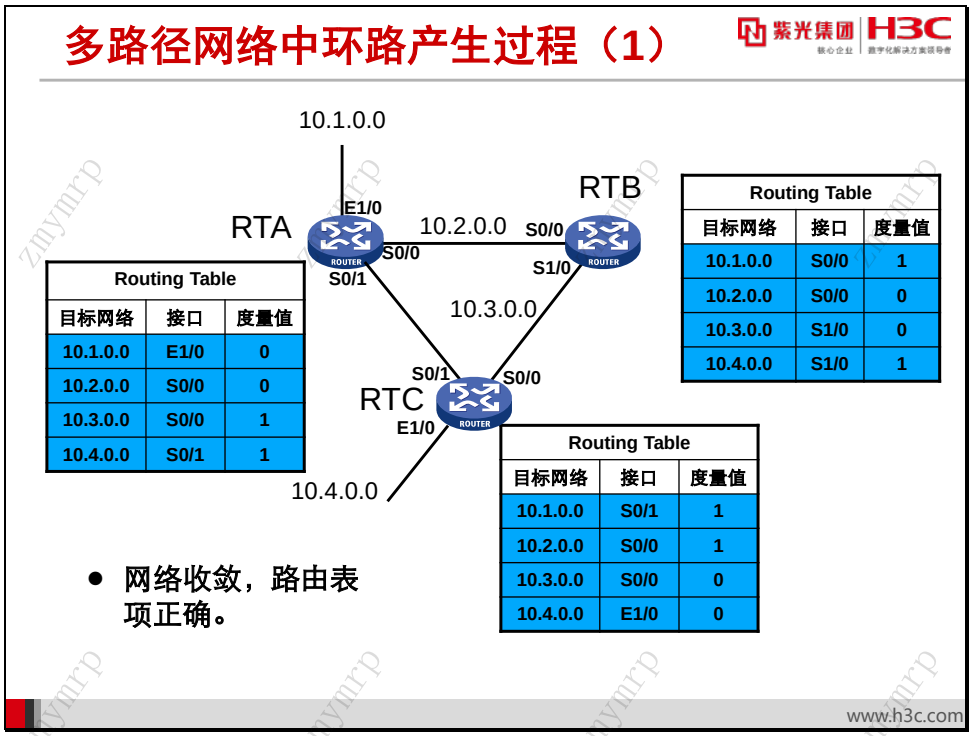


毒性逆转（Poison Reverse）是另一种避免环路的方法。毒性逆转是指，RIP 从某个接口学到路由后，将该路由的度量值设置为无穷大（16），并从原接口发回邻居路由器。

上图中，应用毒性逆转后，RTB 在发送路由更新给 RTC 时，更新中包含了路由 10.4.0.0，度量值为 16。相当于显式的告诉 RTC，不可能从 RTB 到达网络 10.4.0.0。

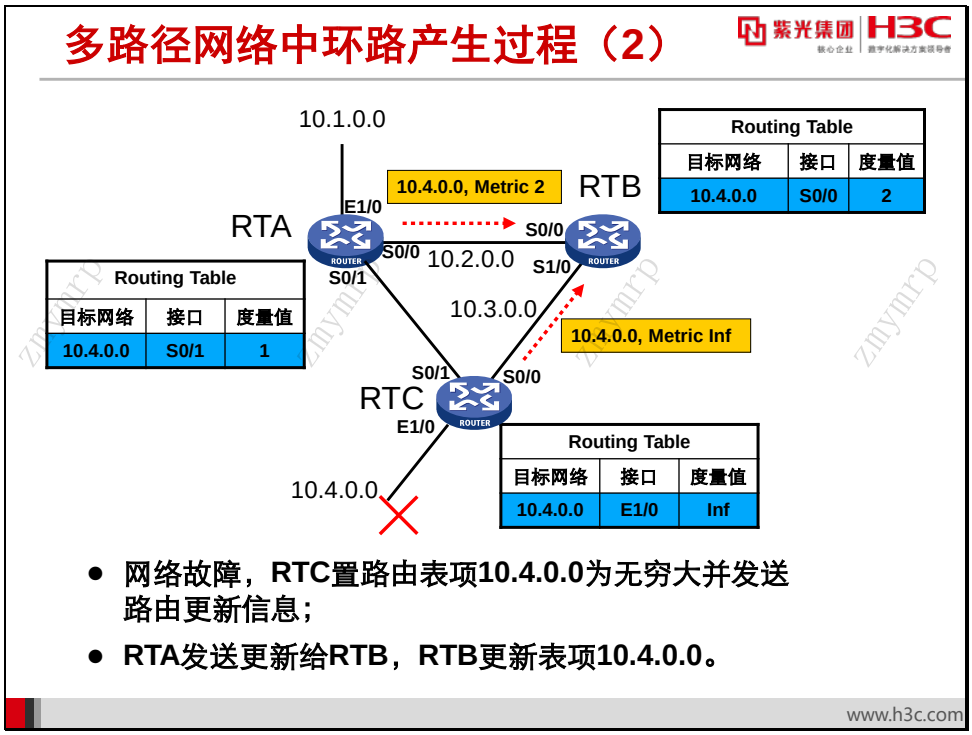
与水平分割相比，毒性逆转更加健壮和安全。因为毒性逆转是主动把网络不可达信息通知给其它路由器。毒性逆转的缺点是路由更新中路由项数量增多，浪费网络带宽与系统开销。

24.5 多路径网络中环路产生与避免

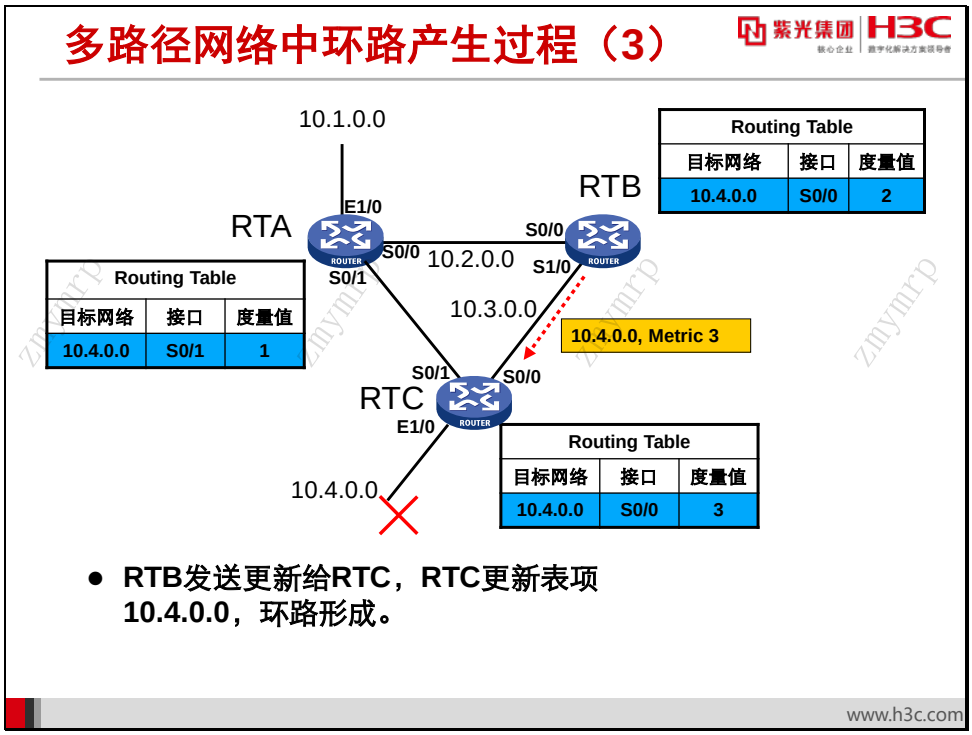


在简单的单路径网络环境中，水平分割能够较好的解决环路问题；但在多路径网络环境中，水平分割并不能阻止环路生成。

如上图，一个环形网络达成收敛，各路由器的路由表项均正确。

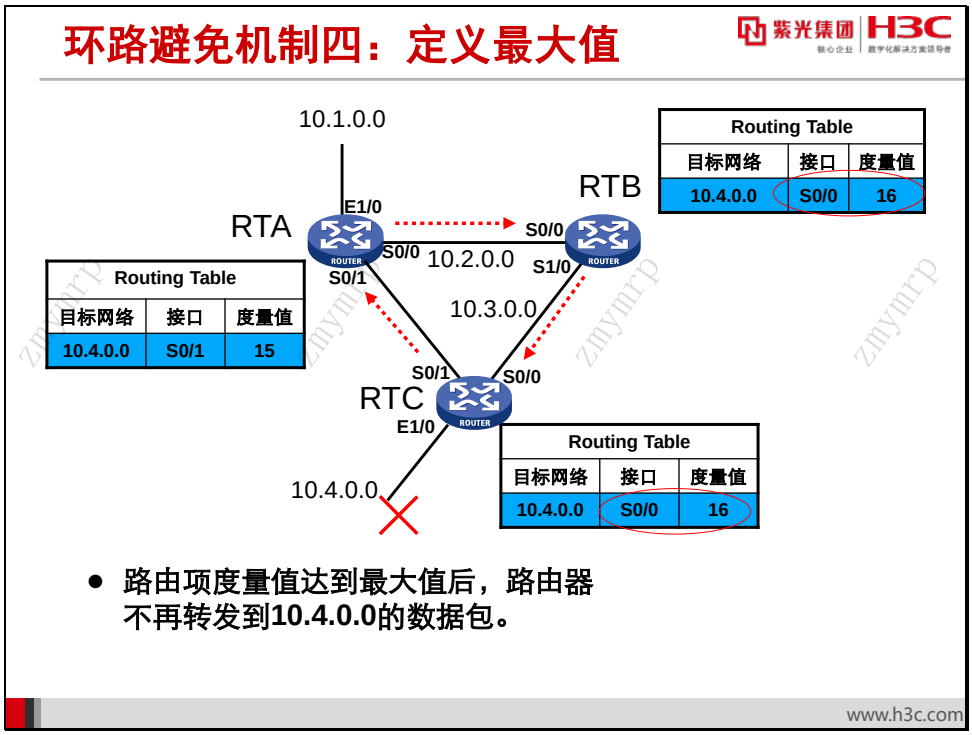


RTC 的接口产生故障后，RTC 会向邻居路由器 RTB 和 RTA 发送更新消息，告知 RTA 和 RTB 网络 10.4.0.0 经由 RTC 不再可达。但是，假设 RTB 已经收到 RTC 的更新，而在 RTC 的这个更新到达 RTA 之前，RTA 的更新周期恰巧到来，RTA 会向 RTB 发送路由更新，其中含有路由项 10.4.0.0，跳数为 2。RTB 收到 RTC 的路由更新后已经删除了 10.4.0.0 网段的路由，所以会以此路由更新为准，向自己的路由表中加入路由项 10.4.0.0，下一跳指向 RTA，跳数为 2。



随后，RTA 收到 RTC 的更新，删除了 10.4.0.0 的路由。

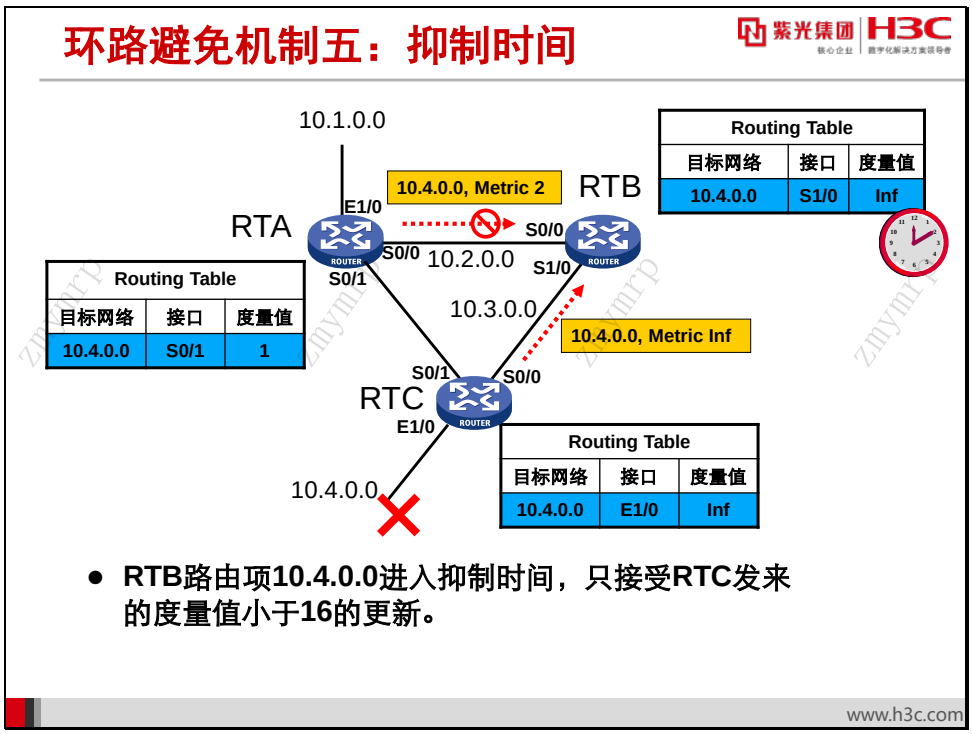
在 RTB 的更新周期到来后，RTB 会向 RTC 发送路由更新，RTC 据此更新自己路由表，改变路由项 10.4.0.0 的下一跳指向 RTB，跳数为 3，至此路由环路形成。RTC 也会向 RTA 发送路由更新，RTA 更新自己路由项 10.4.0.0，下一跳指向 RTC，跳数为 4。如此反复循环，每个路由器中路由项 10.4.0.0 的跳数不断增大，网络无法收敛。



为解决上述问题，我们给每种距离矢量路由协议度量值定义一个最大值。在 RIP 路由协议中，规定度量值是跳数，所能达到的最大值为 16。其实，在前面的例子中，我们已经使用跳数 16 来表示度量值的最大值了。

在图中，当跳数到达最大值 16 时，网络 10.4.0.0 被认为是不可达的。路由器会在路由表中显示网络不可达信息，并不再更新到达网络 10.4.0.0 的路由。路由器会丢弃去往网络 10.4.0.0 的数据包，不再转发。

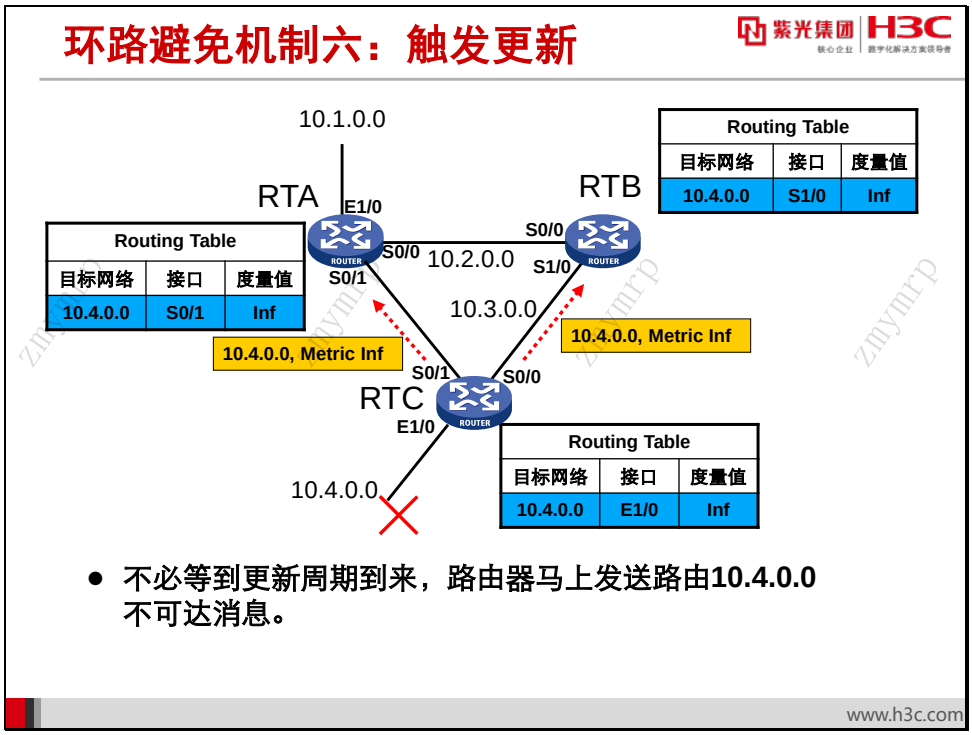
通过定义最大值，距离矢量路由协议可以解决发生环路时路由度量值无限增大的问题，同时也校正了错误的路由信息。但是，在最大度量值到达之前，路由环路还是会存在。也就是说，定义最大值只是一种补救措施，只能减少路由环路存在的时间，并不能避免环路的产生。



抑制时间与路由毒化结合使用，能够在一定程度上避免路由环路产生。抑制时间规定，当一条路由的度量值变为无穷大（16）时，该路由将进入抑制状态。在被抑制状态，只有来自同一邻居且度量值小于无穷大（16）的路由更新才会被路由器接收，取代不可达路由。

如上图所示：

- 当网络 10.4.0.0 发生故障时，RTC 毒化自己路由表中的路由项 10.4.0.0，使其度量值为无穷大（16），以表明网络 10.4.0.0 不可达。同时给路由项 10.4.0.0 设定抑制时间。在更新周期到来后，发送路由更新给 RTB。
- RTB 收到 RTC 发出的路由更新信息后，更新自己的路由项 10.4.0.0，同时启动抑制时间，在抑制时间结束之前的任何时刻，如果从同一相邻路由器 RTC 又接收到网络 10.4.0.0 可达的更新信息，路由器就将路由项 10.4.0.0 标识为可达，并删除抑制时间。
- 在抑制时间结束之前的任何时刻，如果接收到其他的相邻路由器如 RTA 的有关网络 10.4.0.0 的更新信息，路由器 RTB 会忽略此更新信息，不更新路由表。
- 抑制时间结束后，路由器如果收到任何相邻路由器发出的有关网络 10.4.0.0 的更新信息，路由器都将会更新路由表。



路由环路所产生的很重要的一个原因是 RIP 周期性的发送路由更新信息。如图中 RTC 得知网络 10.4.0.0 产生故障，但在更新周期未到时，RTC 不发送路由更新信息，在此期间 RTA、RTB 也就不知道网络 10.4.0.0 产生故障，RTA、RTB 就有可能在网络中传播网络 10.4.0.0 可达信息，从而造成环路。

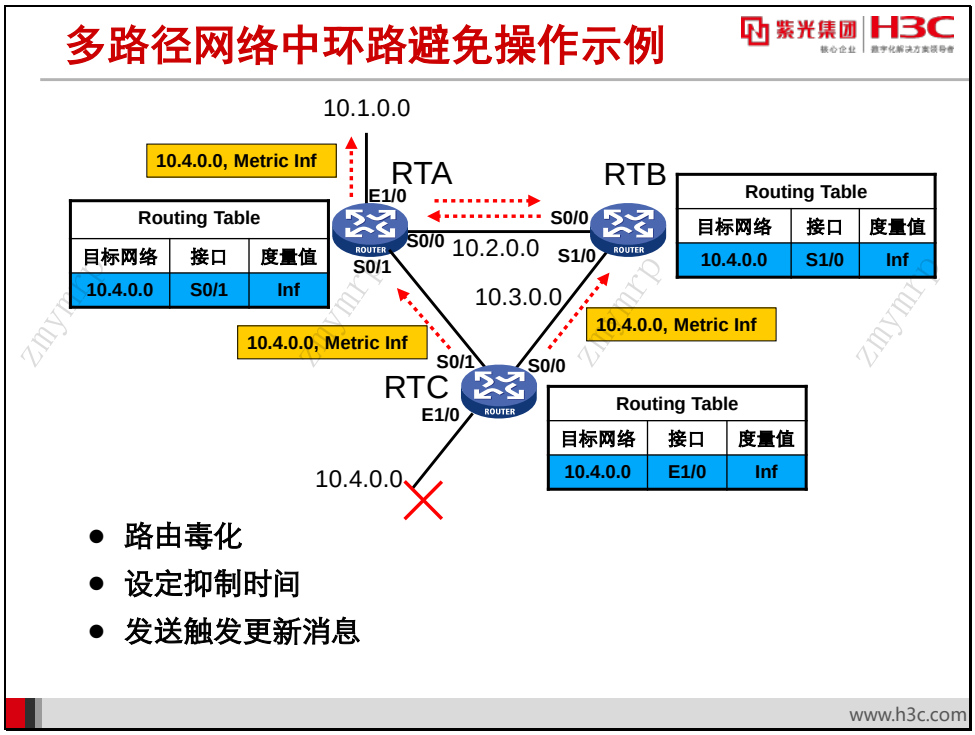
触发更新机制是指，当路由表中路由信息产生改变时，路由器不必等到更新周期到来，而立即发送路由更新给相邻路由器。在图中，当网络 10.4.0.0 产生故障后，RTC 不必等待更新周期到来，而是立即发送路由更新消息以通告网络 10.4.0.0 不可达信息，RTA、RTB 接收到这个信息后，也立即向邻居发送路由更新消息，这样，网络 10.4.0.0 不可达信息会很快传播到整个网络。

由以上可以看出，触发更新极大地加快了网络收敛。

使用触发更新方法能够在一定程度上避免路由环路发生。但是，仍然存在两个问题：

- 触发更新信息在传输过程中可能会被丢掉或损坏。
- 如果触发更新信息还没有来得及发送，路由器就接收到相邻路由器的周期性路由更新信息，使路由器更新了错误的路由信息。

抑制时间和触发更新相结合，就可以解决上述问题。在抑制时间内，路由器不理会从其他路由器传来的相关路由项可达信息，相当于确保路由项不可达信息不被错误的可达信息取代。



在实际网络中，上述的所有防止环路机制结合在一起使用，以最大可能的避免环路，加快网络收敛。

在上面的例子中，当网络 10.4.0.0 发生故障时，会有下面的情形发生：

1) 路由毒化。当 RTC 检测到网络 10.4.0.0 故障时，RTC 毒化路由表中路由项 10.4.0.0，使到此网络的跳数为无穷大。

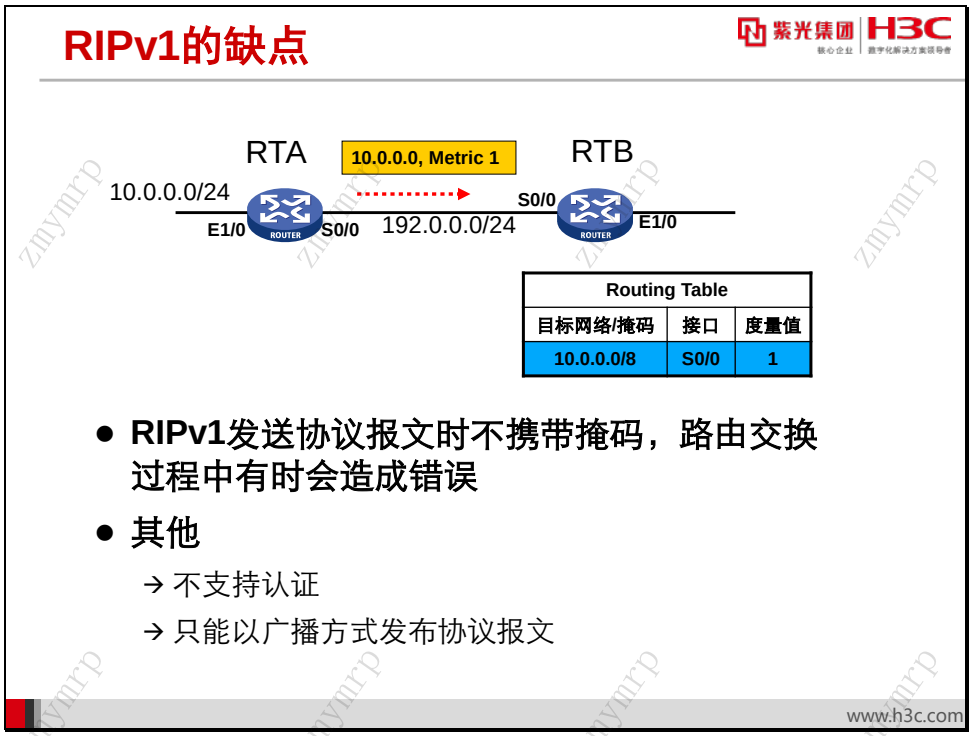
2) 设定抑制时间。RTC 给路由项 10.4.0.0 设定一个抑制时间。其默认值为 120 秒。

3) 发送触发更新信息。RTC 向 RTA、RTB 发送触发更新信息，指出网络 10.4.0.0 故障。RTA、RTB 接收到触发更新信息以后，使路由项 10.4.0.0 进入抑制状态，在抑制状态下不接受来自其它路由器的相关更新。然后，RTA 和 RTB 也向其他接口发送网络 10.4.0.0 故障的触发更新信息。

至此，全网所有路由器的路由表中，表项 10.4.0.0 的度量值均为无穷大，并且进入抑制状态，路由器会丢弃目的地为网络 10.4.0.0 的数据包。

网络 10.4.0.0 恢复正常后，RTC 解除抑制时间，同时用触发更新向 RTA、RTB 传播。RTA、RTB 也解除抑制时间，路由表恢复正常。

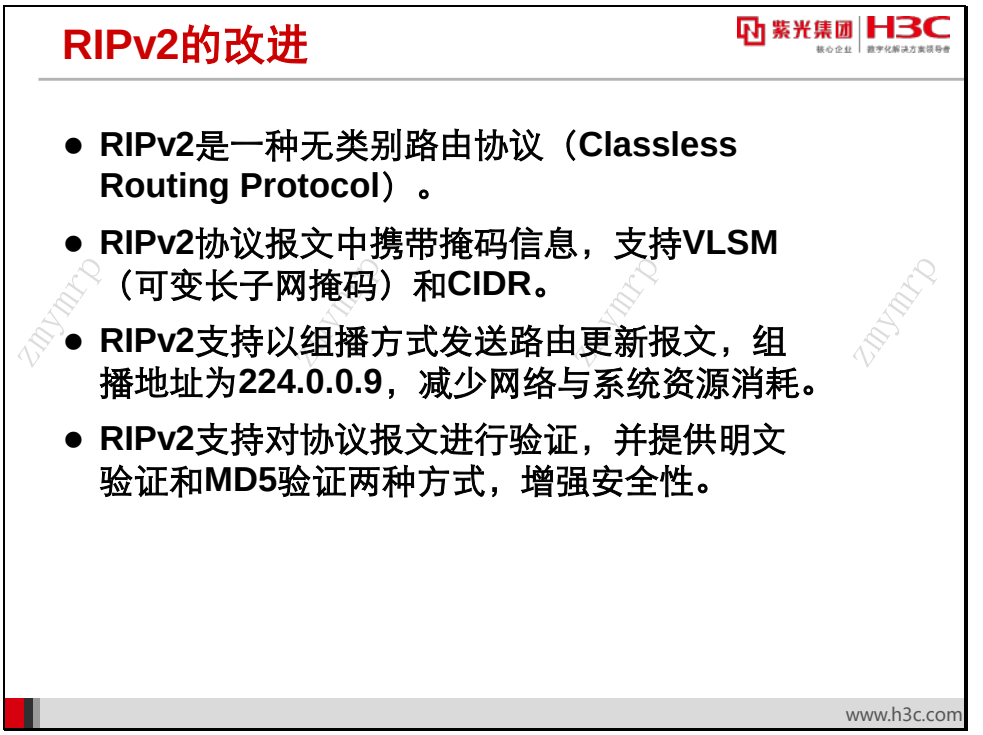
24.6 RIPv2的改进



RIPv1 是有类路由协议，它的协议报文不携带掩码信息，在交换子网路由时，有时会发生错误。

如上图所示，RTA 发送了路由 10.0.0.0 给 RTB。因此路由无掩码信息，且 10.0.0.0 是一个 A 类地址，所以 RTB 收到后，会给此路由加自然掩码。也就是说，RTB 的路由表中路由项目的地址/掩码是 10.0.0.0/8。这样就造成了错误的路由信息。

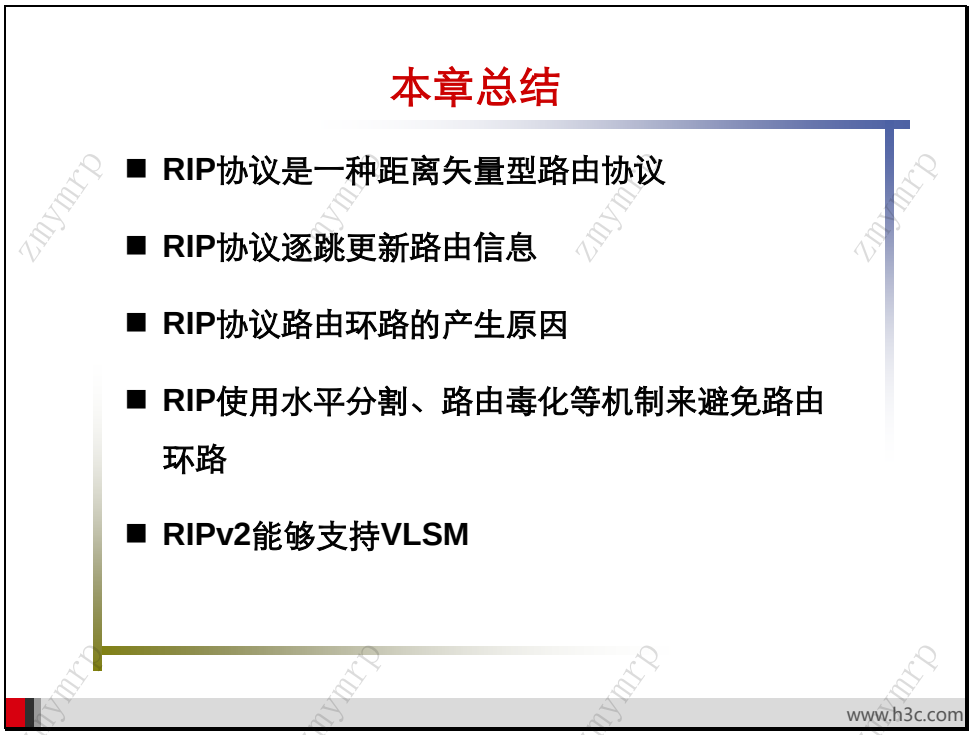
RIPv1 的其他缺点有：只支持以广播方式发布协议报文，系统和网络开销都较大；RIPv1 不支持验证，协议安全没有保证。RIPv1 的上述缺点在 RIPv2 得到了改进。



RIPv2 是一种无类别路由协议（Classless Routing Protocol），与 RIPv1 相比，它有以下优势：

- 报文中携带掩码信息，支持 VLSM（可变长子网掩码）和 CIDR（Classless Inter-Domain Routing，无类域间路由）。
- 支持组播路由发送更新报文，减少资源消耗。
- 支持对协议报文进行验证，并提供明文验证和 MD5 验证两种方式，增强安全性。

24.7 本章总结

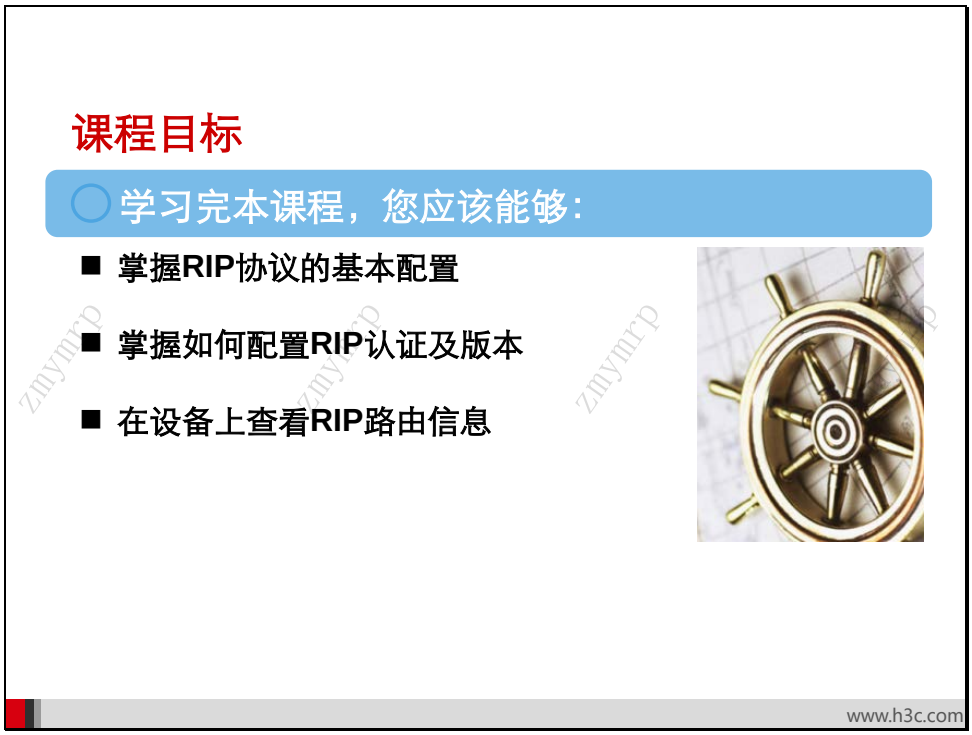

本章总结
RIP协议是一种距离矢量型路由协议
RIP协议逐跳更新路由信息
RIP协议路由环路的产生原因
RIP使用水平分割、路由毒化等机制来避免路由环路
RIPv2能够支持VLSM
www.h3c.com

第25章 配置 RIP

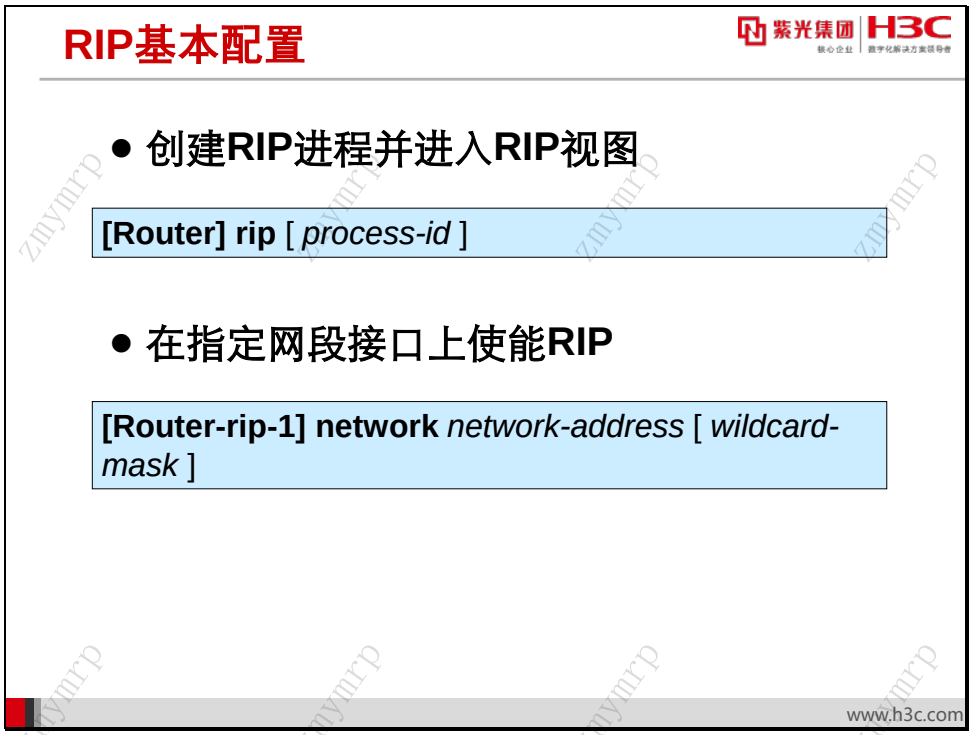
理解了 RIP 的工作原理之后，要在路由器上配置 RIP，必须掌握相关的配置步骤和命令。RIPv1 和 RIPv2 的配置有所区别，配置时也应注意。

本章首先介绍了路由器上 RIP 协议的基本配置命令，并介绍了一些 RIP 的可选命令，如配置被动接口等。然后介绍了如何配置 RIPv2，如何在 RIPv2 中启用认证、取消聚合功能等。

25.1 本章目标



25.2 RIP协议基本配置



通常，我们在路由器上启用 RIP 协议时，首先需要对 RIP 进行一个基本的配置。进行 RIP 基本配置的步骤如下：

第1步：在系统视图下用 rip 命令启动 RIP 进程并进入 RIP 视图。配置命令为：

rip [*process-id*]

process-id：进程 ID。通常我们不必指定，系统自动选用 RIP 进程 1 作为当前 RIP 的进程。

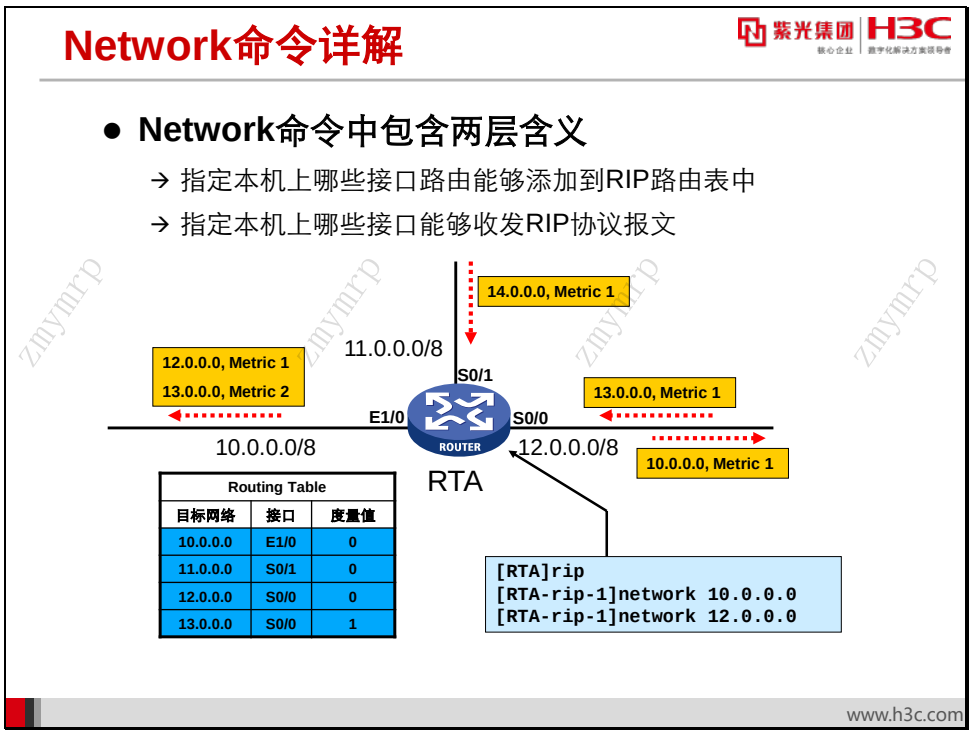
第2步：在 RIP 视图下用 network 命令指定哪些网段接口使能 RIP。配置命令为：

network *network-address* [*wildcard-mask*]

network-address：指定网段的地址，其取值可以为各个接口的 IP 网络地址。

wildcard-mask：IP 地址掩码的反码，相当于将 IP 地址的掩码取反（0 变 1，1 变 0）。

network 0.0.0.0 命令用来在所有接口上使能 RIP。

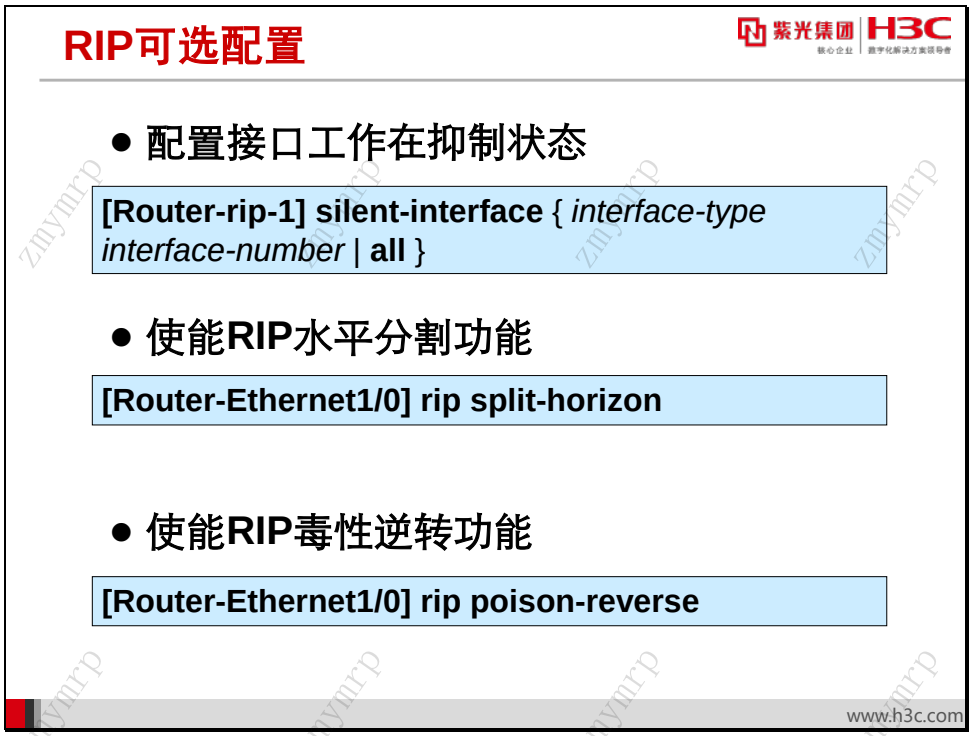


network 命令实际上有两层含义，一方面用来指定本机上哪些直连路由被 RIP 进程加入到 RIP 路由表中，另一方面用来指定哪些接口能够收发 RIP 协议报文。

配置 network 命令后，RIP 进程会将指定网段所包含的直连路由添加到 RIP 路由表中，RIP 路由表以路由更新的方式从接口向外广播；RIP 进程会在指定网段所包含的接口上接收和发送 RIP 路由更新。对于不在指定网段上的接口，RIP 既不在它上面接收和发送路由，也不将它的接口直连路由转发出去。

如图中，路由器 RTA 有三个接口。启用 RIP 协议并用 network 命令指定后，接口 E1/0 和 S0/0 所连接的直连路由 10.0.0.0 和 12.0.0.0 被加入到 RIP 路由表中；同时，接口 E1/0 和 S0/0 能够收发 RIP 协议报文。路由器从接口 S0/0 收到 RIP 路由 13.0.0.0，把它加到 RIP 路由表中；在路由更新周期到来后，从接口 E1/0 上发送出去。而由于接口 S0/1 的 IP 地址不在 network 命令所指定范围内，所以它虽然接收到了路由 14.0.0.0，但是不会加入到 RIP 路由表中，也不会从其它接口发送出去。

25.3 RIP可选配置



在不同网络环境中，可适当对 RIP 的配置做一些调整，以使 RIP 更好的运行。

有时候我们想让路由器的某些接口只接收 RIP 协议报文，而不发送 RIP 协议报文。比如，一台路由器的以太网口连接 PC，对于 PC 来说，它不需要接收 RIP 协议报文，所以路由器没有必要发送 RIP 协议报文给它。这种情况下，我们可以在 RIP 视图下用 silent-interface 命令来使某些接口只接收而不发送 RIP 协议报文。相关配置命令为：

silent-interface { *interface-type interface-number* | **all** }

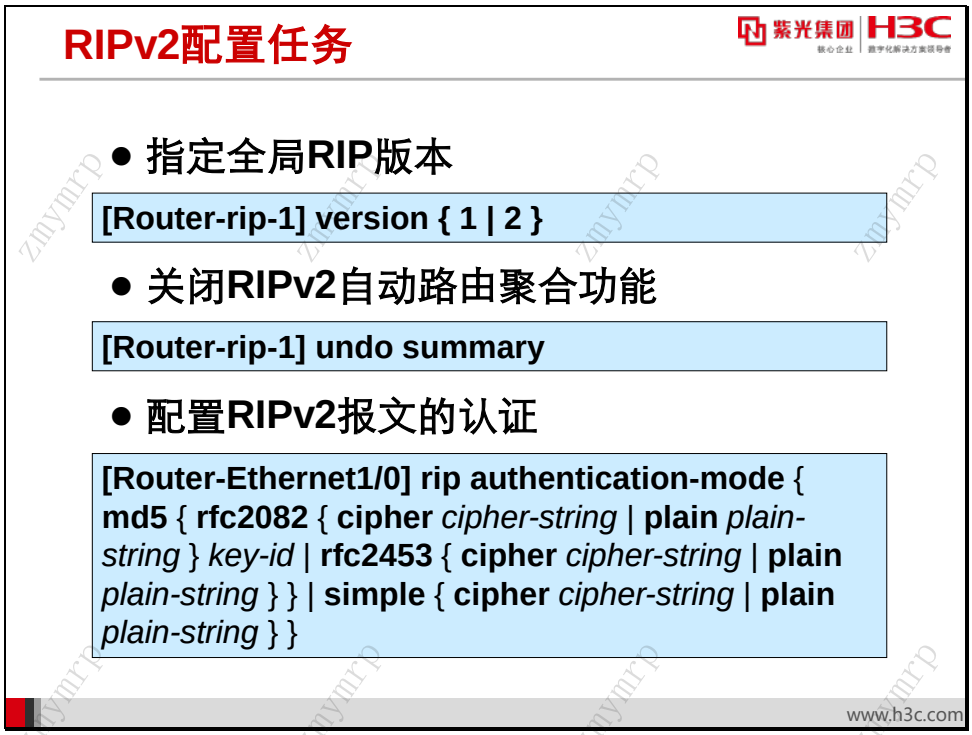
启动 RIP 后，水平分割功能默认是启用的。但如果水平分割被人为关闭，我们可以在接口视图下用以下命令来使能水平分割功能：

rip split-horizon

另外一种避免环路的机制是毒性逆转。毒性逆转功能默认是关闭的。要使能毒性逆转，需要在接口视图下用以下命令打开：

rip poison-reverse

25.4 RIPv2相关配置



RIPv1 不支持不连续子网和认证等机制，所以在网络中使用 RIPv2 是比较理想的选择。在 RIP 视图下使用 version 命令来指定 RIP 的全局版本：

version { 1 | 2 }

使用上述命令指定 RIP 版本为 1 后，路由器的所有接口都以广播形式发送 RIP 协议报文。

另外，也可以在接口视图下指定接口所运行 RIP 的版本和形式：

rip version { 1 | 2 [broadcast | multicast] }

RIPv1 和 RIPv2 都支持路由自动聚合功能。路由聚合是指将同一自然网段内的不同子网的路由聚合成一条自然掩码的路由然后发送，目的是为了减少网络上的流量。在 RIPv1 中，自动聚合功能默认是打开的，且不能关闭；RIPv2 支持关闭自动聚合。当需要将所有子网路由广播出去时，可以在 RIP 视图下关闭 RIPv2 的自动路由聚合功能：

undo summary

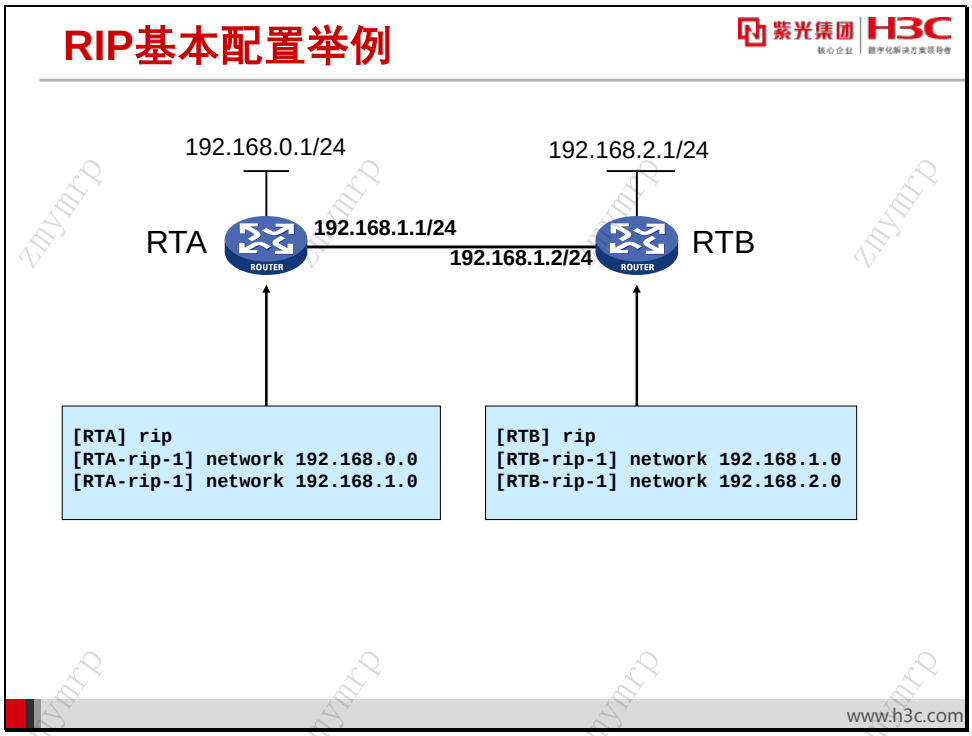
RIPv2 支持两种认证方式：明文认证和 MD5 密文认证。明文认证不能提供安全保障，未加密的认证字随报文一同传送，所以明文认证不能用于安全性要求较高的情况。在接口视图下可以启用认证并指定认证类型：

rip authentication-mode { md5 { rfc2082 { cipher *cipher-string* **| plain** *plain-string* **}** *key-id* **| rfc2453 { cipher** *cipher-string* **| plain** *plain-string* **} } | simple { cipher** *cipher-string* **| plain** *plain-string* **} }**

参数含义如下：

- **md5**：MD5 验证方式。
- **rfc2082**：指定 MD5 验证报文使用 RFC 2082 规定的报文格式。
- **cipher**：表示输入的密码为密文。
- *cipher-string*：表示设置的密文密码，为 33～53 个字符的字符串，区分大小写。
- **plain**：表示输入的密码为明文。
- *plain-string*：表示设置的明文密码，为 1～16 个字符的字符串，区分大小写。
- *key-id*：MD5 **rfc2082** 验证标识符，取值范围为 1～255。
- **rfc2453**：指定 MD5 验证报文使用 RFC 2453 规定的报文格式（IETF 标准）。
- **simple**：简单验证方式。

25.5 RIP基本配置举例



上图是 RIP 的基本配置示例。图中所有的网络使用自然掩码，没有子网划分，所以可以使用 RIPv1。在两台路由器所有的接口上使能 RIP。

配置 RTA：

```
[RTA] rip
[RTA-rip] network 192.168.0.0
[RTA-rip] network 192.168.1.0
```

配置 RTB：

```
[RTB] rip
[RTB-rip] network 192.168.1.0
[RTB-rip] network 192.168.2.0
```

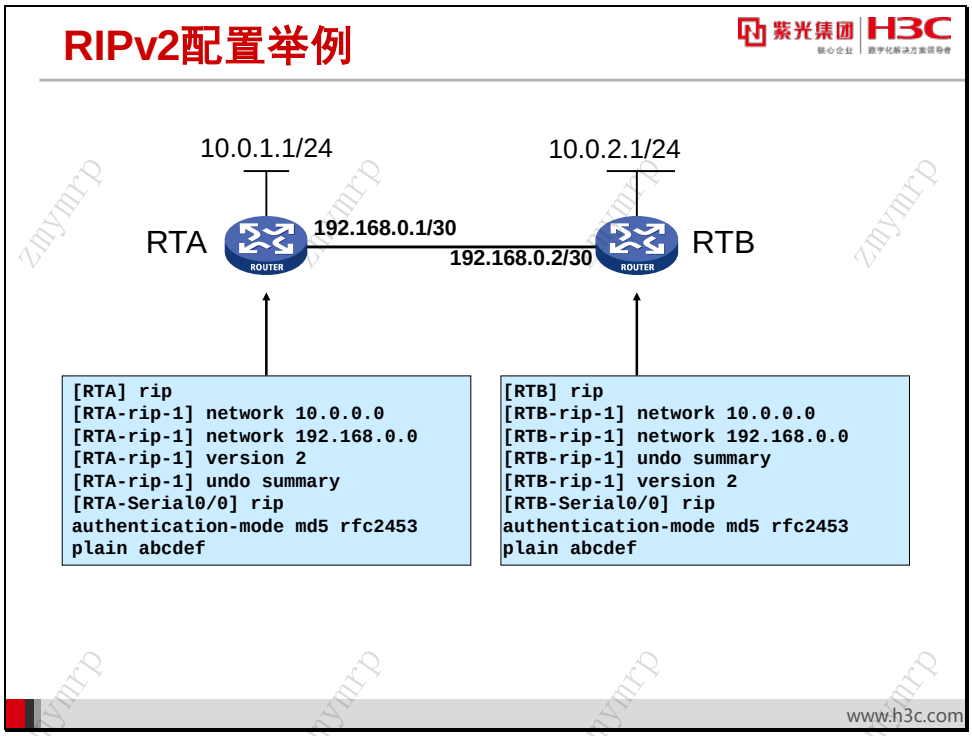
配置完成后，在 RTA 上查看 IP 路由表：

```
[RTA]display ip routing-table

Destinations : 18      Routes : 18

Destination/Mask   Proto   Pre Cost       NextHop         Interface
0.0.0.0/32          Direct  0   0          127.0.0.1       InLoop0
127.0.0.0/8         Direct  0   0          127.0.0.1       InLoop0
127.0.0.0/32        Direct  0   0          127.0.0.1       InLoop0
127.0.0.1/32        Direct  0   0          127.0.0.1       InLoop0
127.255.255.255/32 Direct  0   0          127.0.0.1       InLoop0
192.168.0.0/24      Direct  0   0          192.168.0.1     GE0/0
192.168.0.0/32      Direct  0   0          192.168.0.1     GE0/0
192.168.0.1/32      Direct  0   0          127.0.0.1       InLoop0
192.168.0.255/32    Direct  0   0          192.168.0.1     GE0/0
```

```
192.168.1.0/24      Direct  0   0         192.168.1.1     Ser2/0
192.168.1.0/32      Direct  0   0         192.168.1.1     Ser2/0
192.168.1.1/32      Direct  0   0         127.0.0.1       InLoop0
192.168.1.2/32      Direct  0   0         192.168.1.2     Ser2/0
192.168.1.255/32    Direct  0   0         192.168.1.1     Ser2/0
192.168.2.0/24      RIP     100 1         192.168.1.2     Ser2/0
224.0.0.0/4         Direct  0   0         0.0.0.0         NULL0
224.0.0.0/24        Direct  0   0         0.0.0.0         NULL0
255.255.255.255/32  Direct  0   0         127.0.0.1       InLoop0
```

可以看到，RTA 通过 RIP 协议学习到了路由 192.168.2.0/24。下一跳为 192.168.1.2，说明是经过 RTB 学习到的；代价是 1，说明到 192.168.2.0/24 需要经过一跳。

25.6 RIPv2配置举例



RIPv2 能够支持在协议报文中携带掩码，并支持认证。由于上图中使用了子网划分，且子网掩码也不连续，所以需要在两台路由器间运行 RIPv2。

配置 RTA：

```
[RTA] rip
[RTA-rip] network 10.0.0.0
[RTA-rip] network 192.168.0.0
[RTA-rip] version 2
[RTA-rip] undo summary
[RTA-Serial2/0] rip authentication-mode md5 rfc2453 plain abcdef
```

配置 RTB：

```
[RTB] rip
[RTB-rip] network 10.0.0.0
[RTB-rip] network 192.168.0.0
[RTB-rip] undo summary
[RTB-rip] version 2
[RTB-Serial2/0] rip authentication-mode md5 rfc2453 plain abcdef
```

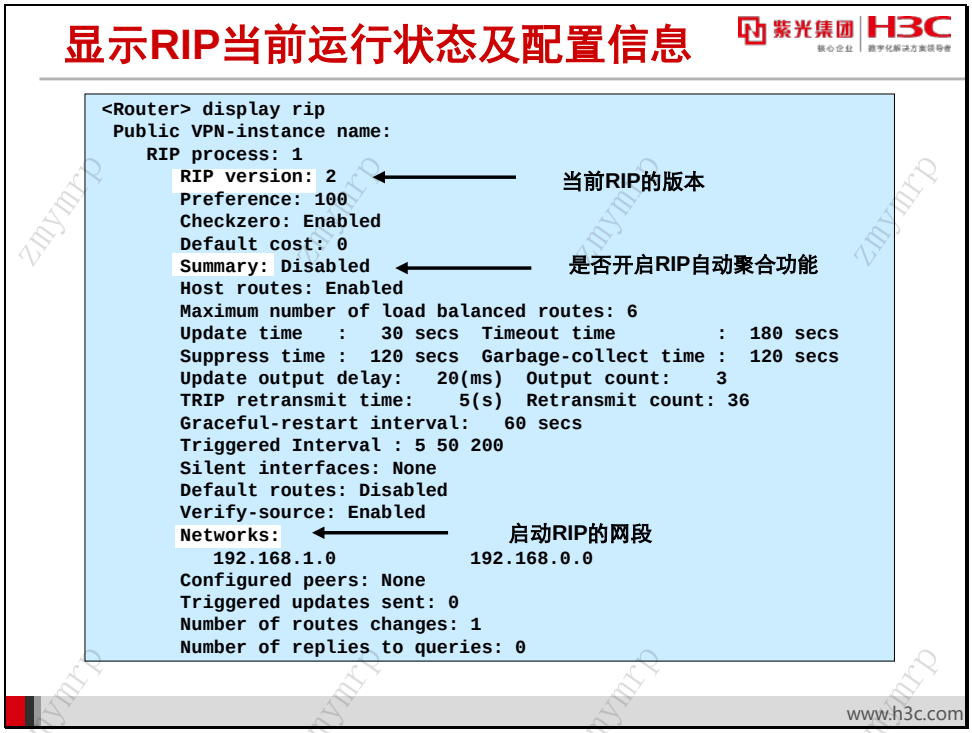
配置完成后，在 RTA 上查看 IP 路由表：

```
[RTA]dis ip routing-table

Destinations : 18      Routes : 18

Destination/Mask   Proto   Pre Cost       NextHop         Interface
0.0.0.0/32         Direct  0   0          127.0.0.1       InLoop0
10.0.1.0/24        Direct  0   0          10.0.1.1        GE0/0
10.0.1.0/32        Direct  0   0          10.0.1.1        GE0/0
```

```
10.0.1.1/32       Direct  0   0          127.0.0.1      InLoop0
10.0.1.255/32     Direct  0   0          10.0.1.1       GE0/0
10.0.2.0/24       RIP     100 1          192.168.0.2    Ser2/0
127.0.0.0/8       Direct  0   0          127.0.0.1      InLoop0
127.0.0.0/32      Direct  0   0          127.0.0.1      InLoop0
127.0.0.1/32      Direct  0   0          127.0.0.1      InLoop0
127.255.255.255/32 Direct  0   0          127.0.0.1      InLoop0
192.168.0.0/30    Direct  0   0          192.168.0.1    Ser2/0
192.168.0.0/32    Direct  0   0          192.168.0.1    Ser2/0
192.168.0.1/32    Direct  0   0          127.0.0.1      InLoop0
192.168.0.2/32    Direct  0   0          192.168.0.2    Ser2/0
192.168.0.3/32    Direct  0   0          192.168.0.1    Ser2/0
224.0.0.0/4       Direct  0   0          0.0.0.0        NULL0
224.0.0.0/24      Direct  0   0          0.0.0.0        NULL0
255.255.255.255/32 Direct  0   0          127.0.0.1      InLoop0
```

可以看到，RTA 通过 RIP 协议学习到了路由 10.0.2.0/24。

25.7 RIP运行状态及配置信息查看



在任意视图下可以使用 display rip 来查看 RIP 当前运行状态及配置信息。

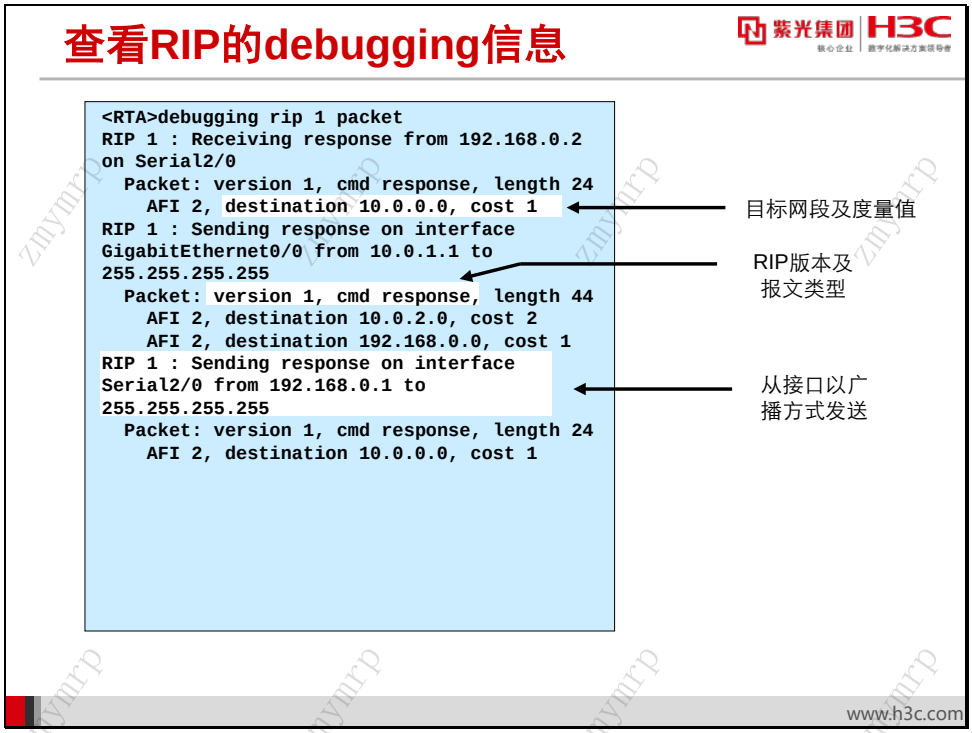
display rip

由图中命令输出我们可得知，当前 RIP 的运行版本是 RIPv2；自动聚合功能是关闭的；使能 RIP 的网段为 192.168.1.0 和 192.168.0.0。另外，图中常用的 RIP 信息及含义如下表：

表25-1 RIP 输出信息

字段	描述
RIP process	RIP进程号
Preference	RIP路由优先级
Update time	Update定时器的值，单位为秒
Timeout time	Timeout定时器的值，单位为秒
Garbage-collect time	Garbage-Collect定时器的值，单位为秒
Silent interfaces	抑制接口数（这些接口不发送周期更新报文）
Default routes	是否向RIP邻居发布一条默认路由，Enable表示发布，Disabled表示不发布
Triggered updates sent	发送的触发更新报文数

25.8 查看RIP的debugging信息



在用户视图下可以使用 debugging 命令来查看 RIP 协议收发报文的情况。相应命令为：

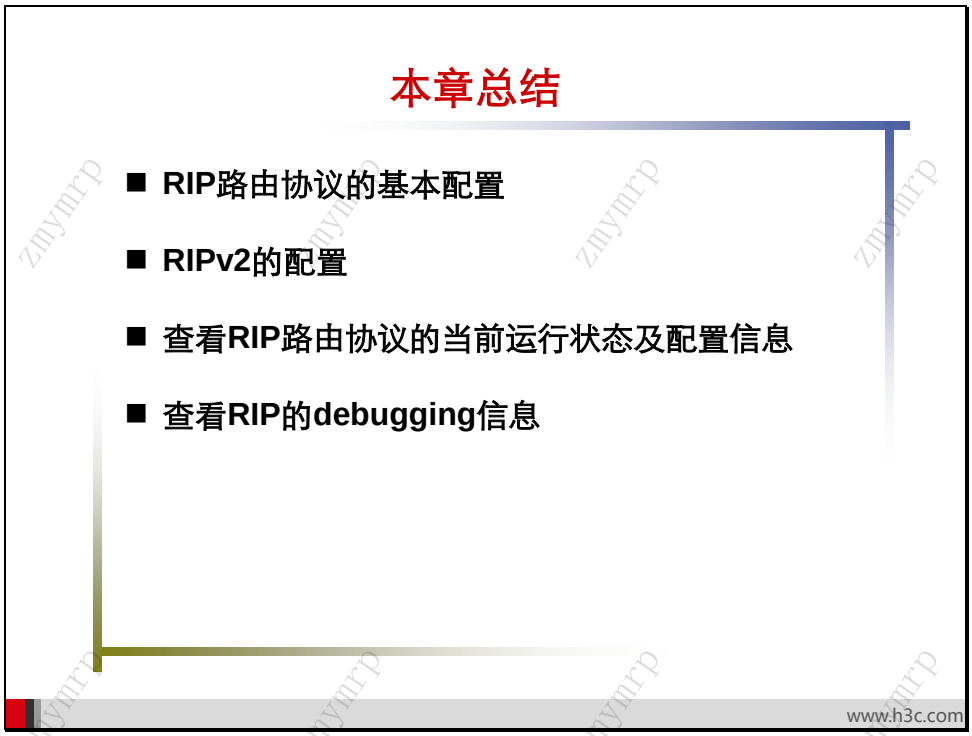
debugging rip 1 packet

由图中的 debugging 输出信息我们可知道接口发送 RIP 协议报文的版本，是以广播方式还是组播方式发送的；路由更新中含有哪些目标网段，相应的度量值是多少。

另外还可以获知接口收到的路由更新有哪些路由，相应的度量值。

示例中使用 RIPv1。如果路由器运行 RIPv2，路由更新中还应该包含了路由掩码。

25.9 本章总结

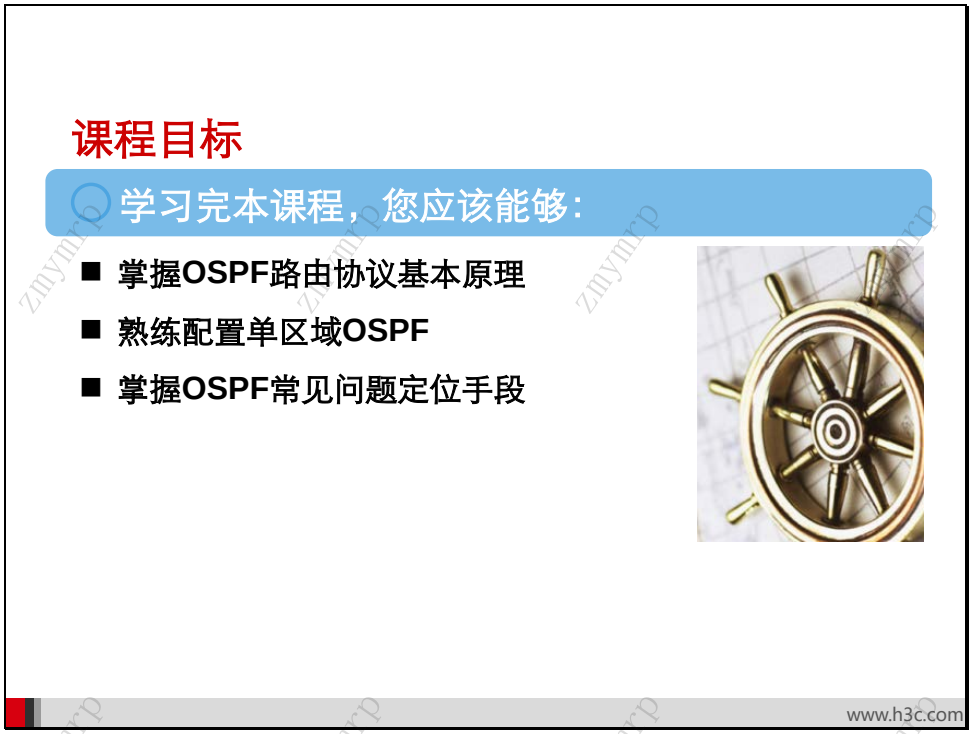

本章总结
RIP路由协议的基本配置
RIPv2的配置
查看RIP路由协议的当前运行状态及配置信息
查看RIP的debugging信息
www.h3c.com

第26章 OSPF 基础

由于 RIP 路由协议存在无法避免的缺陷，所以在规划网络时，其多用于构建中小型网络。但随着网络规模的日益扩大，一些小型企业网的规模几乎等同于十几年前的中型企业网，RIP 路由协议显然已经不能完全满足这样的需求。

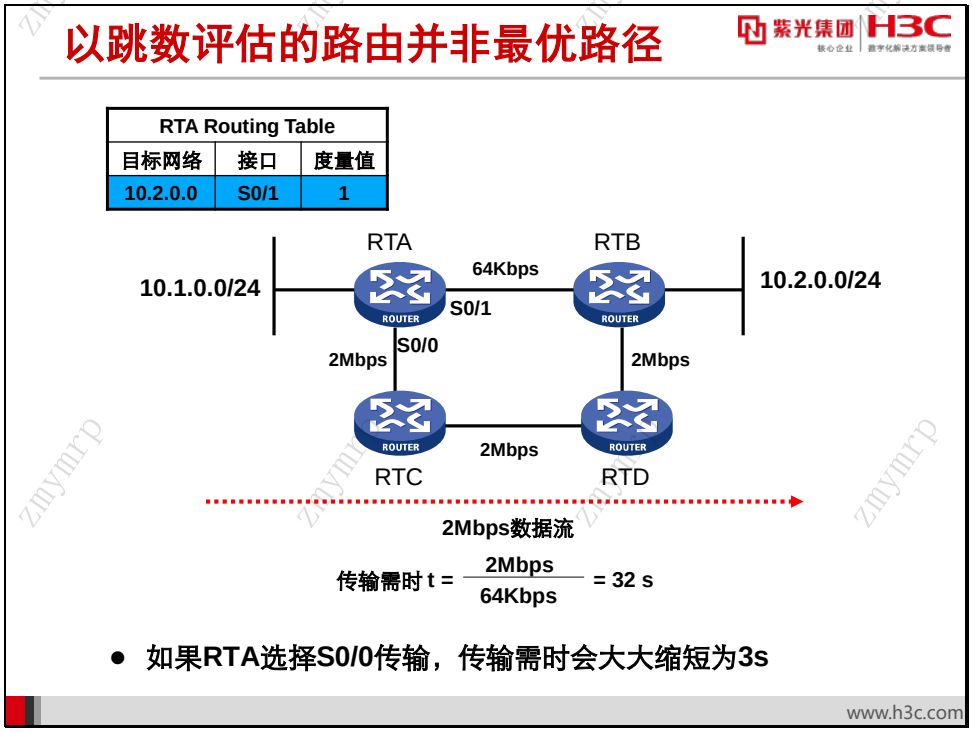
在这种背景下，OSPF（Open Shortest Path First，开放最短路径优先）路由协议以其众多的优势脱颖而出。它解决了很多 RIP 路由协议无法解决的问题，因而得到了广泛应用。

26.1 本章目标



26.2 RIP的缺陷

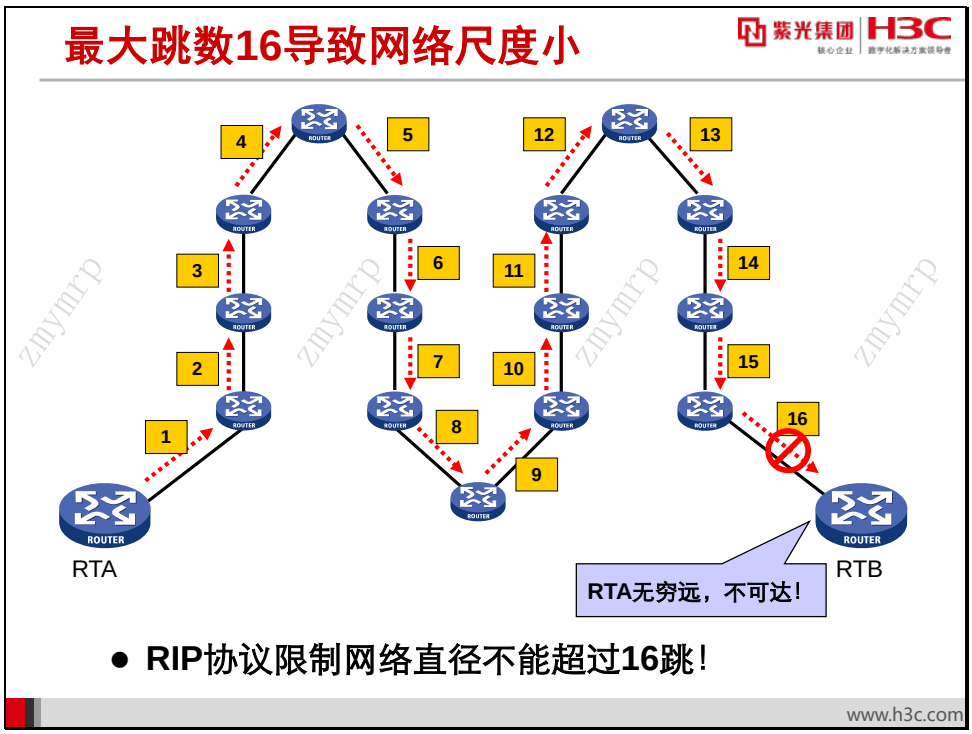
RIP 路由协议由于其自身算法的限制，不可避免的会引入路由环路。尽管后续增加了水平分割、抑制时间和毒性逆转等方法来避免这个问题，但一方面这些功能使得 RIP 网络的路由计算变得复杂，网络收敛慢，另一方面它们对于稍大些的复杂网络仍然无能为力，无法从理论上完全避免路由环路产生。除此以外，RIP 还存在其他无法避免的缺陷，使其只能用于中小型网络中。



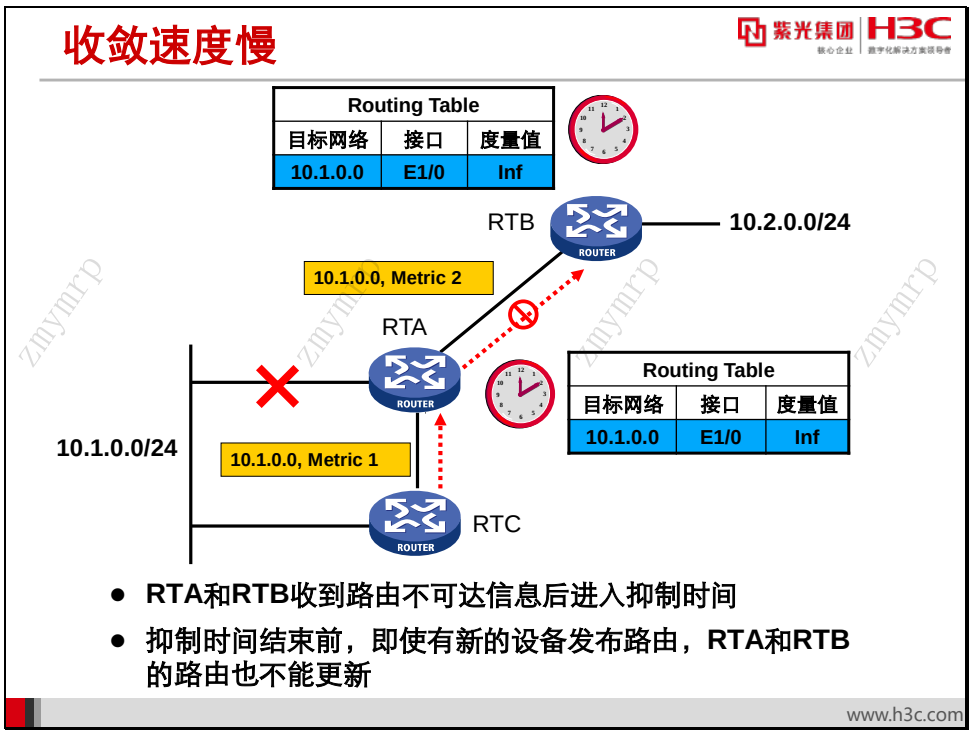
RIP 通过跳数来衡量到达目标网络的最优路径，但这种评估路由优先级的方式在大多数网络中是不合适的。

在如图所示的简单网络结构中，如果使用 RIP 协议，那么 RTA 路由器将认为到达目标网络 10.2.0.0/24 的最优路径是直接通过 S0/1 接口达到 RTB 路由器，因为根据距离矢量算法计算，RTA 和 RTB 直连，它们之间的跳数最小。但如果从网络传输时延的角度评估，这种选择显然是不恰当的，因为通过 RTC 到达目的网络的另一条路径的带宽远远高于所选定的路径。

事实上，在大多数网络中，以网络带宽和链路时延来衡量网络质量更加合理。



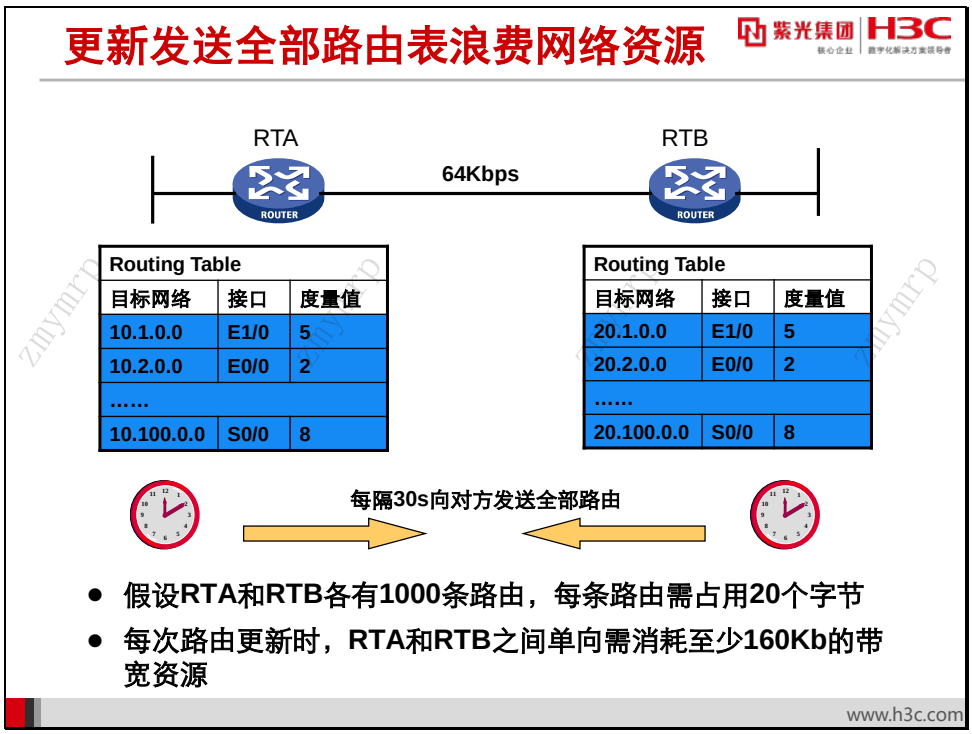
RIP 支持的最大跳数为 16，这决定了其无法用于构建规模较大的网络。在启用 RIP 协议的网络里，每一个 RIP 路由器只能接收到网络中相邻路由器的路由表，接收到相邻路由器的路由信息后，RIP 路由器将路由信息的度量值（Metric）增加一跳后再传送给相邻路由器。这种逐步传递路由信息的过程只发生在相邻路由器之间。当跳数增加到 16 以后，路由器会认为距离无穷远，目标网络不可达。这一限制决定了任意两个设备之间的距离不能超过 16 跳。



RIP 的收敛速度慢，只适用于网络结构变化缓慢的地区性网络。为防止路由环路，RIP 采用抑制时间的机制对不可达路由信息更新进行抑制。在这种机制下，即使有新设备发布路由，只要新发布的路由的度量值比原有路由的大，RIP 也不能及时更新路由表，必须等到抑制时间结束后才能更新该路由信息。

在上图所示网络中，因为 RTA 所连接网络 10.1.0.0/24（度量值为 0）产生故障，所以 RTA 将此路由度量值设为无穷大，并通知 RTB 将其路由表中网络 10.1.0.0/24 的度量值也设为无穷大。而当 RTC 通知 RTA 网络 10.1.0.0/24 可达（其度量值为 1）时，RTA 认为这条路由不可信（因为其度量值大于原来的度量值 0），所以 RTA 并不更新自己的路由表，也不会通知 RTB 这个路由已经有了更新。

同时，由于 RIP 的更新周期比较长，一个邻居路由器突然离线，其他路由器可能需要较长的时间才能察觉，这也造成 RIP 的收敛速度较慢。



为保证网络同步，RIP 每 30 秒向相邻路由器发布自身的全部路由信息。RIP 的网络规模越大，在路由刷新周期内需要发送的路由信息越多。网络上可能会充斥大量的 RIP 路由信息包，占用大量的网络资源。如图所示，每条 RIP 路由需要占用 20 个字节，假设 RTA 和 RTB 各有 1000 条路由，当路由更新时，RTA 和 RTB 之间单向就需要消耗至少 160Kb 的带宽资源，这对于本来带宽资源就很少的网络而言，显然是一个很重的负担。同时，发布和传输这些路由将占用较多的时间，网络收敛速度受到极大限制。

综上可见，RIP 路由协议并不适合大规模的网络。目前，RIP 主要用于较小型网络的构建。相对地，OSPF 协议很好地解决了上述问题，因此得到了广泛的使用。

26.3 OSPF基本原理

26.3.1 什么是 OSPF

OSPF（Open Shortest Path First，开放最短路径优先）是由 IETF（Internet Engineering Task Force，Internet 工程任务组）开发的基于链路状态（Link State）的自治系统内部路由协议，用来替代存在一些问题的 RIP 协议。目前通用的 OSPF 协议第二版由 RFC 2328 定义。

与距离矢量协议不同，链路状态路由协议使用 Dijkstra 的最短路径优先算法（Shortest Path First，SPF）计算和选择路由。这类路由协议关心网络中链路或接口的状态（up、down、IP 地址、掩码、带宽、利用率和时延等），每个路由器将其已知的链路状态向该区域的其他路由器通告，通过这种方式，网络上的每台路由器对网络结构都会有相同的认识。随后，路由器以其为依据，使用 SPF 算法计算和选择路由。

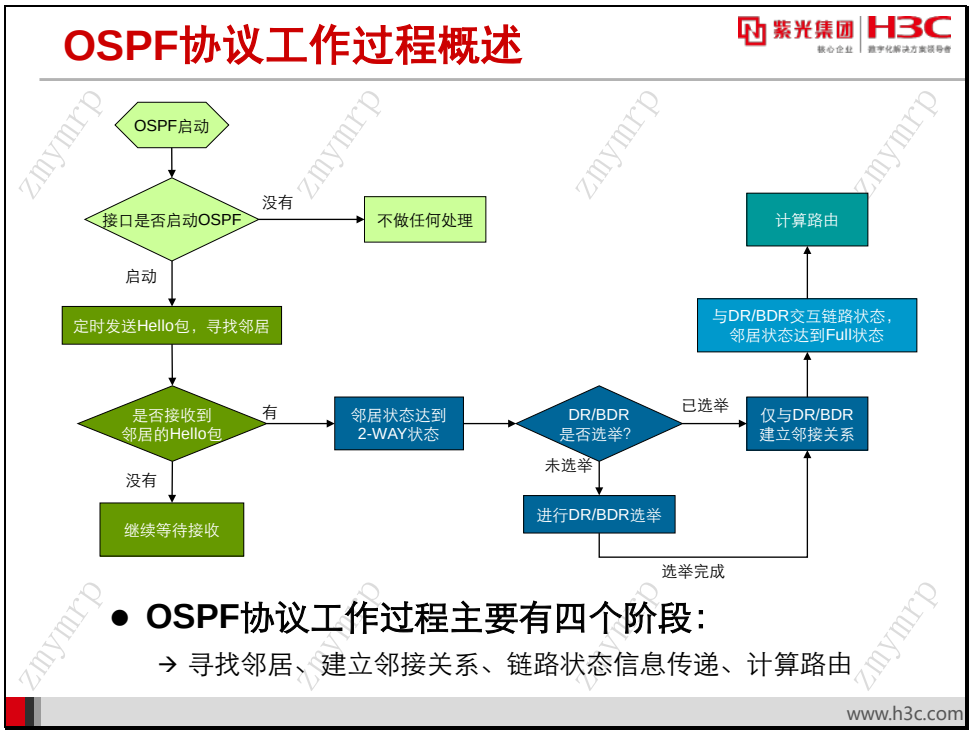
OSPF 协议在有组播发送能力的链路层上以组播地址发送协议包，既达到了节约资源的目的，又最大程度地减少了对其他网络设备的干扰。

OSPF 将协议包直接封装在 IP 包中，协议号 89。由于 IP 协议本身是无连接的，所以 OSPF 传输的可靠性需要协议本身来保证。因此，OSPF 协议定义了一些机制保证协议包安全可靠地传输。

总体说来，OSPF 协议比 RIP 具有更大的扩展性、快速收敛性和安全可靠性，同时，它采用路由增量更新的机制在保证全区域路由同步的同时，尽可能地减少了对网络资源的浪费。但

是 OSPF 的算法耗费更多的路由器内存和处理能力，在大型网络里，路由器本身承受的压力会很大。因此，OSPF 协议适合企业中小型网络构建。

26.3.2 OSPF 协议工作过程概述



上图描述了 OSPF 协议的四个主要工作过程：

- 寻找邻居

不同于 RIP，OSPF 协议运行后，并不立即向网络广播路由信息，而是先寻找网络中可与自己交互链路状态信息的周边路由器。可以交互链路状态信息的路由器互为邻居（Neighbor）。

- 建立邻接关系

邻接关系（Adjacency）可以想象为一条点到点的虚链路，它是在一些邻居路由器之间构成的。只有建立了可靠邻接关系的路由器才相互传递链路状态信息。

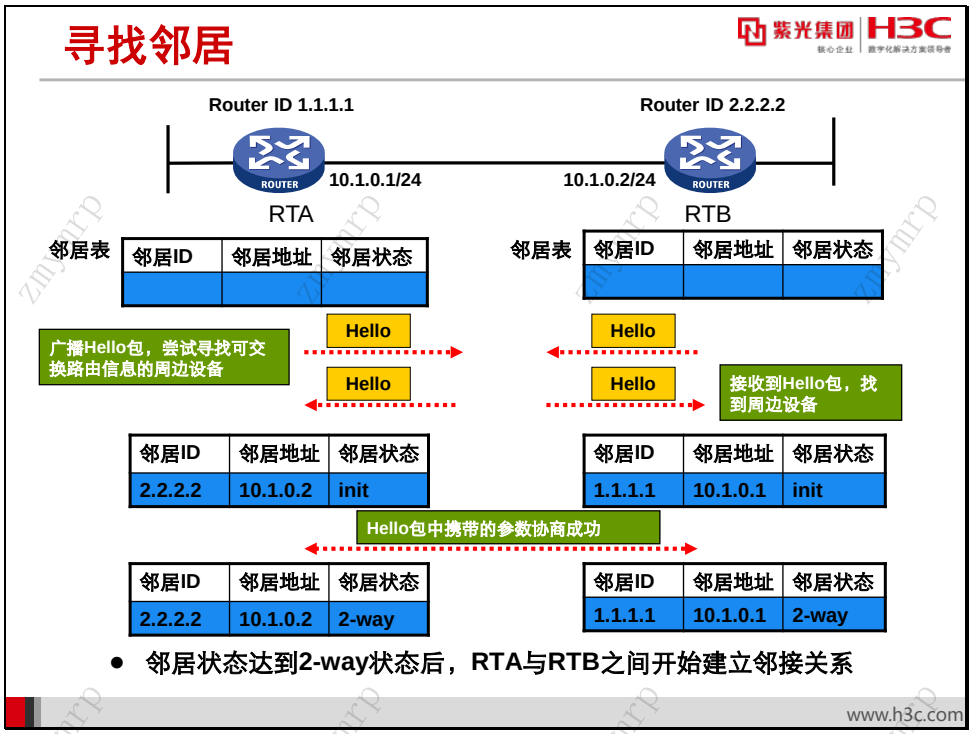
- 链路状态信息传递

OSPF 路由器将建立描述网络链路状况的 LSA（Link State Advertisement，链路状态公告），建立邻接关系的 OSPF 路由器之间将交互 LSA，最终形成包含网络完整链路状态信息的 LSDB（Link State DataBase，链路状态数据库）。

- 计算路由

获得了完整的 LSDB 后，OSPF 区域内的每个路由器将会对该区域的网络结构有相同的认识，随后各路由器将依据 LSDB 的信息用 SPF（Shortest Path First，最短路径优先）算法独立计算出路由。

26.3.3 寻找邻居



OSPF 路由器周期性地从其启动 OSPF 协议的每一个接口以组播地址 224.0.0.5 发送 Hello 包，以寻找邻居。Hello 包里携带有一些参数，比如始发路由器的 Router ID（路由器 ID）、始发路由器接口的区域 ID（Area ID）、始发路由器接口的地址掩码、选定的 DR 路由器、路由器优先级等信息。

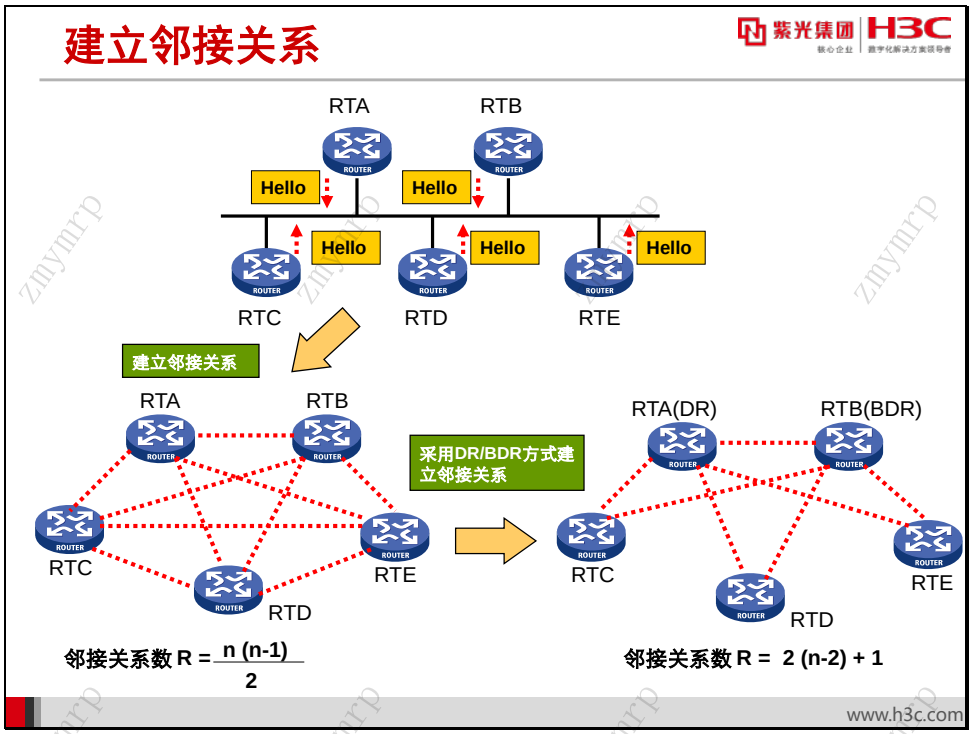
当两台路由器共享一条公共数据链路，并且相互成功协商它们各自 Hello 包中所指定的某些参数时，它们就能成为邻居。而邻居地址一般为启动 OSPF 协议并向外发送 Hello 包的路由器接口地址。

路由器通过记录彼此的邻居状态来确认是否与对方建立了邻接关系（Adjacency）。路由器初次接收到某路由器的 Hello 包时，仅将该路由器作为邻居候选人，将其状态记录为 Init 状态；只有在相互成功协商 Hello 包中所指定的某些参数后，才将该路由器确定为邻居，将其状态修改为 2-way 状态。当双方的链路状态信息交互成功后，邻居状态将变迁为 Full 状态，这表明邻居路由器之间的链路状态信息已经同步。

一台路由器可以有很多邻居，也可以同时成为几台其他路由器的邻居。邻居状态和维护邻居路由器的一些必要的信息都被记录在一张邻居表内，为了跟踪和识别每台邻居路由器，OSPF 协议定义了 Router ID（路由器 ID）。

Router ID 在 OSPF 区域内唯一标识一台路由器的 IP 地址。一台路由器可能有多个接口启动 OSPF，这些接口分别处于不同的网段，它们各自使用自己的接口 IP 地址作为邻居地址与网络里其他路由器建立邻居关系，但网络里的所有其他路由器只会使用 Router ID 来标识这台路由器。

26.3.4 建立邻接关系



可以将邻接关系比喻为一条点到点的虚连接，那么可以想象，在广播型网络的 OSPF 路由器之间的邻接关系是很复杂的。假设 OSPF 区域内有 5 台路由器，它们彼此互为邻居并都建立邻接关系，那么总共会有 10 个邻接关系；如果是 10 台路由器，那么就有 45 个邻接关系；如果有 n 台路由器，那么就有 n(n-1)/2 个邻接关系。邻接关系需要消耗较多的资源来维持，而且邻接路由器之间要两两交互链路状态信息，这也会造成网络资源和路由器处理能力的巨大浪费。

为了解决这个问题，OSPF 要求在广播型网络里选举一台 DR（Designated Router，指定路由器）。DR 负责用 LSA 描述该网络类型及该网络内的其他路由器，同时也负责管理他们之间的链路状态信息交互过程。

DR 选定后，该广播型网络内的所有路由器只与 DR 建立邻接关系，与 DR 互相交换链路状态信息以实现 OSPF 区域内路由器链路状态信息同步。值得注意的是，一台路由器可以有多个接口启动 OSPF，这些接口可以分别处于不同的网段里，这就意味着，这台路由器可能是其中一个网段的指定路由器，而不是其他网段的指定路由器，或者可能同时是多个网段的指定路由器。换句话说，DR 是一个 OSPF 路由器接口的特性，不是整台路由器的特性；DR 是某个网段的 DR，而不是全网的 DR。

如果 DR 失效，所有的邻接关系都会消失，此时必须重新选取一台新的 DR，网络上的所有路由器也要重新建立新的邻接关系并重新同步全网的链路状态信息。当这种问题发生时，网络将在一个较长时间内无法有效地传送链路状态信息和数据包。

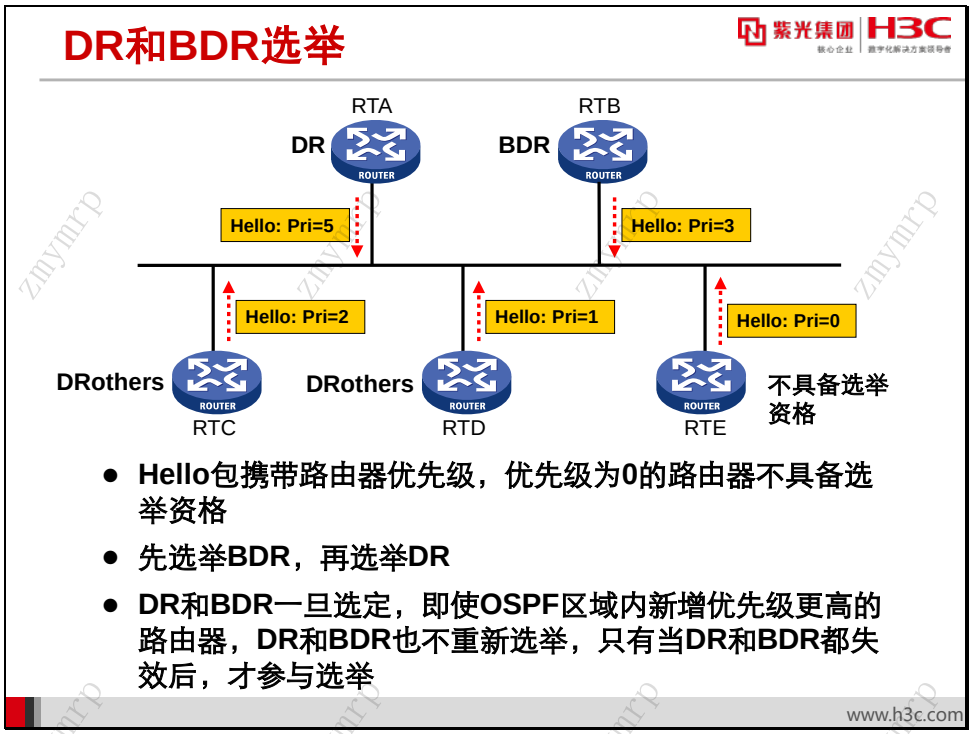
为加快收敛速度，OSPF 在选举 DR 的同时，还会再选举一个 BDR（Backup Designated Router，备份指定路由器）。网络上所有的路由器将与 DR 和 BDR 同时形成邻接关系，如果 DR 失效，BDR 将立即成为新的 DR。

采用选举 DR 和 BDR 的方法，广播型网络内的邻接关系减少为 2(n-2)+ 1 条，即 5 台路由器的邻接关系为 7 条，10 台路由器为 17 条。

注意：

邻居与邻接关系并不是一个概念。在广播型网络里，OSPF 区域内的路由器可以互为邻居，但只与 DR 和 BDR 建立邻接关系。

在 OSPF 的某些网络类型里，建立邻接关系时并不需要进行 DR 和 BDR 选举。本书未讨论全部细节，而只关注广播型网络（如以太网）的邻接关系的建立。



在初始阶段，OSPF 路由器会在 Hello 包里将 DR 和 BDR 指定为 0.0.0.0。当路由器接收到邻居的 Hello 包后，检查 Hello 包携带的路由器优先级（Router Priority）、DR 和 BDR 等字段，然后列出所有具备 DR 和 BDR 资格的路由器（优先级不为 0 的路由器均具备选举资格）。

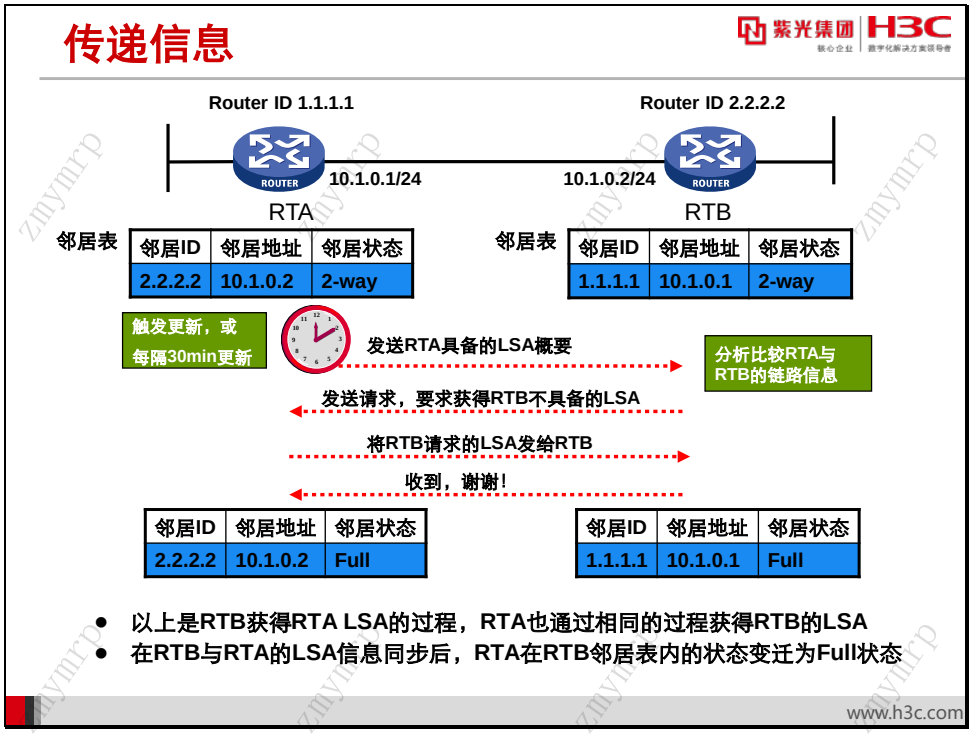
路由器优先级（Router Priority）取值范围从 0 至 255。在具备选举资格的路由器中，优先级最高的将被宣告为 BDR，优先级相同则 Router ID 大的优先。BDR 选举成功后，进行 DR 选举。如果同时有一台或多台路由器宣称自己为 DR，则优先级最高的将被宣告为 DR，优先级相同，则 Router ID 大的优先。如果网络里没有路由器宣称自己为 DR，则将已有的 BDR 推举为 DR，然后再执行一次选举过程选出新的 BDR。DR 和 BDR 选举成功后，OSPF 路由器会将

DR 和 BDR 的 IP 地址设置到 Hello 包的 DR 和 BDR 字段上，表明该 OSPF 区域内的 DR 和 BDR 已经有效。

虽然路由器的优先级可以影响选举过程，但它不能强制更改已经有效的 DR 和 BDR。当一台 OSPF 路由器加入一个 OSPF 区域时，如果该区域内尚未选举出 DR 和 BDR，则该路由器参与 DR 和 BDR 的选举，如果该区域内已经有有效的 DR 和 BDR，即使该路由器的优先级很高，也只能接受已经存在的 DR 和 BDR。因此在广播型网络里，最先初始化的具有 DR 选举资格的两台路由器将成为 DR 和 BDR。

一旦 DR 和 BDR 选举成功，其他路由器（DRothers）只与 DR 和 BDR 之间建立邻接关系。此后，所有路由器继续组播 Hello 包（组播地址为 224.0.0.5）来寻找新的邻居和维持旧邻居关系，而 DRothers 路由器只与 DR 和 BDR 交互链路状态信息，故 DRothers 与 DR、DRothers 与 BDR 之间的邻居状态可以达到 Full 状态，而 DRothers 之间的邻居状态只能停留在 2-way 状态。

26.3.5 链路状态信息传递

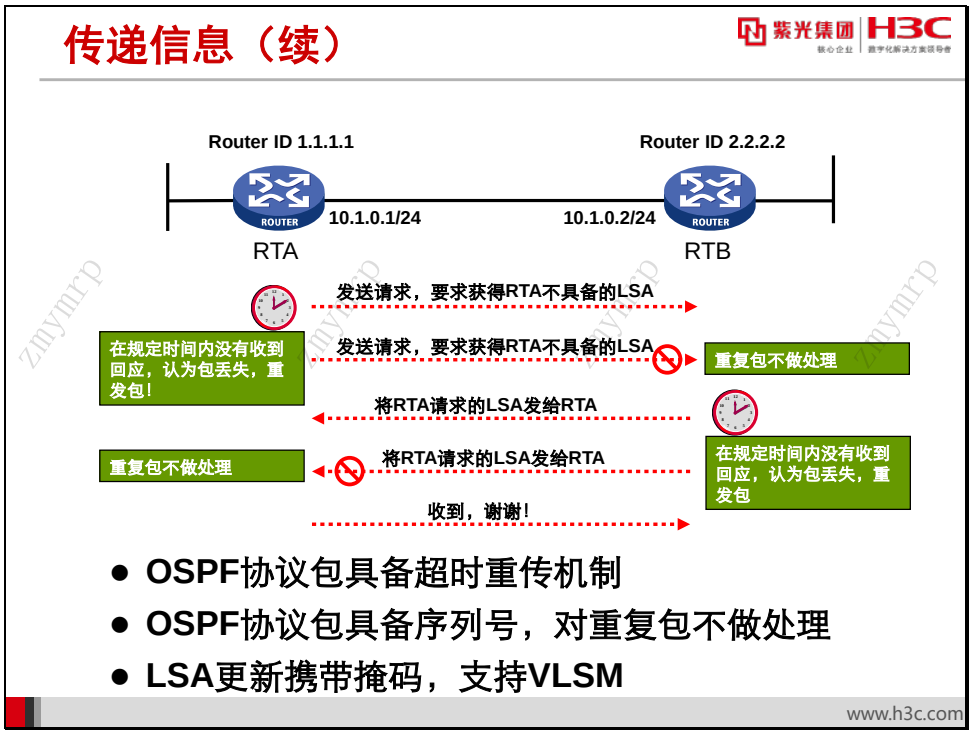


建立邻接关系的 OSPF 路由器之间通过发布 LSA（Link State Advertisement，链路状态公告）来交互链路状态信息。通过获得对方的 LSA，同步 OSPF 区域内的链路状态信息后，各路由器将形成相同的 LSDB（Link State DataBase，链路状态数据库）。

LSA 通告描述了路由器所有的链路信息（或接口）和链路状态信息。这些链路可以是到一个末梢网络（指没有和其他路由器相连的网络）的链路，也可以是到其他 OSPF 路由器的链路或是到外部网络的链路等。

为避免网络资源浪费，OSPF 路由器采取路由增量更新的机制发布 LSA，即只发布邻居缺失的链路状态给邻居。当网络变更时，路由器立即向已经建立邻接关系的邻居发送 LSA 摘要信息；而如果网络未发生变化，OSPF 路由器每隔 30 分钟向已经建立邻接关系的邻居发送一次 LSA 的摘要信息。摘要信息仅对该路由器的链路状态进行简单的描述，并不是具体的链路信息。邻居接收到 LSA 摘要信息后，比较自身链路状态信息，如果发现对方具有自己不具备的链路信息，则向对方请求该链路信息，否则不做任何动作。当 OSPF 路由器接收到邻居发来的请求某个 LSA 的包后，将立即向邻居提供它所需要的 LSA，邻居在接收到 LSA 后，会立即给对方发送确认包进行确认。

综上可见，OSPF 协议在发布 LSA 时进行了四次握手，这种方式不仅有效避免了类似 RIP 协议发送全部路由带来的网络资源浪费的问题，还保证了路由器之间信息传递的可靠性，提高了收敛速度。

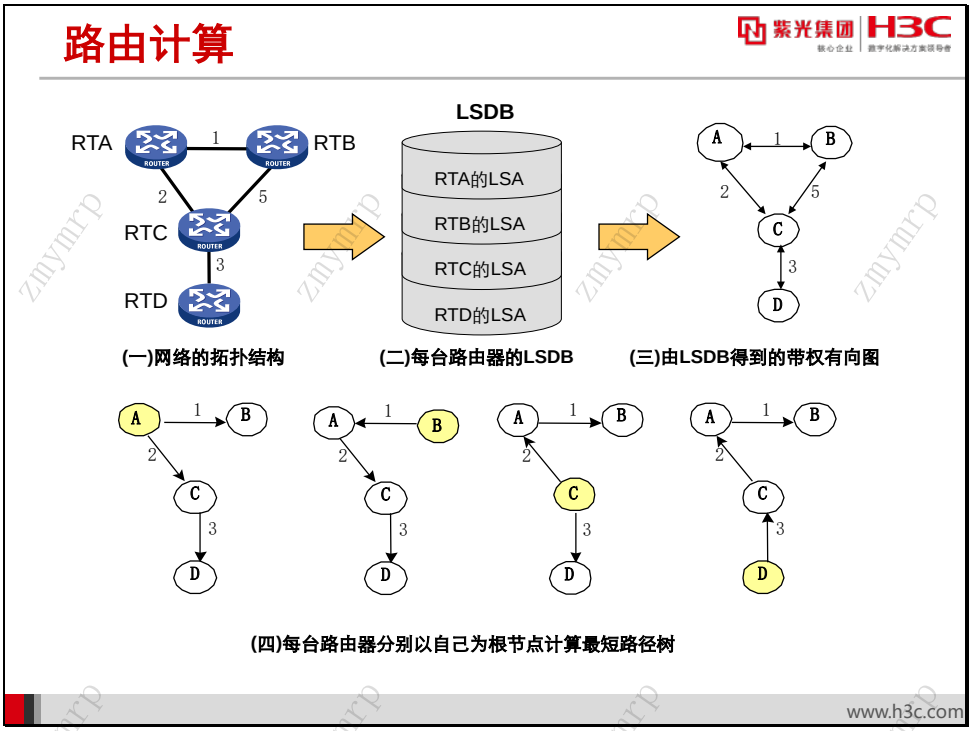


OSPF 协议具备超时重传机制。在 LSA 更新阶段，如果发送的包在规定时间内没有收到对方的回应，则认为包丢失，重新发送包。

为避免网络时延大造成路由器超时重传，OSPF 协议为每个包编写从小到大的序列号，当路由器接收到重复序列号的包时，只响应第一个包。

同时，由于 LSA 更新时携带掩码，OSPF 支持 VLSM（Variable-Length Subnet Mask，变长子网掩码），能准确反映实际网络情况。

26.3.6 路由计算



OSFP 路由计算通过以下步骤完成。

第一步：评估一台路由器到另一台路由器所需要的开销（Cost）

OSPF 协议是根据路由器的每一个接口指定的度量值来决定最短路径的，这里的度量值指的就是接口指定的开销。一条路由的开销是指沿着到达目的网络的路径上所有路由器出接口的开销总和。

Cost 值与接口带宽密切相关。H3C 路由器的接口开销是根据公式 100/带宽（Mbps）计算得到的，它可作为评估路由器之间网络资源的参考值。此外，用户也可以通过命令 **ospf cost** 手工指定路由器接口的 Cost 值。

第二步：同步 OSPF 区域内每台路由器的 LSDB

OSPF 路由器通过交换 LSA 实现 LSDB 的同步。LSA 不但携带了网络连接状况信息，而且携带了各接口的 Cost 信息。

由于一条 LSA 是对一台路由器或一个网段拓扑结构的描述，整个 LSDB 就形成了对整个网络的拓扑结构的描述。LSDB 实质上是一张带权的有向图，这张图便是对整个网络拓扑结构的真实反映。显然，OSPF 区域内所有路由器得到的是一张完全相同的图。

第三步：使用 SPF（Shortest Path First，最短路径优先算法）计算出路由

OSPF 路由器用 SPF 算法以自身为根节点计算出一棵最短路径树，在这棵树上，由根到各节点的累计开销最小，即由根到各节点的路径在整个网络中都是最优的，这样也就获得了由根去往各个节点的路由。计算完成后，路由器将路由加入 OSPF 路由表。当 SPF 算法发现有两

条到达目标网络的路径的 Cost 值相同，就会将这两条路径都将加入 OSPF 路由表，形成等价路由。

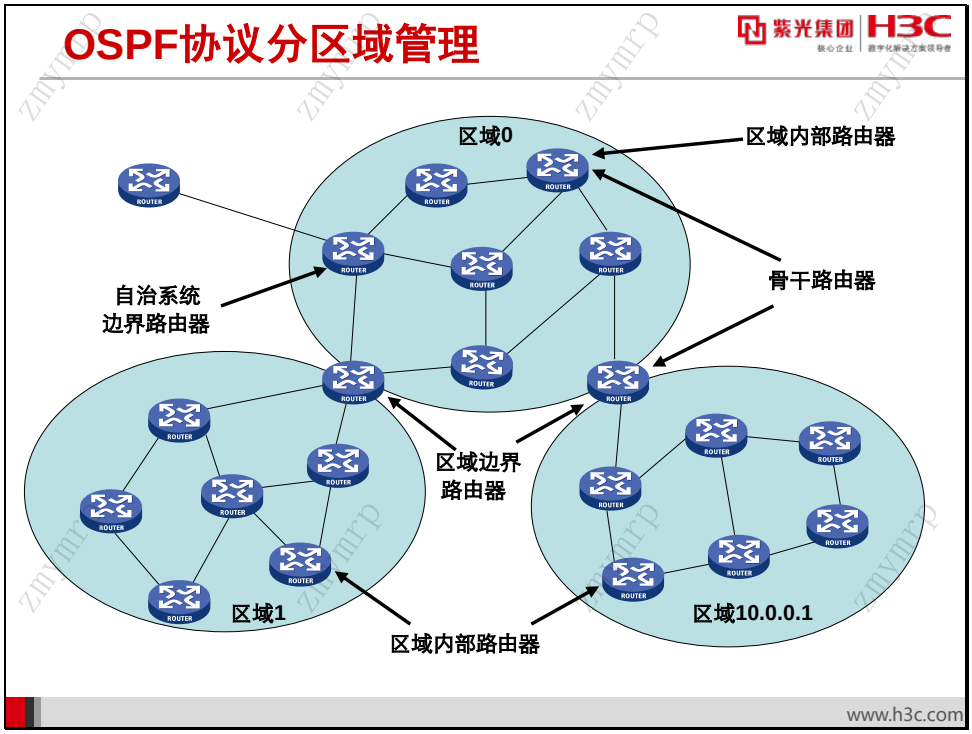
从 OSPF 协议的工作过程，能清晰地看出 OSPF 具备的优势：

- OSPF 区域内的路由器对整个网络的拓扑结构有相同的认识，在此基础上计算的路由不可能产生环路；
- 当网络结构变更时，所有路由器能迅速获得变更后的网络拓扑结构，网络收敛速度快；
- 由于引入了 Router ID 的概念，OSPF 区域内的每台路由器的行为都能很好地被跟踪；
- 使用 SPF 算法计算路由，路由选择与网络能力直接挂钩，选路更合理；
- OSPF 采用多种手段保证信息传递的可靠性、准确性，确保每台路由器网络信息同步，同时，避免了不必要的网络资源浪费。

综合起来看，OSPF 的确解决了 RIP 路由协议的一些固有缺陷，成为企业网络中最常用的路由协议之一。

26.3.7 OSPF 分区域管理

OSPF 协议使用了多个数据库和复杂的算法，这势必会耗费路由器更多的内存和 CPU 资源。当网络的规模不断扩大时，这些对路由器的性能要求就会显得过多，甚至会达到路由器性能极限。另一方面，Hello 包和 LSA 更新包也随着网络规模的扩大给网络带来难以承受的负担。为减少这些不利的影响，OSPF 协议提出分区域管理的解决方法。



OSPF 将一个大的自治系统划分为几个小的区域（Area），路由器仅需要与其所在区域的其他路由器建立邻接关系并共享相同的链路状态数据库，而不需要考虑其他区域的路由器。在这种情况下，原来庞大的数据链路状态数据库被划分为几个小数据库，并分别在每个区域里进行维护，从而降低了对路由器内存和 CPU 的消耗；同时，Hello 包和 LSA 更新包也被控制在一个区域内，更有利于网络资源的利用。

为区分各个区域，每个区域都用一个 32 位的区域 ID（Area ID）来标识。区域 ID 可以表示为一个十进制数字，也可以表示为一个点分十进制的数字，例如配置区域 0 等同于配置区域 0.0.0.0。

划分区域以后，OSPF 自治系统内的通信将划分为三种类型：

区域内通信——在同一个区域内的路由器之间的通信。

区域间通信——不同区域的路由器之间的通信。

区域外部通信——OSPF 域内路由器与另一个自治系统内的路由器之间的通信。

为完成上述的通信，OSPF 需要对本自治系统内的各区域及路由器进行任务分工。

OSPF 划分区域后，为有效管理区域间通讯，需要有一个区域作为所有区域的枢纽，负责汇总每一个区域的网络拓扑路由到其他所有的区域，所有的区域间通信都必须通过该区域，这个区域称为骨干区域（Backbone Area）。协议规定区域 0 是骨干域保留的区域 ID 号。

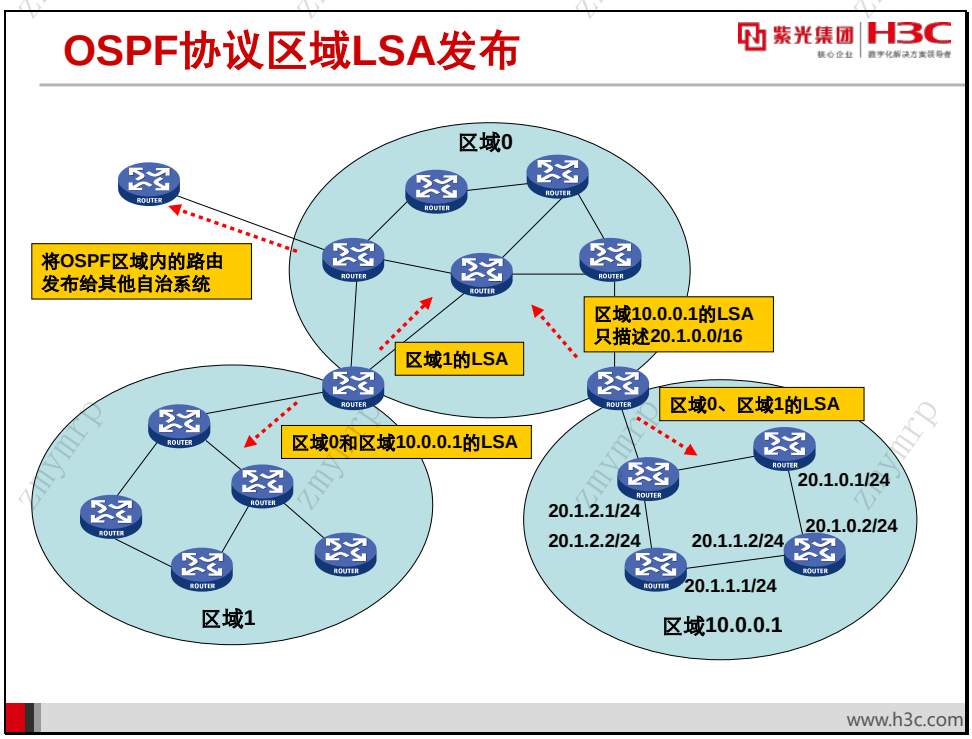
所有非骨干区域都必须与骨干区域相连，非骨干区域之间不能直接交换数据包，它们之间的路由传递只能通过区域 0 完成。区域 ID 仅是对区域的标识，与它内部的路由器 IP 地址分配无关。

至少有一个接口与骨干区域相连的路由器被称为骨干路由器（Backbone Router）。连接一个或多个区域到骨干区域的路由器被称为区域边界路由器（Area Border Routers，ABR），这些路由器一般会成为域间通信的路由网关。

OSPF 自治系统要与其他的自治系统通讯，必然需要有 OSPF 区域内的路由器与其他自治系统相连，这种路由器称为自治系统边界路由器（Autonomous System Boundary Router，ASBR）。自治系统边界路由器可以是位于 OSPF 自治系统内的任何一台路由器。

所有接口都属于同一个区域的路由器叫做内部路由器（Internal Router），它只负责域内通讯或同时承担自治系统边界路由器的任务。

划分区域后，仅在同一个区域的 OSPF 路由器能建立邻居和邻接关系。为保证区域间能正常通信，区域边界路由器需要同时加入两个及以上的区域，负责向它连接的区域发布其他区域的 LSA 通告，以实现 OSPF 自治系统内的链路状态同步，路由信息同步。因此，在进行 OSPF 区域划分时，会要求区域边界路由器的性能较强一些。

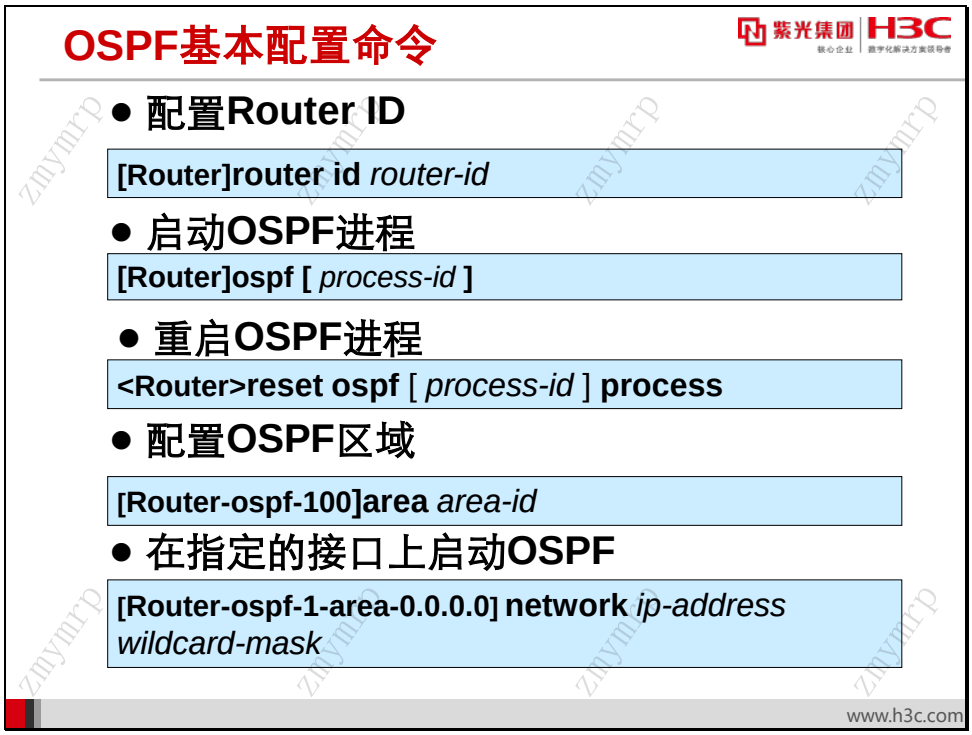


如图所示，区域 1 和区域 10.0.0.1 只向区域 0（骨干区域）发布自己区域的 LSA，而区域 0 则必须负责将其自身 LSA 向其他区域发布，并且负责在非骨干区域之间传递路由信息。为进

一步减少区域间 LSA 的数量，OSPF 区域边界路由器可以执行路由聚合，即区域边界路由器只发布一个包含某一区域内大多数路由或所有路由的网段路由。如在区域 10.0.0.1 内，所有路由器的 IP 地址都在 20.1.0.0/16 网段范围内，那么可以在连接区域 0 和区域 10.0.0.1 的区域边界路由器上配置路由聚合，让其在向区域 0 发布区域 10.0.0.1 的 LSA 时，只描述 20.1.0.0/16 网段即可，不需要具体描述区域 10.0.0.1 内的 20.1.2.0/24、20.1.0.0/24 等网段的 LSA。这样不仅大大减少了区域间传递的 LSA 的数量，还能降低整个 OSPF 自治系统内路由器维护 LSDB 数据库的资源要求，降低 SPF 算法计算的复杂度。

26.4 配置OSPF

26.4.1 OSPF 基本配置命令



在系统视图下使用命令 **ospf** *process-id* 可以启动 OSPF 进程并进入此进程的配置视图。参数 *process-id* 为进程号。一台路由器上可以同时启动多个 OSPF 进程，系统用进程号区分它们。用 **undo ospf** *process-id* 命令则可以关闭指定的 OSPF 进程并删除其配置。

在系统视图下使用命令 **router id** 可以对该路由器上所有的 OSPF 进程配置 Router ID。

如果不配置 Router ID，路由器将自动选择其某一接口的 IP 地址作为 Router ID。由于这种方式下 Router ID 的选择存在一定的不确定性，不利于网络运行和维护，通常不建议使用。

为方便OSPF区域规划和问题排查，一般建议将某一Loopback接口地址配置为Router ID。

不论是手工配置或自动选择的 Router ID，都在 OSPF 进程启动时立即生效。生效后如果更改了 Router ID 或接口地址，则只有重新启动 OSPF 协议或重启路由器后才会生效。

在用户视图下使用命令 **reset ospf** *process-id* **process** 可以重启指定的 OSPF 进程。

OSFP 路由器至少必须属于一个区域，故在 OSPF 进程启动后，应首先划分区域。

在 OSPF 视图下用命令 **area** *area-id* 配置一个区域并进入此区域视图；用 **undo area** *area-id* 命令删除一个区域。

参数 *area-id* 标识 OSPF 区域 ID，既可以是一个十进制数字，也可以是一个形如 IP 地址的点分十进制的数字。路由器允许用户使用这两种方式进行配置，但仅以点分十进制数字的方

式显示用户配置的区域。例如当用户配置为 **area** 256 时，路由器显示出用户配置的区域为 **area** 0.0.1.0。

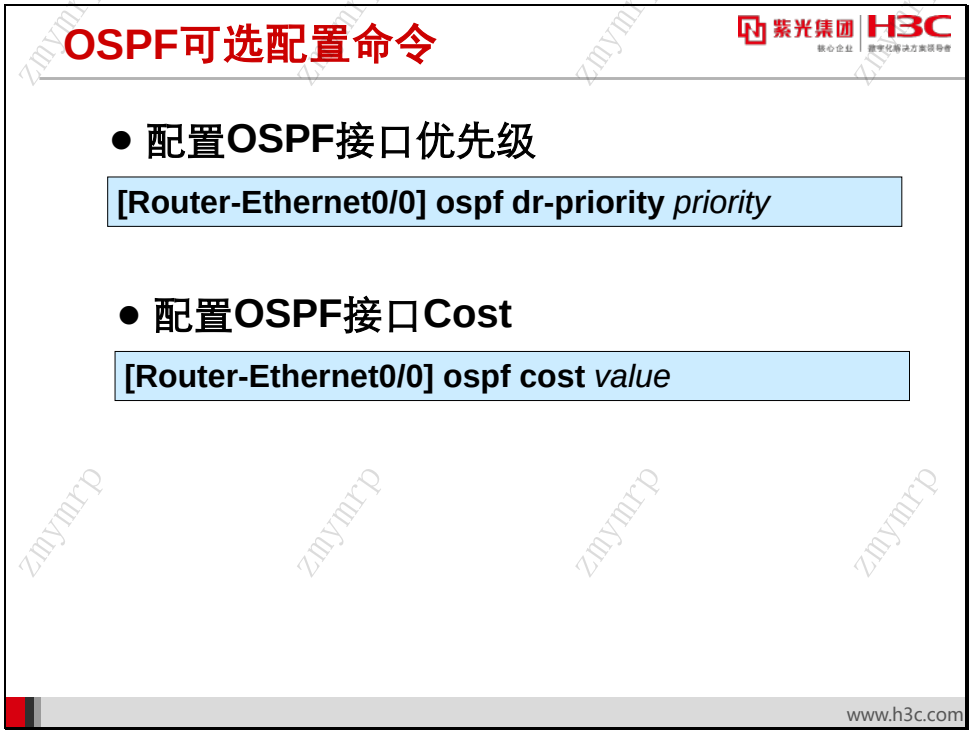
配置区域后，需要将路由器的接口加入适当的 OSPF 区域，使该接口可以执行该区域内的邻居发现、邻接关系建立、DR/BDR 选举、LSA 通告等行为，也使该接口的 IP 网段信息能通过 LSA 发布出去。一个接口只能加入一个区域。

在区域视图下使用 **network** *ip-address wildcard-mask* 命令将指定的接口加入该区域。该命令可以一次在一个区域内配置一个或多个接口运行 OSPF 协议。凡是主 IP 地址处于 *ip-address* 和 *wildcard-mask* 参数共同规定的网络范围内的接口均被加入相应的 OSPF 区域并启动 OSPF。其中参数 *ip-address* 指定一个网络地址；而参数 *wildcard-mask* 为 32 位二进制通配符掩码的点分十进制表示，其化为二进制后若某位为 0，表示必须比较 *ip-address* 和接口地址中与该位对应的位，为 1 表示不比较 *ip-address* 和接口地址中与该位对应的位。若 *ip-address* 和接口地址中所有须比较的位均匹配，则该接口被加入该区域并启动 OSPF。

在区域视图下用 **undo network** *ip-address wildcard-mask* 命令将指定的接口由该区域删除。

完成上述的命令配置后，OSPF 即可工作。

26.4.2 OSPF 可选配置命令

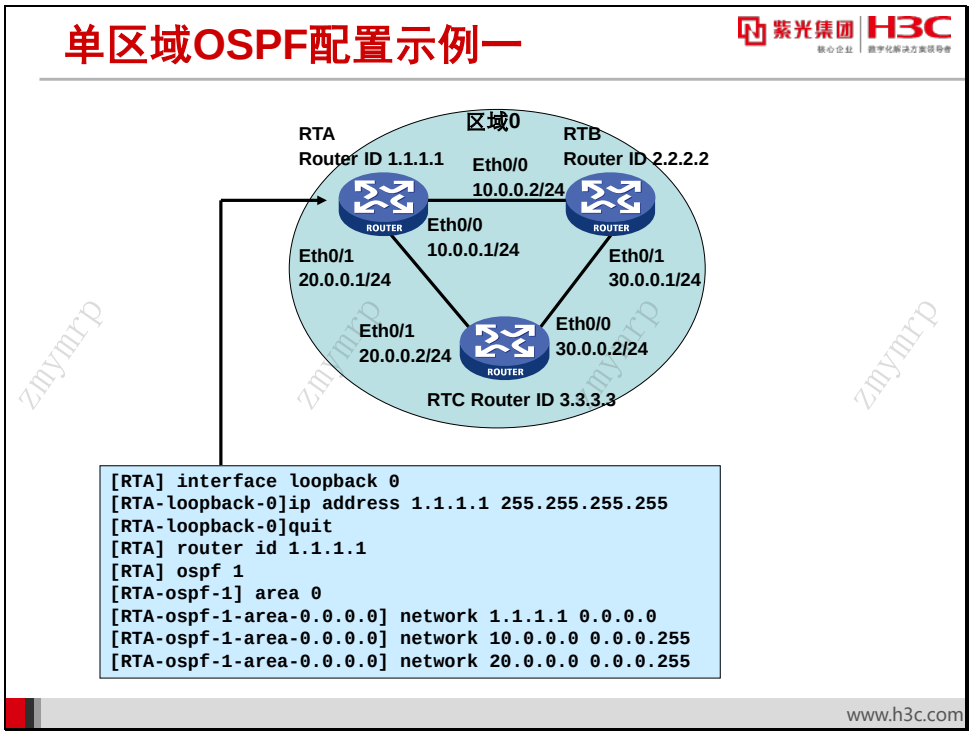


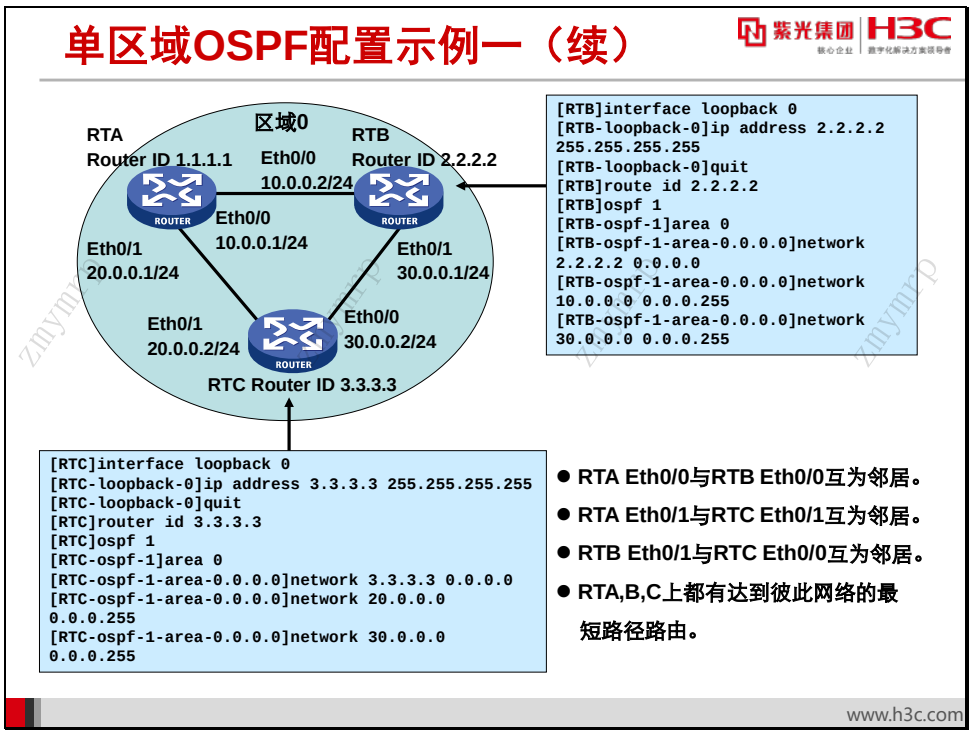
对于广播型网络来说，DR/BDR 选举是 OSPF 路由器之间建立邻接关系时很重要的步骤。OSPF 路由器的优先级对 DR/BDR 选举具有重要的作用。同样，启动 OSPF 的接口的 Cost 值直接影响到路由器计算路由过程。通常直接使用接口默认的 dr-priority 和 Cost 值即可，但如果

想人工控制 OSPF 路由器间的 DR 和 BDR 选举，或实现路由备份等，可以 OSPF 接口下配置在 **ospf dr-priority** *priority* 命令修改 dr-priority 和 Cost 值；用 **undo ospf dr-priority** 命令恢复 OSPF 接口默认优先级。

在 OSPF 接口下用命令 **ospf cost** *value* 可以直接指定 OSPF 的接口 Cost 值；用 **undo ospf cost** 命令可恢复 OSPF 接口默认 Cost 值。OSPF 路由器计算路由时，只关心路径单方向的 Cost 值，故改变一个接口的 Cost 值，只对从此接口发出数据的路径有影响，不影响从这个接口接收数据的路径。

26.4.3 单区域 OSPF 配置示例一





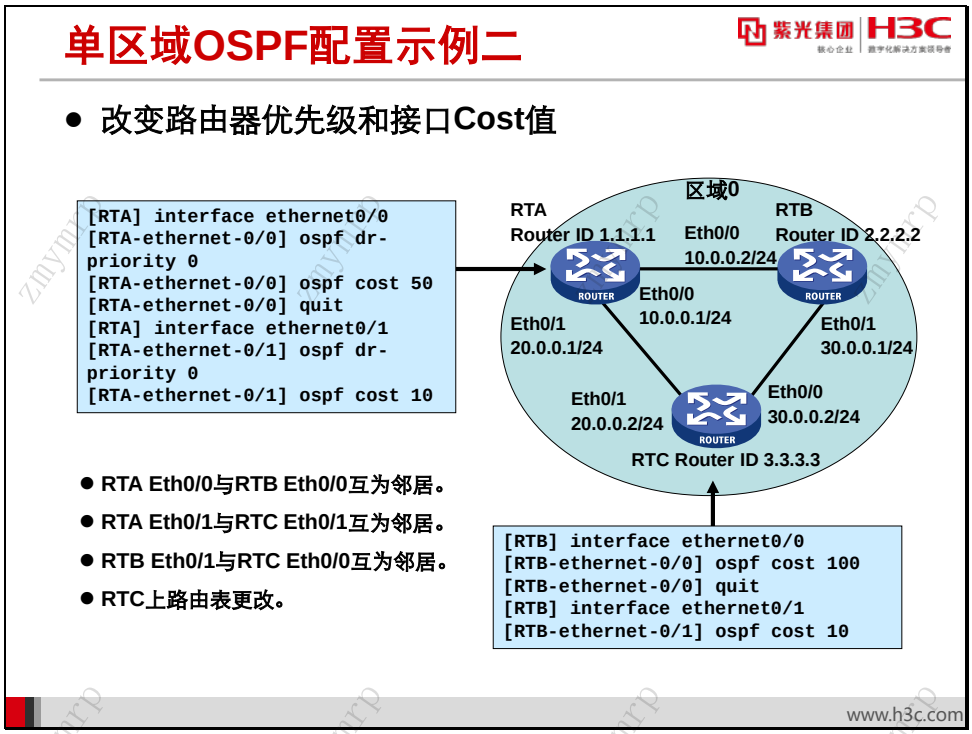
在本例中，区域 0 具有三台路由器 RTA、RTB 和 RTC，它们彼此连接。

将 RTA 上的 loopback 接口 0 的 IP 地址 1.1.1.1 设置为 RTA 的 Router ID，将 RTA 所有接口的都加入 OSPF 的区域 0。将 RTB 上的 loopback 接口 0 的 IP 地址 2.2.2.2 设置为 RTB 的 Router ID，将 RTC 上的 loopback 接口 0 的 IP 地址 3.3.3.3 设置为 RTC 的 Router ID。完成上述配置后，由于 RTA 的 Ethernet0/0 与 RTB 的 Ethernet0/0 共享同一条数据链路，并且在同一个网段内，故它们互为邻居，假设 RTA 的 OSPF 先启动，那么 RTA 的 Ethernet0/0 会被选举为 RTA 与 RTB 之间网络的 DR，假设 RTA 和 RTB 的 OSPF 同时启动，根据优先级相同时 Router ID 大的优先的原则，RTB 的 Ethernet0/0 会被选举为 RTA 与 RTB 之间网络的 DR。

同理，RTA 的 Ethernet0/1 与 RTC 的 Ethernet0/1 互为邻居，假设 RTA 的 OSPF 先启动，那么 RTA 的 Ethernet0/1 会被选举为 RTA 与 RTC 之间网络的 DR，假设 RTA 和 RTC 的 OSPF 同时启动，RTC 的 Ethernet0/1 会被选举为 RTA 与 RTB 之间网络的 DR。RTC 的 Ethernet0/0 与 RTB 的 Ethernet0/1 互为邻居，假设 RTC 的 OSPF 先启动，那么 RTC 的 Ethernet0/1 会被选举为 RTC 与 RTB 之间网络的 DR，假设 RTC 和 RTB 的 OSPF 同时启动，RTB 的 Ethernet0/0 会被选举为 RTA 与 RTB 之间网络的 DR。

在 RTC 路由表上将记录到达地址 1.1.1.1/32 网段出接口为 Ethernet0/1，到达地址 2.2.2.2/32 网段出接口为 Ethernet0/0。

26.4.4 单区域 OSPF 配置示例二

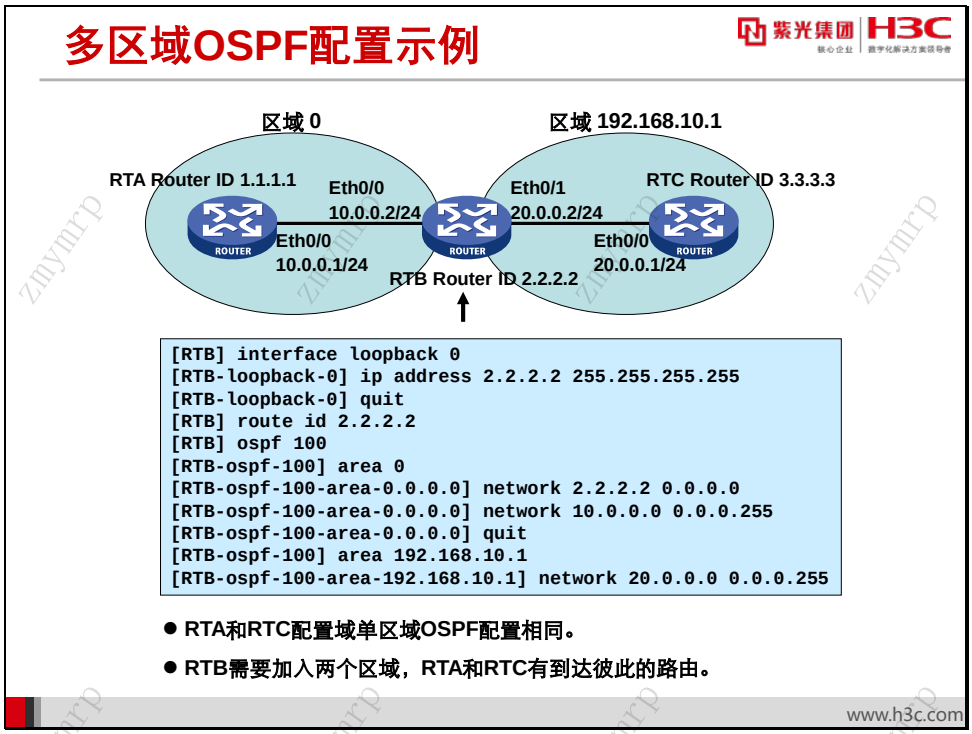


本例在上一例基础上修改了 RTA 和 RTC 的 OSPF 接口配置，在 RTA 的 Ethernet0/0 配置接口优先级为 0，接口 Cost 为 50；在 RTA 的 Ethernet0/1 上配置接口优先级为 0，接口 Cost 为 10。在 RTC 的 Ethernet0/0 上配置接口 Cost 为 100，在 RTC 的 Ethernet0/1 上配置接口开销为 10。

完成以上的配置后，将 RTA、RTB、RTC 的所有物理接口 shutdown，然后先将 RTA 的接口 UP，再将 RTB 的接口 UP，最后将 RTC 的接口 UP。由于 RTA 的 Ethernet0/0 和 Ethernet0/1 的接口优先级为 0，它们都不具备 DR/BDR 的选举权，故在 RTA 和 RTB 之间的网络上 RTB 为 DR，在 RTA 和 RTC 之间的网络上 RTC 为 DR，在 RTB 和 RTC 之间的网络上，由于 RTB 先启动，RTB 将作为该网络的 DR。

在 RTC 路由表上将记录到达地址 1.1.1.1/32 网段出接口为 Ethernet0/1，到达地址 2.2.2.2/32 网段出接口也为 Ethernet0/1，因为 RTC 从 Ethernet0/1 出发到达 RTA、再由 RTA 的 Ethernet0/0 出发到达 RTB 的 Cost 为 60，比从 RTC 直接从 Ethernet0/0 出发到达 RTB 的开销 100 要低。

26.4.5 多区域 OSPF 配置示例

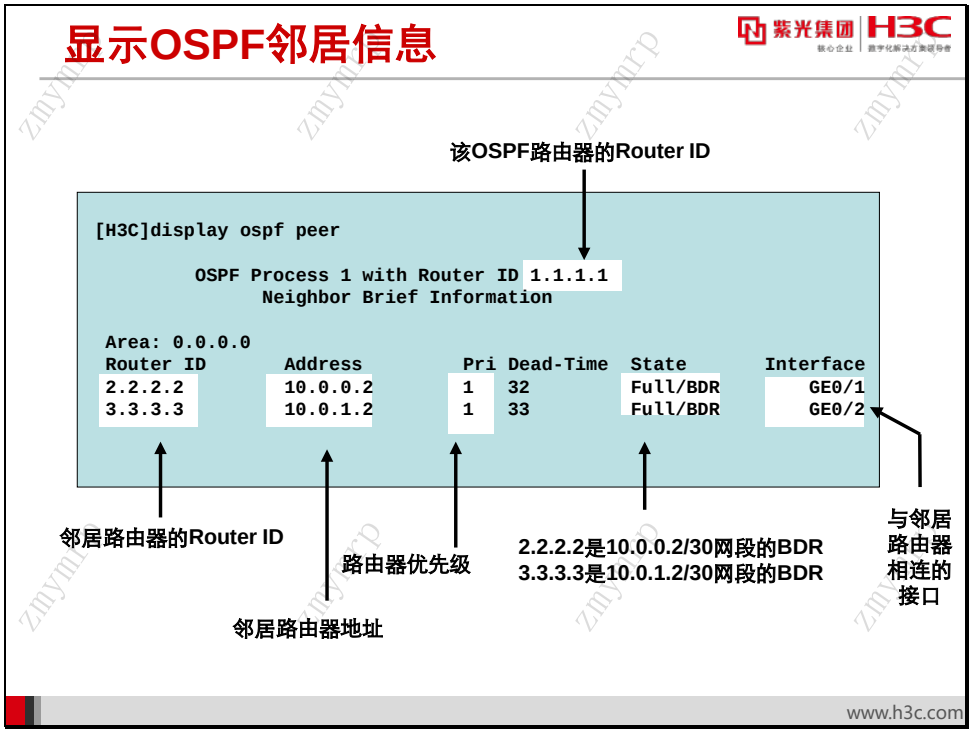


本例中，RTA 和 RTC 的配置与单区域 OSPF 的配置相同，重点集中在 RTB 的配置上。RTB 作为区域边界路由器，需要同时加入 RTA 和 RTC 所在的区域，需要注意的是在 RTB 指定接口加入 OSPF 区域 0 的时候，不在该区域的接口 Ethernet0/1 的地址不能加入区域 0。同样，接口 Ethernet0/0 的地址不能加入区域 192.168.0.1。

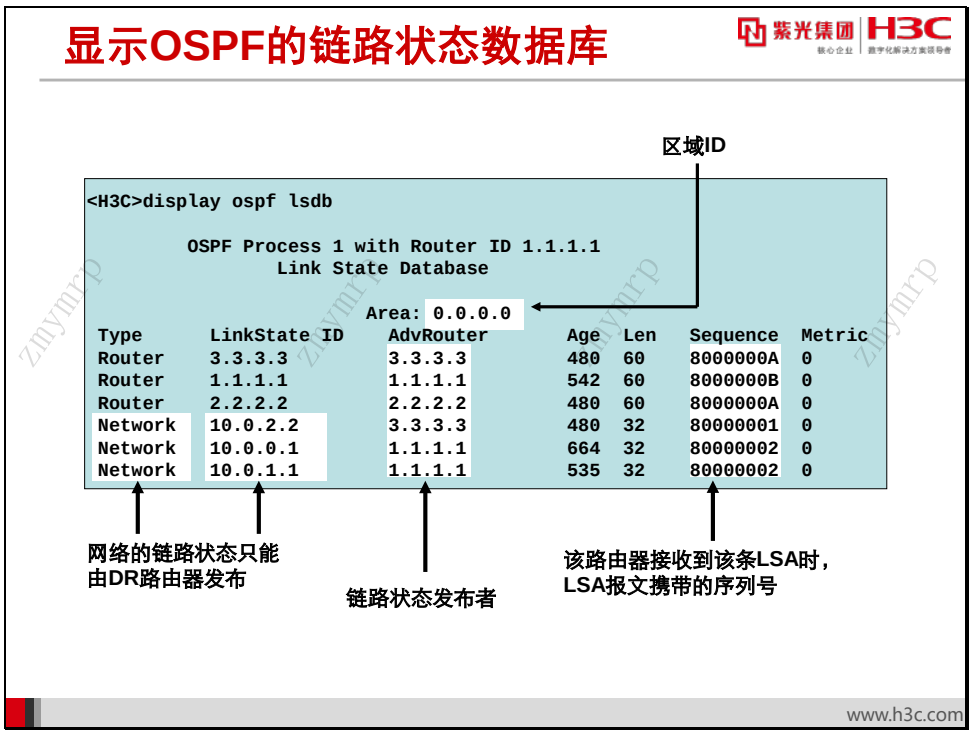
26.5 OSPF信息显示与调试

为了便于在OSPF环境下迅速定位故障，系统为用户提供了功能强大的显示和调试的工具。

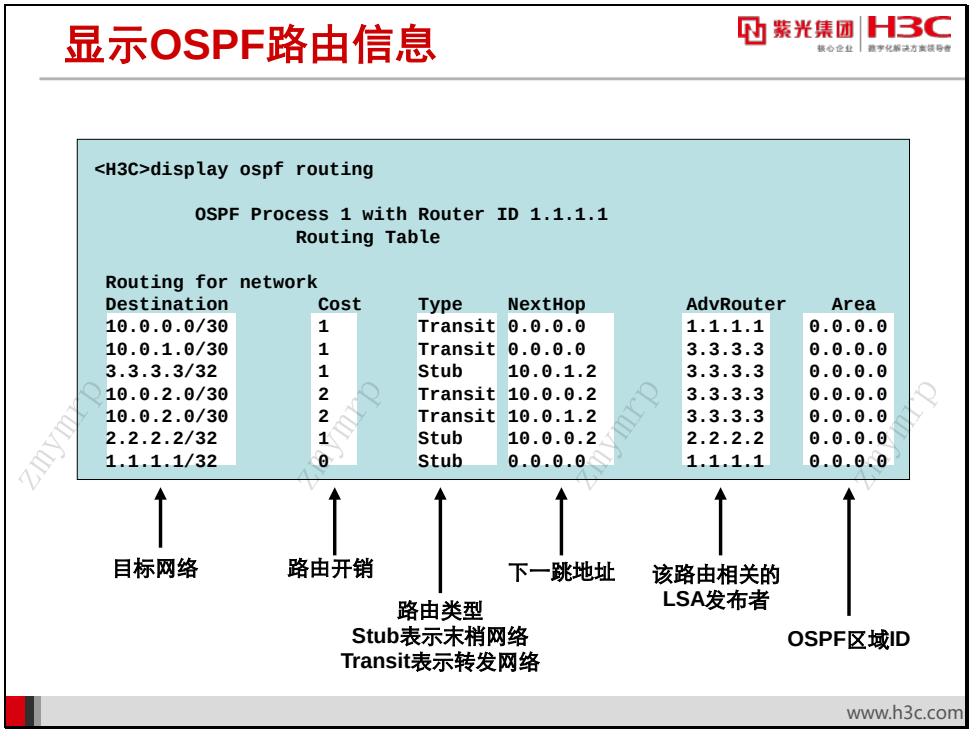
26.5.1 OSPF 信息显示



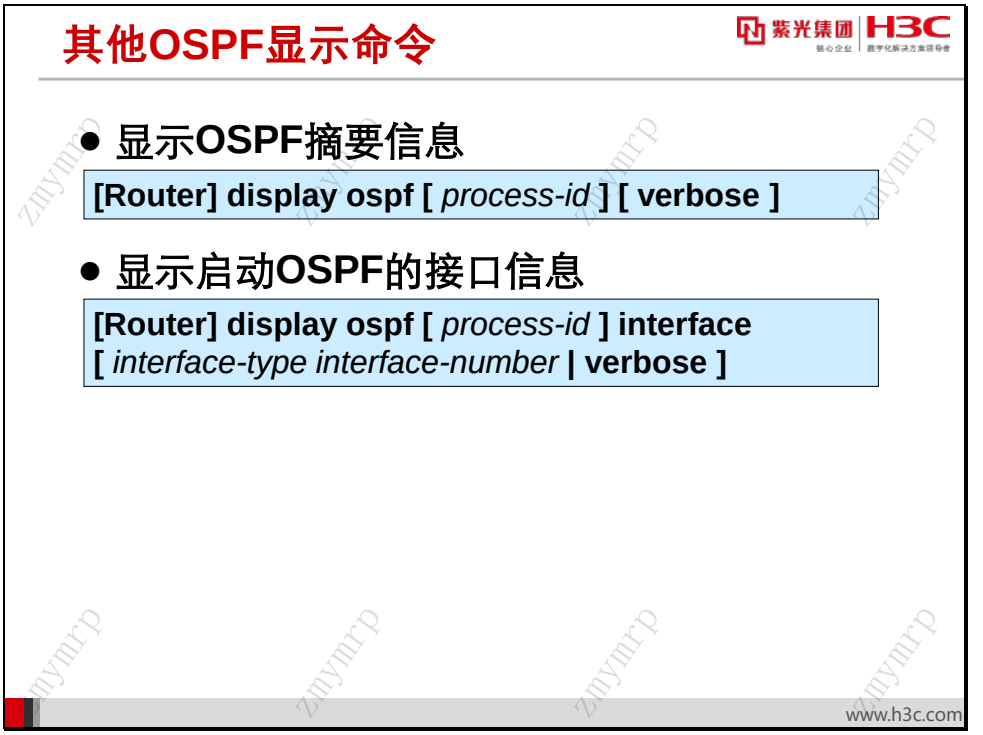
通过 **display ospf peer** 命令可以查看路由器的 OSPF 邻居关系。在广播型网络里，路由器只有与 DR 和 BDR 的邻居状态能够达到 Full 状态，Full 状态说明该网络的 OSPF 路由器的链路状态已经同步。



通过 **display ospf lsdb** 命令，可以查看路由器的链路状态数据库，OSPF 区域内的各 OSPF 路由器的链路状态数据库应该都是一样的。

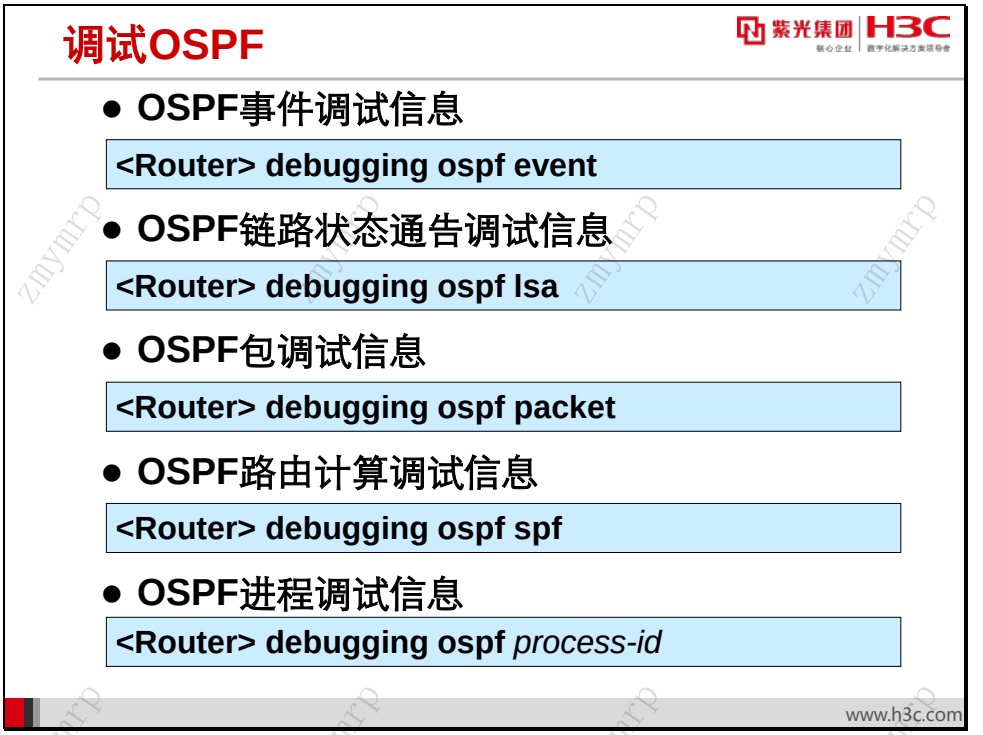


通过 **display ospf routing** 命令，可以查看路由器的 OSPF 路由情况，并不是所有的 OSPF 路由就一定会被路由器使用，路由器还需要权衡其他协议提供的路由及路由器接口连接方式等，如果 OSPF 提供的路由与直连路由相同，路由器会选择直连路由加入全局路由表。



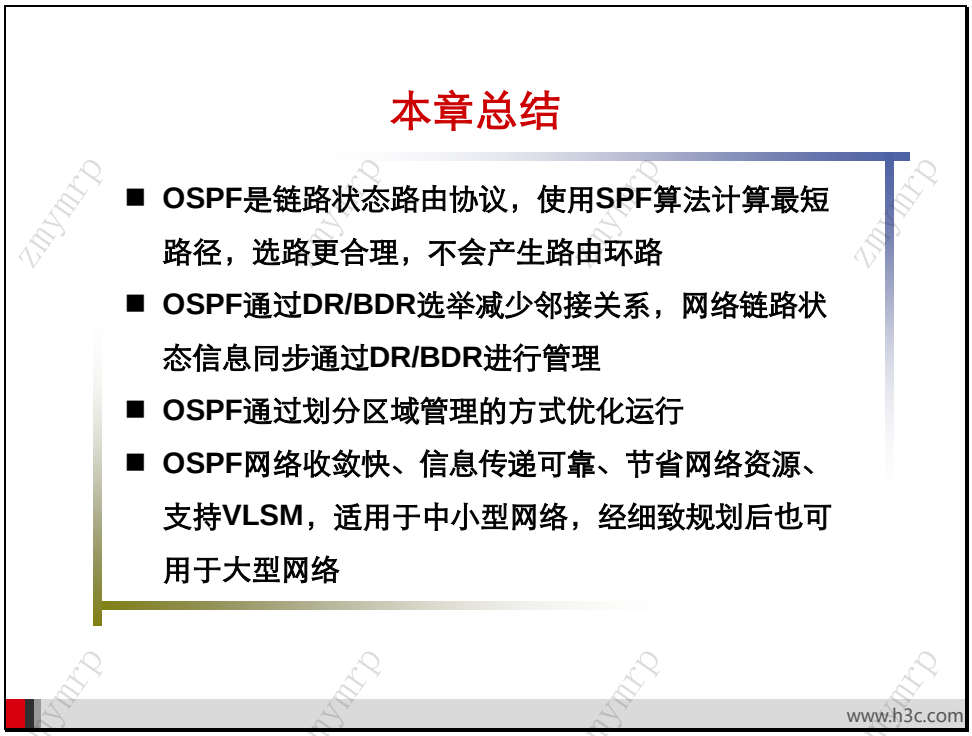
另外，可以通过 **display ospf [** *process-id* **] [verbose]**、**display ospf [** *process-id* **] interface [** *interface-type interface-number* **| verbose]**查看其他 OSPF 信息。

26.5.2 调试 OSPF



在用户视图下输入 **debugging ospf event**、**debugging ospf lsa**、**debugging ospf packet**、**debugging ospf spf** 和 **debugging ospf** *process-id* 命令，可以调试 OSPF。

26.6 本章总结


本章总结
OSPF是链路状态路由协议，使用SPF算法计算最短路径，选路更合理，不会产生路由环路
OSPF通过DR/BDR选举减少邻接关系，网络链路状态信息同步通过DR/BDR进行管理
OSPF通过划分区域管理的方式优化运行
OSPF网络收敛快、信息传递可靠、节省网络资源、支持VLSM，适用于中小型网络，经细致规划后也可用于大型网络
www.h3c.com