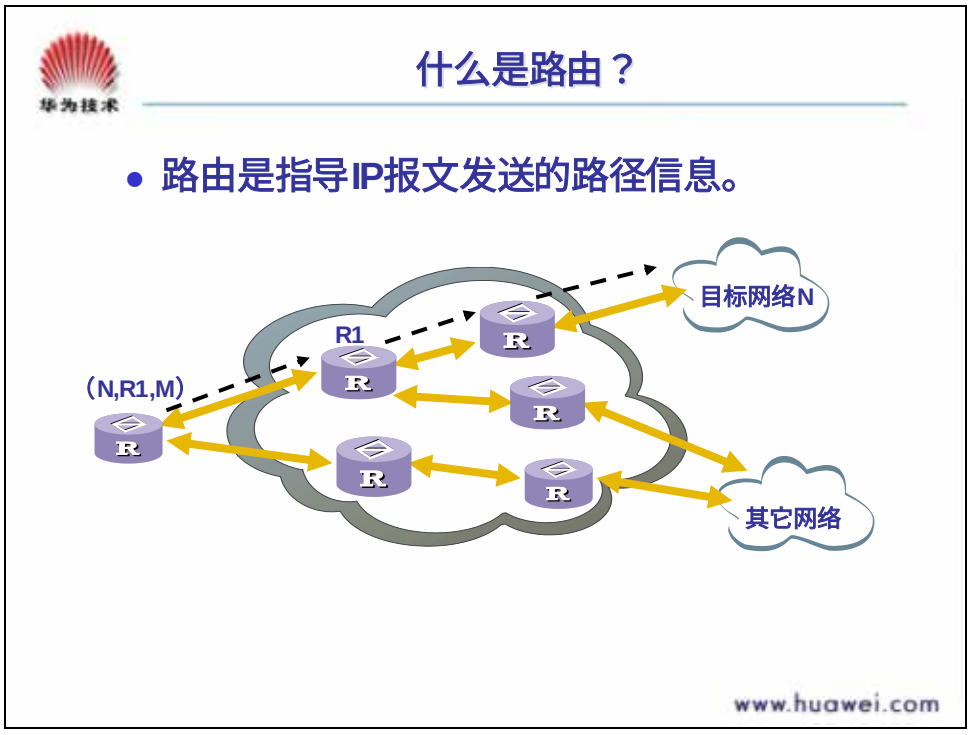
7.1 路由及路由表

7.1.1 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。

在互连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自己本站数据包通过最优的路径转发，通过多个路由器一站一站的接力将数据包通过最优最佳路径转发到目的地，当然有时候由于实施一些路由策略数据包通过的路径并不一定是最佳路由。

根据路由的目的地不同，可以划分为：

- 子网路由：目的地为子网
- 主机路由：目的地为主机

另外，根据目的地与该路由器是否直接相连，又可分为：

- 直接路由：目的地所在网络与路由器直接相连
- 间接路由：目的地所在网络与路由器不是直接相连

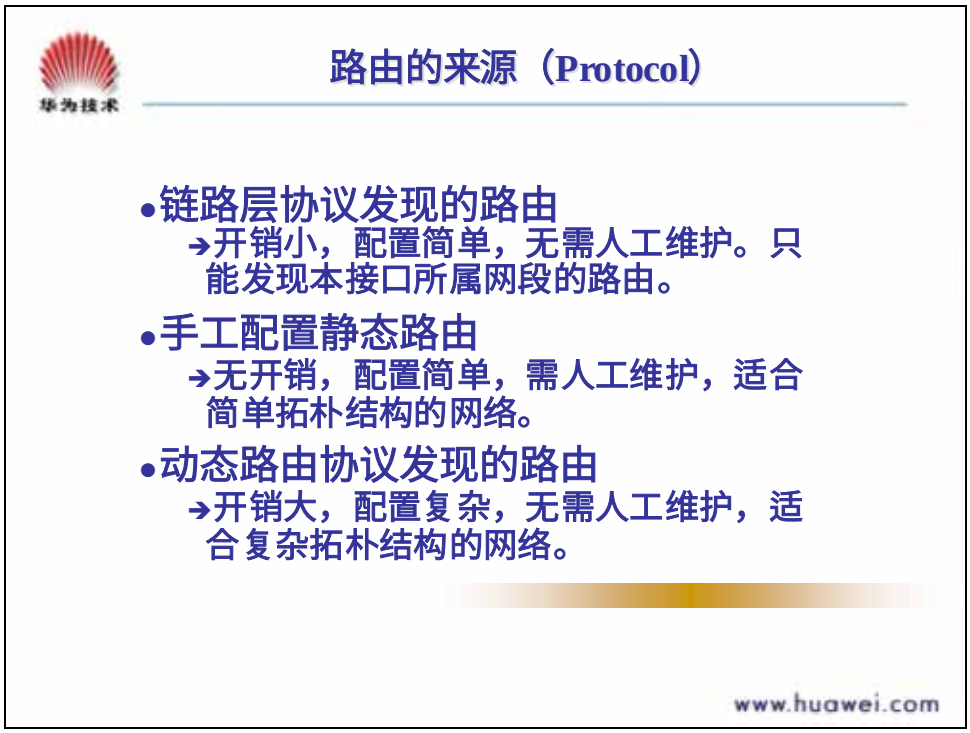
7.1.2 通过路由表进行选路

路由器转发数据包的关键是路由表。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关键项：

- 目的地址（Destination）：用来标识 IP 包的目的地址或目的网络。
- 网络掩码（Mask）：与目的地址一起来标识目的主机或路由器所在的网段的地址。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。
- 输出接口（Interface）：说明 IP 包将从该路由器哪个接口转发。
- 下一跳 IP 地址（Nexthop）：说明 IP 包所经由的下一个路由器的接口地址。

7.1.3 路由表中路由的来源



在路由表中有一个 Protocol 字段：指明了路由的来源，即路由是如何生成的。路由的来源主要有 3 种：

- 链路层协议发现的路由（Direct）

开销小，配置简单，无需人工维护，只能发现本接口所属网段拓扑的路由。

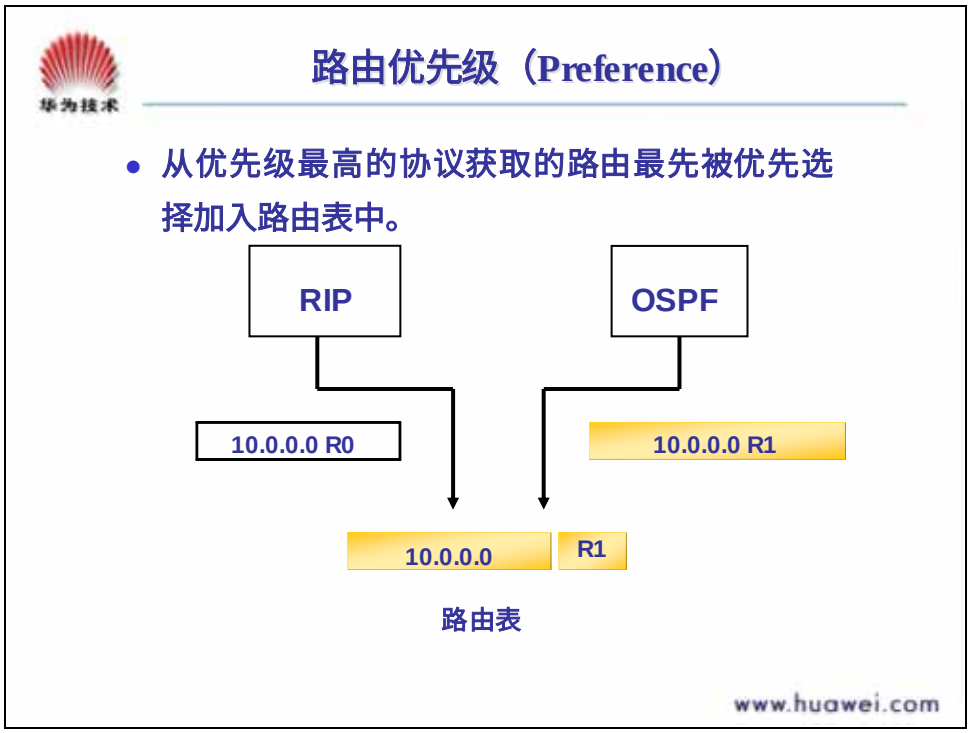
- 手工配置的静态路由（Static）

静态路由是一种特殊的路由，它由管理员手工配置而成。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。

- 动态路由协议发现的路由（RIP、OSPF ……）

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，让其自动发现和修改路由，无需人工维护，但动态路由协议开销大，配置复杂。

7.1.4 路由优先级



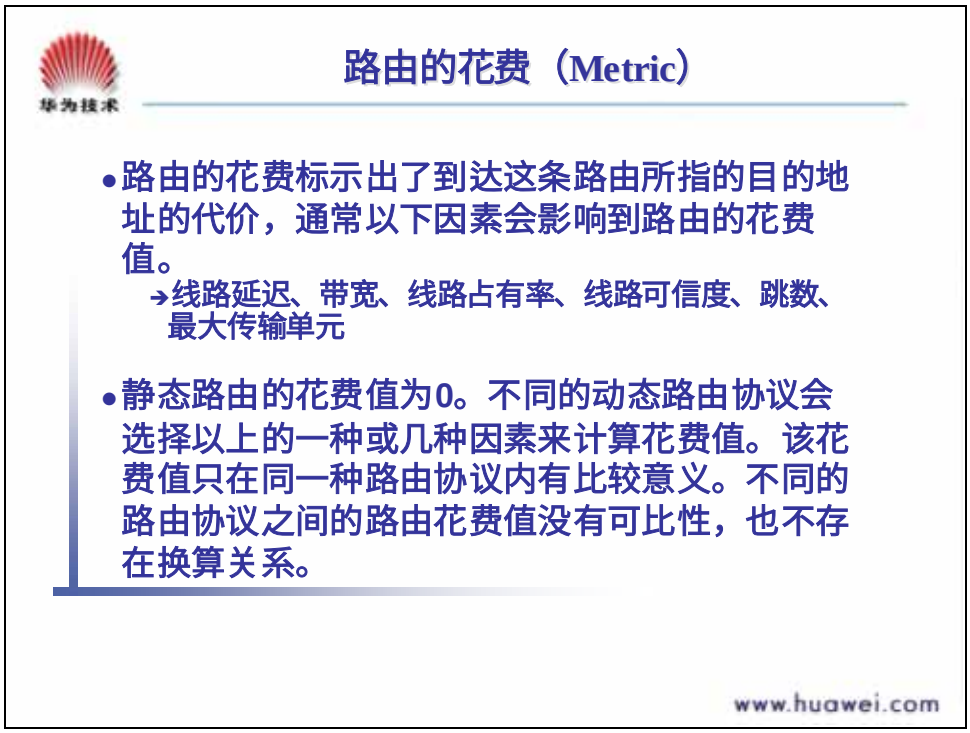
到相同的目的地，不同的路由协议（包括静态路由）可能会发现不同的路由，但并非这些路由都是最优的。事实上，在某一时刻，到某一目的地的当前路由仅能由唯一的路由协议来决定。这样，各路由协议（包括静态路由）都被赋予了一个优先级，这样，当存在多个路由信息源时，具有较高优先级（数值越小表明优先级越高）的路由协议发现的路由将成为最优路由，并被加入路由表中。

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。华为 Quidway 路由器的缺省优先级如下表所示。其中：0 表示直接连接的路由，255 表示任何来自不可信源端的路由。

路由协议或路由种类	相应路由的优先级
DIRECT	0
OSPF	10
STATIC	60
RIP	100
IBGP	130
OSPF ASE	150
EBGP	170
UNKNOWN	255

除了直接路由（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。

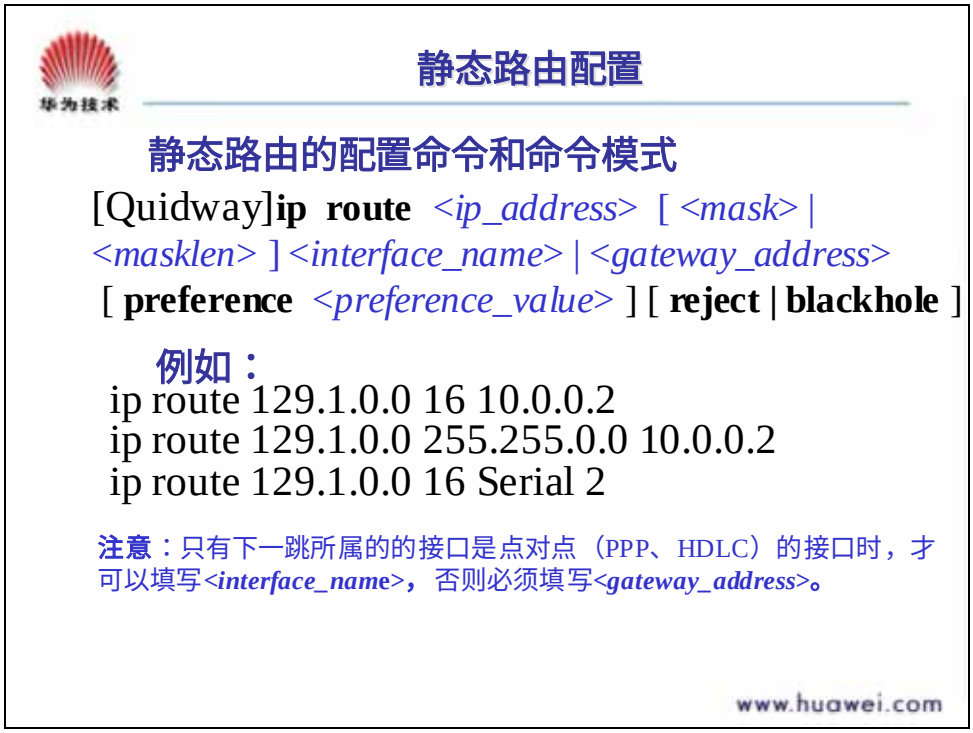
7.1.5 路由的花费



路由的花费（metric）标识出了到达这条路由所指的目的地址的代价，通常路由的花费值会受到线路延迟、带宽、线路占有率、线路可信度、跳数、最大传输单元等因素的影响，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花费值（如 RIP 用跳数来计算花费值）。该花费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花费值没有可比性，也不存在换算关系。静态路由的花费值为 0。

7.2 静态路由及配置

7.2.1 静态路由配置



在组网结构比较简单的网络中，只需配置静态路由就可以使路由器正常工作，仔细设置和使用静态路由可以改进网络的性能，并可为重要的应用保证带宽。

还有一种静态路由类型为称为接口静态路由，它用于表示那些直接连接到路由器接口上的目的网络。接口静态路由优先级是 0，这意味着它是直接连接网络的路由。

静态路由还有如下的属性：

- 可达路由：正常的路由都属于这种情况，即 IP 报文按照目的地标示的路由被送往下一跳，这是静态路由的一般用法。
- 目的地不可达的路由：当到某一目的地的静态路由具有“ reject ”属性时，任何去往该目的地的 IP 报文都将被丢弃，并且通过 ICMP 消息通知源主机目的地不可达。
- 目的地为黑洞的路由：当到某一目的地的静态路由具有“ blackhole ”属性时，任何去往该目的地的 IP 报文都将被丢弃。同“ reject ”的区别是不向源主机发送任何消息。

其中各参数的解释如下：

（1）<ip_address>[<mask>|<masklen>]：目的 IP 地址和掩码

IP 地址为点分十进制格式，掩码可以用点分十进制表示，也可用掩码长度（即掩码中‘1’的位数）表示。

（2）<interface_name>|<gateway_address>：发送接口或下一跳地址

在配置静态路由时，可指定发送接口 *interface-name*，也可指定下一跳地址 *gateway-address*，是指定发送接口还是指定下一跳地址要视具体情况而定。

实际上，所有的路由项都必需明确下一跳地址。IP 在发送报文时，首先根据报文的目的地址寻找路由表中与之匹配的路由。只有路由指定了下一跳地址，链路层才能通过下一跳 IP 地址找到对应的链路层地址，然后按照该地址将报文转发。

在以下几种情况下可以指定发送接口：

- 对于支持网络地址到链路层地址解析的接口（如以太网口支持 ARP），当 ip-address 和 mask（或 mask-length）指定了一个主机地址，而且该目的地址就在该接口的直接连接网络中，这时可以指定发送接口。
- 对于点到点接口，指定发送接口即隐含指定了下一跳地址，这时认为与该接口相连的对端接口地址就是路由的下一跳地址。如串口封装 PPP 协议，通过 PPP 协商获取对端的 IP 地址，这时可以不用指定下一跳地址，只需指定发送接口即可。
- 对于 NBMA 接口（如封装 X.25 或帧中继的接口、拨号口等），支持点到多点，这时除了配置 IP 路由外，还需在链路层建立二次路由，即 IP 地址到链路层地址的映射（如 dialer map ip、x.25 map ip 或 frame-relay map ip 等）。这种情况下配置静态路由就不能指定发送接口，而应配置下一跳 IP 地址。

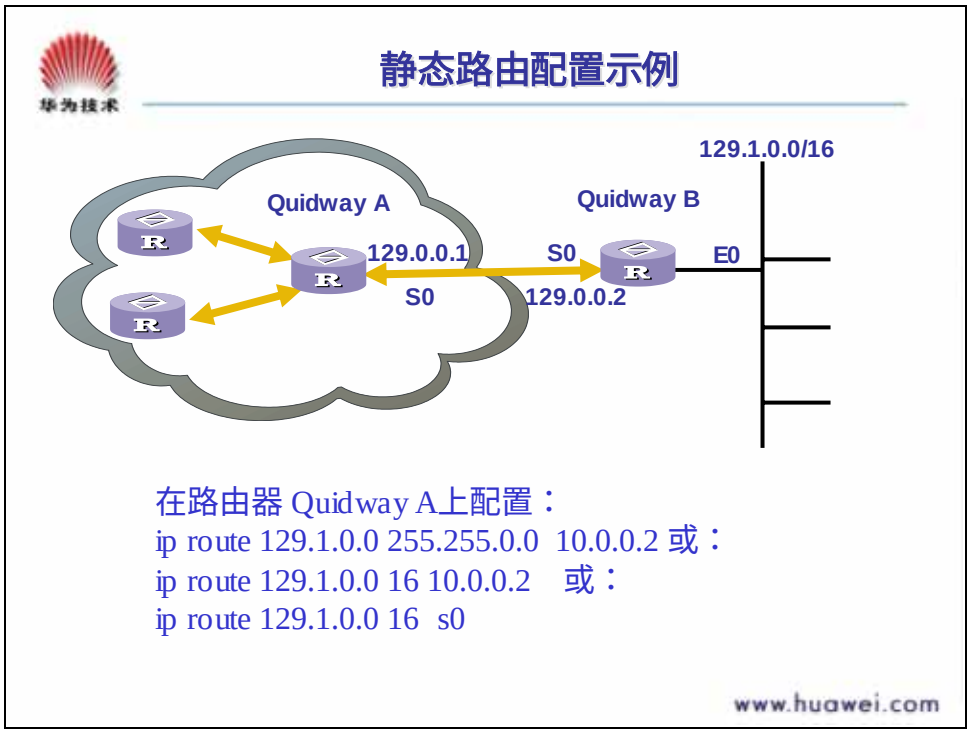
（3）<preference_value>：优先级

对优先级 **preference** 的不同配置，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。在同一命令中优先级可以多次输入，但只有最后一个有效。

（4）其它参数

属性 **reject** 和 **blackhole** 分别指明不可达路由和黑洞路由。

7.2.2 静态路由配置示例



在路由器 QuidwayA 上配置一条到目的网段 129.1.0.0/16 的静态路由，下一跳地址为路由器 QuidwayB 的 S0 接口的 IP 地址 10.0.0.2。如果链路的封装是 PPP 或 HDLC，也可以指定本路由器的转发接口。

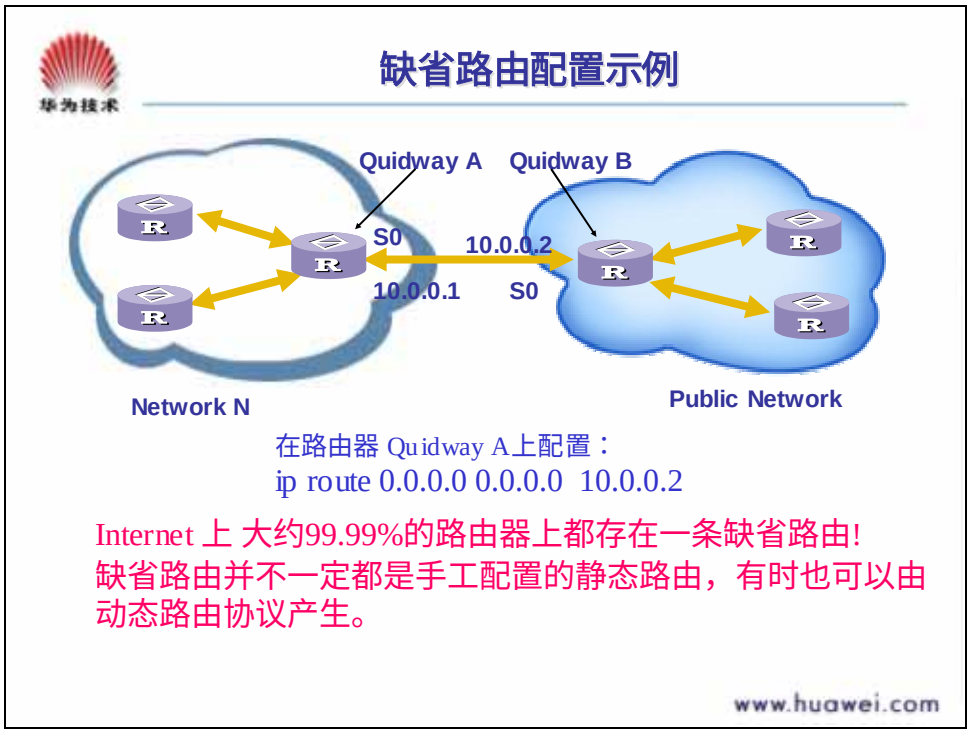
静态路由配置命令：

[QuidwayA]ip route 129.1.0.0 16 s 0 或

[QuidwayA]ip route 129.1.0.0 16 10.0.0.2 或

[QuidwayA]ip route 129.1.0.0 255.255.0.0 10.0.0.2 。

7.2.3 缺省路由的配置



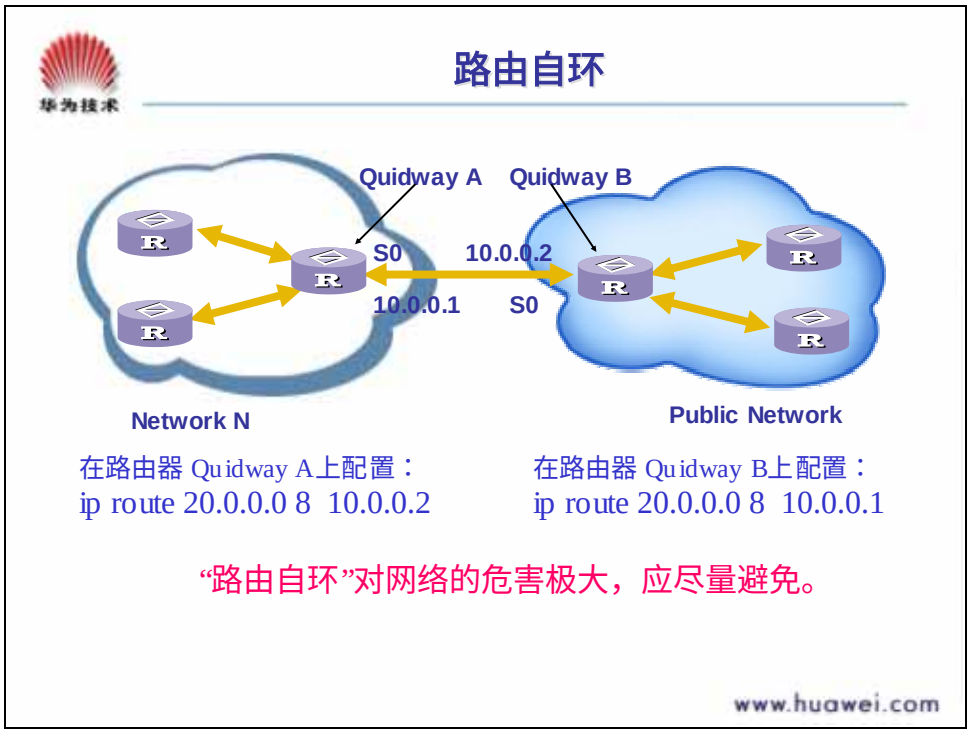
缺省路由也是一种静态路由。简单地说，缺省路由就是在没有找到匹配的路由表入口项时才使用的路由。即只有当没有合适的路由时，缺省路由才被使用。在路由表中，缺省路由以到网络 0.0.0.0（掩码为 0.0.0.0）的路由形式出现。可通过命令 display ip route 的输出看它是否被设置。如果报文的目的地址不能与路由表的任何入口项相匹配，那么该报文将选取缺省路由。如果没有缺省路由且报文的目的地址不在路由表中，那么该报文被丢弃的同时，将返回源端一个 ICMP 报文指出该目的地址或网络不可达。

缺省路由在网络中是非常有用的。在一个包含上百个路由器的典型网络中，选择动态路由协议可能耗费较大量的带宽资源，使用缺省路由意味着采用适当带宽的链路来替代高带宽的链路以满足大量用户通信的需求。

Internet 上大约 99.99%的路由器上都存在一条缺省路由！

缺省路由并不一定都是手工配置的静态路由，有时也可以由动态路由协议产生。比如 OSPF 路由协议配置了 Stub 区域的路由器会动态产生一条缺省路由。

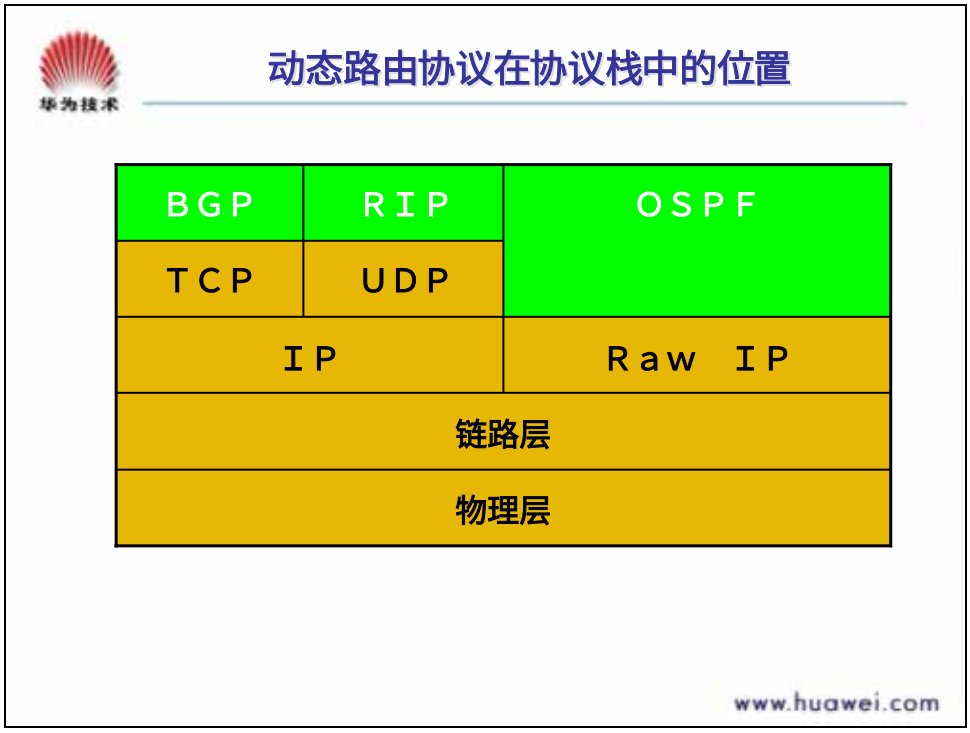
7.2.4 路由自环



“路由自环”是指某个报文从一台路由器发出，经过几次转发之后又回到初始的路由器。原因是其中部分路由器的路由表出现错误。产生的原因可能是配置静态路由有误，或者是动态路由协议错误地计算路由（虽然这种情况发生的几率很小）。当产生路由自环时，报文会在几个路由器之间循环转发，直至 TTL=0 时才被丢弃，极大地浪费了网络资源，因此应该尽量避免“路由自环”的产生。

7.3 动态路由协议概述

7.3.1 动态路由协议在协议栈中的位置



所有的动态路由协议在 TCP/IP 协议栈中都属于应用层的协议。但是不同的路由协议使用的底层协议不同。

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。

7.3.2 路由协议的基本原理

路由协议的基本原理（一）

- 动态路由协议是做什么的
  - 计算路由的．计算本地路由器到网络中其它网段的路由．
- 如何做到这一点
  - 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，由于大家都这样做，最终每台路由器都会收到网络中所有的路由信息．然后运行某种算法，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花费）

www.huawei.com

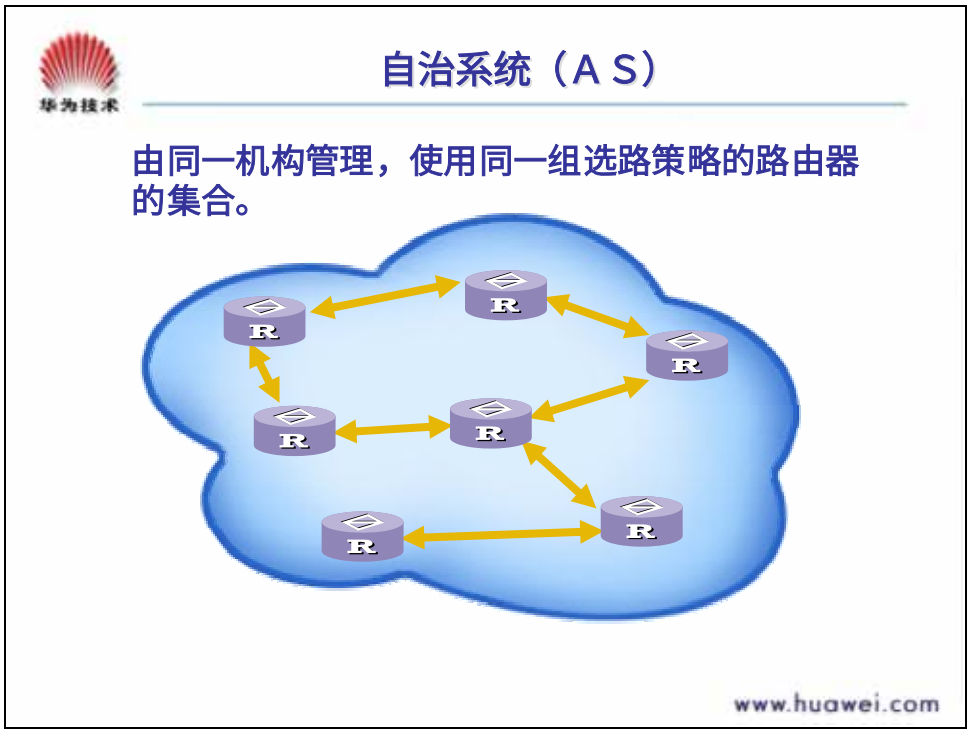
路由协议的基本原理（二）

动态路由协议是做什么的

- “天王盖地虎”—“宝塔镇河妖”
  - 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．
- “初次见面，请多关照”
  - 一台新加入的路由器应该主动把自己介绍给网段内的其它路由器．通过发送广播报文或发送给指定的路由器邻居来做到这一点．
- “好久不见，近况如何”
  - 为了能够观察到某台路由器突然失败（路由器本身故障或连接线路中断）这种异常情况，规定两台路由器之间的协议报文应该周期性地发送．

www.huawei.com

7.3.3 自治系统（AS）

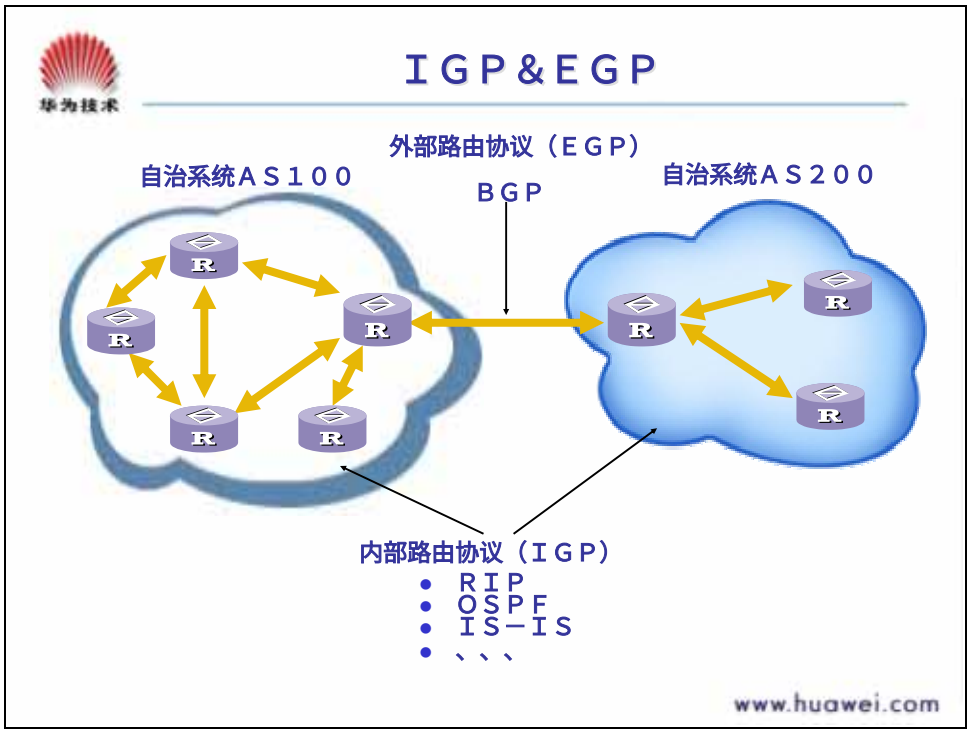


一个 AS 是一组共享相似的路由策略并在单一管理域中运行的路由器的集合。一个 AS 可以是一些运行单个 IGP（内部网关协议）协议的路由器集合，也可以是一些运行不同路由选择协议但都属于同一个组织机构的路由器集合。不管是哪种情况，外部世界都将整个 AS 看作是一个实体。

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 到 65535，其中 1 到 65411 是注册的因特网编号，65412 到 65535 是专用网络编号。

7.3.4 路由协议的分类



按照工作区域，路由协议可以分为 IGP 和 EGP：

- IGP（Interior gateway protocols）内部网关协议

在同一个自治系统内交换路由信息，RIP 和 IS-IS 都属于 IGP。IGP 的主要目的是发现和计算自治域内的路由信息。

- EGP（Exterior gateway protocols）外部网关协议

用于连接不同的自治系统，在不同的自治系统之间交换路由信息，主要使用路由策略和路由过滤等控制路由信息在自治域间的传播，应用的一个实例是 BGP。

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 距离矢量协议（Distant-Vector）和链路状态协议。距离矢量协议包括 RIP 和 BGP，链路状态协议包括 OSPF、IS-IS。

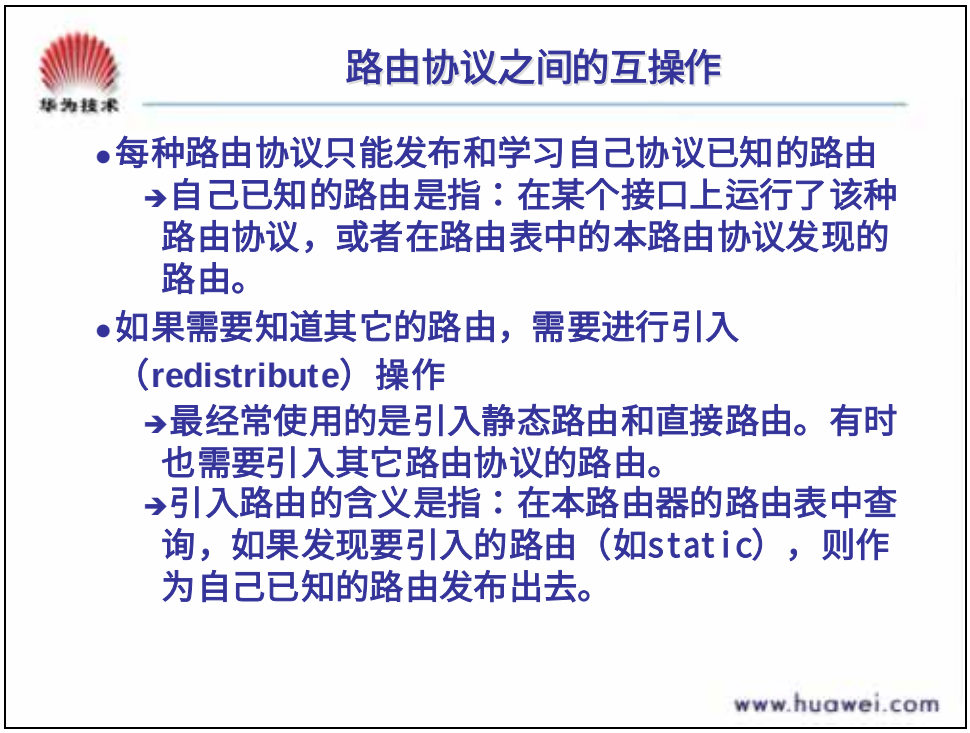
距离矢量路由协议基于贝尔曼－福特算法，使用 D-V 算法的路由器通常以一定的时间间隔向相邻的路由器发送他们完整的路由表。接收到路由表的邻居路由器将收到的路由表和自己的路由表进行比较，新的路由或到已知网络但开销（Metric）更小的路由都被加入到路由表中。相邻路由器然后再继续向外广播它自己的路由表（包括更新后的路由）。距离矢量路由器关心的是到目的网段的距离（Metric）和矢量（方向，从哪个接口转发数据）。在发送数据前，路由协议计算到目的网段的 Metric；在收到邻居路由器通告的路由时，将学到的网段信息和收到此网段信息的接口关联起来，以后有数据要转发到这个网段就使用这个关联的接口。

距离矢量路由协议的优点：配置简单，占用较少的内存和 CPU 处理时间。缺点：扩展性较差，比如 RIP 最大跳数不能超过 16 跳。

链路状态路由协议基于 Dijkstra 算法，有时被称为最短路径优先算法。L-S 算法提供比 RIP 等 D-V 算法更大的扩展性和快速收敛性，但是它的算法耗费更多的路由器内存和处理能力。D-V 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（LSA：Link State Advitisement）。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朴图，形成一个到各个目的网段的带权有向图。

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，这种方式节省了相邻路由器之间的链路带宽。部分更新只包含改变了的链路状态信息，而不是整个的路由表。

7.3.5 路由协议之间的互操作



为了在同一个互联网中支持多种路由协议，必须在这些不同的路由协议之间共享路由信息。例如从 RIP 学到的路由信息可能需要引入到 OSPF 协议中去。这种在不同路由协议中间交换路由信息的过程被称为路由引入。路由引入可以是单向的（例如将 RIP 引入 OSPF），也可以是双向的（RIP 和 OSPF 互相引入）。执行路由引入的路由器一般位于不同自治系统或者不同路由域的边界。

由于各路由协议的算法不同，不同的协议可能会发现不同的路由，因此各路由协议之间存在如何共享各自发现结果的问题。前面我们讲过，不同路由协议之间的花销不存在可比性，也不存在换算关系，所以在引入路由时必须重新设置引入路由的 Metric 值，或者使用系统默认的数值。VRP 支持将一种路由协议发现的路由引入（**import-route**）到另一种路由协议中，每种协议都有相应的路由引入机制。

路由协议的相互引入实现了不同路由信息的共享，但同时也带来了一些问题。使用多种路由协议通常会导致网络管理复杂和额外开销增大。当路由器将从一个自治系统学到的路由信息再发送回同一自治系统，就有可能会产生路由环路。另外，由于各路由协议使用不同的度量值来决定最佳路由，所以利用引入的路由信息进行路径选择有可能会导致次最佳路由。一般情况下，应尽量避免重叠使用路由协议（同一个区域内既使用 RIP，又使用 OSPF），使用不同路由协议的网络之间要有明确的边界；如果有一台以上的路由器担任路由引入点，应只在一个方向上进行路由引入，以避免路由环路和因收敛时间不一致导致的问题。如果在一个路由域中只有一台边界路由器，可以使用双向引入。

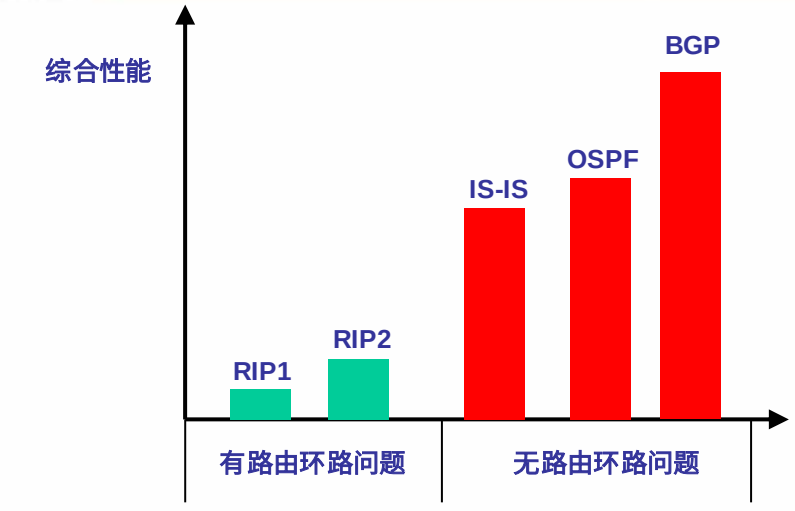
7.3.6 衡量路由协议的一些性能指标

衡量路由协议的一些性能指标

- 正确性
  能够正确找到最优的路由，且无自环。
- 快收敛
  当网络的拓朴结构发生变化之后，能够迅速在自治系统中作相应的路由改变。
- 低开销
  协议自身的开销（内存、CPU、网络带宽）最小。
- 安全性
  协议自身不易受攻击，有安全机制。
- 普适性
  适应各种拓朴结构和规模的网络。

www.huawei.com

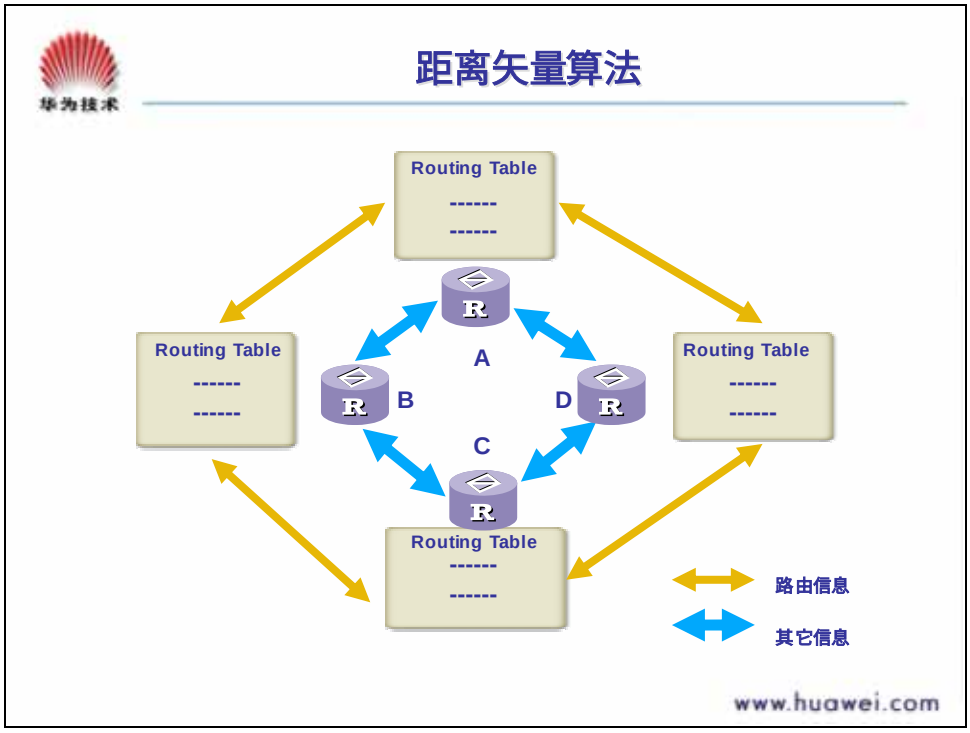
现有路由协议的性能比较



www.huawei.com

7.4 距离矢量路由协议概述

7.4.1 距离矢量算法基本原理



距离矢量（DISTANCE-VECTOR，简称 D-V）算法（也称 BELLMAN-FORD 算法）周期性地将路由表信息的拷贝在路由器之间传送。当网络拓扑变化时，也会将更新信息及时传送给路由器。每一个路由器只能接收到网络中相邻路由器的路由表，就如图所示，路由器 B 接收到相邻路由器 A 的信息，通过增加一个距离矢量数（例如一个跳数）来增大距离矢量，然后将更新的路由表信息传送给相邻路由器 C。这种逐步过程发生在相邻路由器之间。

距离矢量算法的数学模型如下：

我们用 D（i，j）来表示从实体 i 到 j 的最佳路由的 Metric，i、j 可以是系统中的任意一对实体，用 d（i，j）来表示单个跳数的花费，也就是从 i 直接到 j 的花费，如果 i 与 j 不是直接相邻的，则 d（i，j）为无穷大。这样任意两个实体间的最佳 Metric 可以表示如下：

D（i，j）＝0　　对所有的 i

$$D(i,j) = \min_{k} [D(i,k) + d(k,j)]$$　　i 不等于 k 时

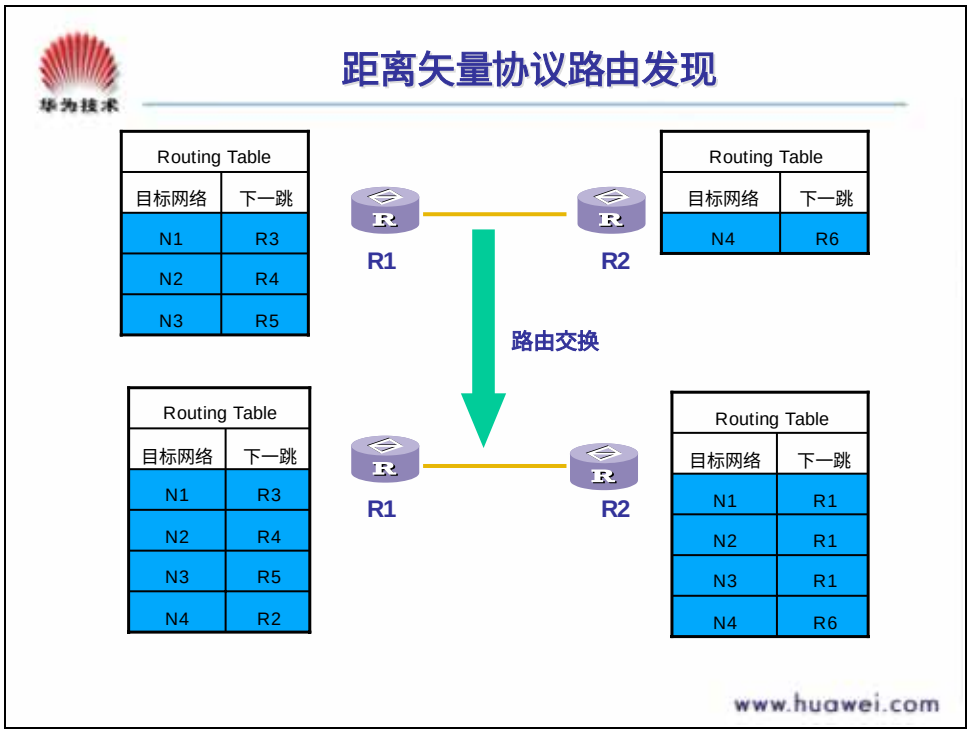
由于我们把非相邻两实体间的 d（i，j）定义为无穷大，当表达式中 k 不是 i 的相邻主机或路由器时，D（i，j）永远不可能为最小，故我们也可以把 k 限定为与 i 相邻。由此我们可以得出一个基于这个数学模型的计算 Metric 的简单算法：实体

i 接收它的邻居们 k 发送给它的到目标主机 j 的距离评价，并加上 d（i，j），在这里是通过 i，k 之间网络所需的 cost 值，接下来 i 比较来自所需邻居的信息，并选择其中最小的。可以证明，在拓扑结构不变的情况下该算法在有限时间内收敛于正确的 D（i，j）。

距离矢量算法通过上述方法累加网络距离，并维护网络拓扑信息数据库。使用这种算法，路由器并不能知道整个网络的确切拓扑结构。

某种程度上，距离矢量信息类似十字路口上指向目的地的路标，沿着路标的指向前进，在下一个十字路口，会再看到一个路标，但在这个路标处，距离目的地就近了一些。只要路径中每下一个路标都能表示到目的地距离的缩短，则这个路径为最优的。

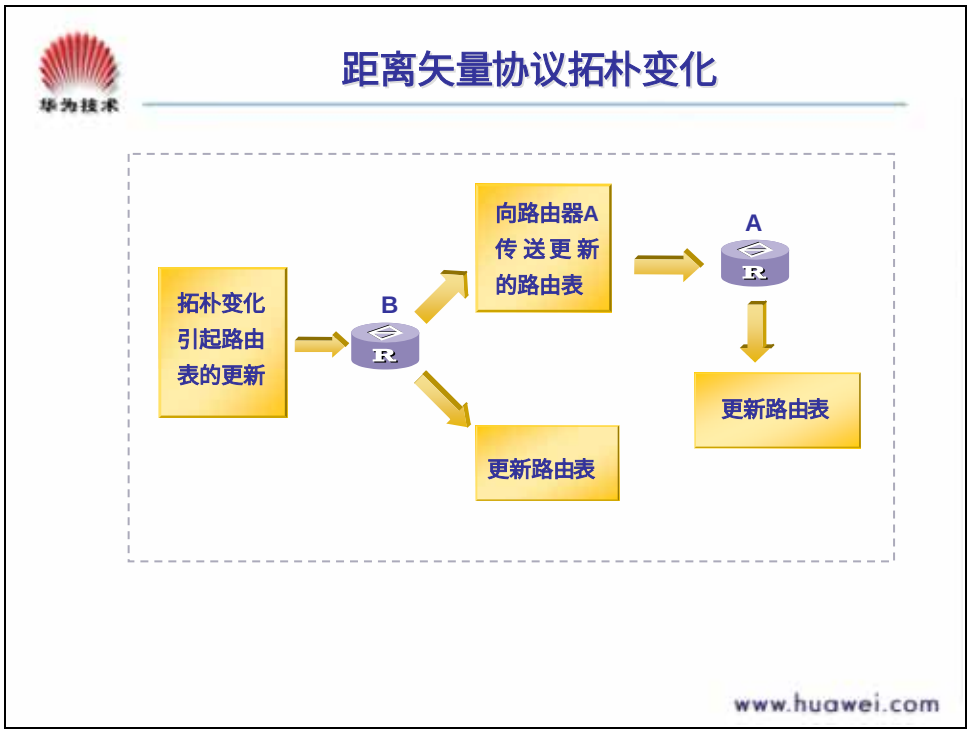
7.4.2 距离矢量协议路由发现



距离矢量协议直接传送各自的路由表信息。网络中的路由器从自己的邻居路由器得到路由信息，并将这些路由信息连同自己的本地路由信息发送给其他邻居，这样一级级的传递下去以达到全网同步。每个路由器都不了解整个网络拓扑，它们只知道与自己直接相连的网络情况，并根据从邻居得到的路由信息更新自己的路由表。

距离矢量协议无论是实现还是管理都比较简单，但是它的收敛速度慢，报文量大，占用较多网络开销，并且为避免路由环路需要做各种特殊处理。

7.4.3 距离矢量协议拓扑变化



距离矢量算法要求每个路由器将自己的路由表传送给相邻的路由器。当路由器接收到更新的路由信息时，首先将更新的信息与原有的路由表中的信息相比较，遇到下述情况之一时，须修改本地路由表（假设 RouterA 收到 RouterB 的 D-V 报文）以反映最新的网络变化：

⑴ RouterB 的路由表中列出的某表项 RouterA 的路由表中没有，则 RouterA 的路由表中须增加相应表项，其目标网络为 RouterB 路由表中的目标网络，其路径开销为 RouterB 表项中的路径开销加 1 （假设以跳数计算路径开销），其下一跳为 RouterB；

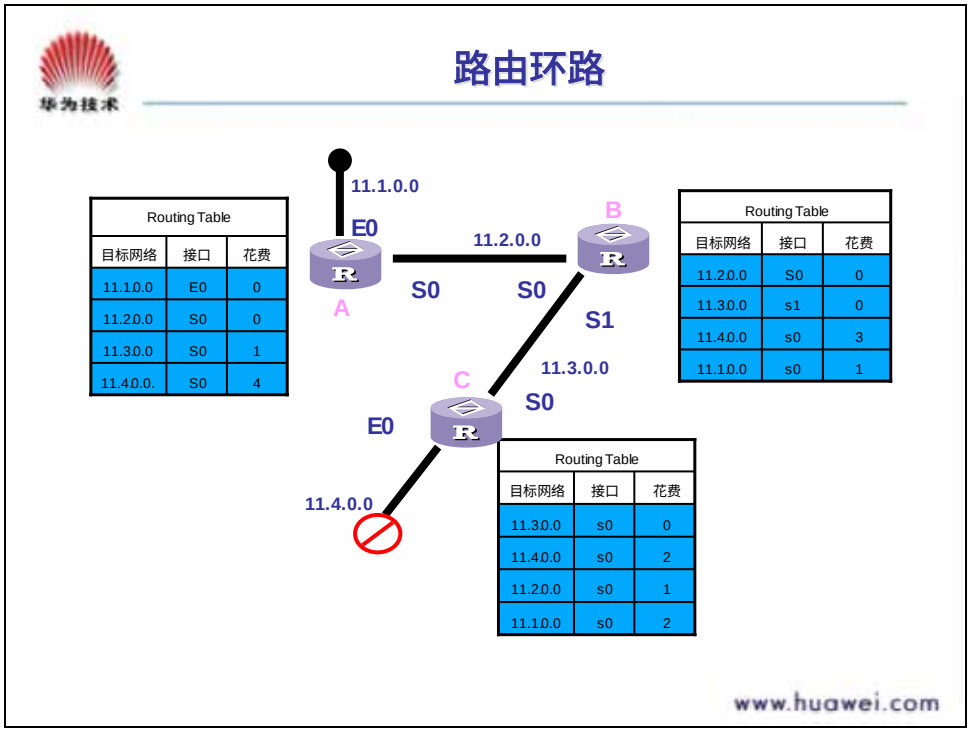
⑵ RouterB 的路由表中去往某目标网络的路径开销比 RouterA 的路由表中去往该目标网络的路径开销减 1 还小，这说明去往该目标网络若经过 RouterB 路径开销会更小，则 RouterA 修改本表项，将下一跳改为 RouterB，路径开销为 RouterB 中的路径开销加 1 ；

⑶ RouterA 的路由表中去往某目标网络的下一跳为 RouterB，而 RouterB 的路由表中去往该目标网络的路径开销发生了变化，则 RouterA 中相应表项的路径开销须修改，以 RouterB 的更新后的路径开销加 1 取代原来的路径开销；

⑷ RouterA 的路由表中去往某目标网络的下一跳为 RouterB，而 RouterB 的路由表中不再包含去往该目标网络的路径，则 RouterA 的路由表中相应路径应删除。

7.5 路由环路问题

7.5.1 路由环路产生



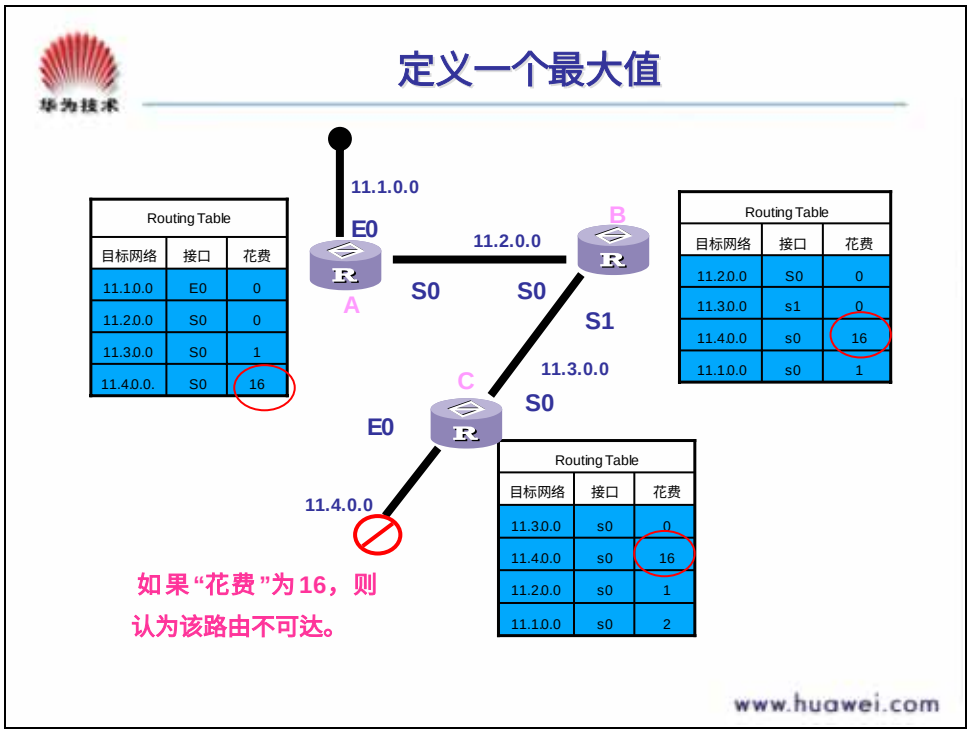
由于网络故障可能会引起路径与实际网络拓扑结构不一致而导致网络不能快速收敛，这时，可能会发生路由环路现象。图中用一个简单的网络结构来说明路由环路的产生。

如上图所示，如果网络 11.4.0.0 故障，就可能会在路由器之间产生路由环路，下面是产生路由环路的步骤：

- 在网络 11.4.0.0 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。在本例中，路径开销用跳数来计算，所以，每条链路的开销是 1。路由器 C 与网络 11.4.0.0 直连，跳数为 0。路由器 B 经过路由器 C 到达网络 11.4.0.0，跳数为 1。路由器 A 经过路由器 B 到达网络 11.4.0.0，跳数为 2。
- 当网络 11.4.0.0 发生故障，路由器 C 最先收到故障信息，路由器 C 把网络 11.4.0.0 设为不可达，并等待更新周期到来通告这一路由变化给相邻路由器。如果，路由器 B 的路由更新周期在路由器 C 之前到来，那么路由器 C 就会从路由器 B 那里学习到去往 11.4.0.0 的新路由（实际上，这一路由已经是错误路由了）。这样路由器 C 的路由表中就记录了一条错误路由（经过路由器 B，可去往网络 11.4.0.0，跳数增加到 2 ）。
- 路由器 C 学习了一条错误信息后，它会把这样的路由信息再次通告给路由器 B，根据通告原则，路由器 B 也会更新这样一条错误路由信息，认为可以通过路由器 A 去往网络 11.4.0.0，跳数增加到 3 。

- 这样，路由器 B 认为 可以通过路由器 C 去往网络 11.4.0.0，路由器 C 认为 可以通过路由器 B 去往网络 11.4.0.0，就形成了环路。

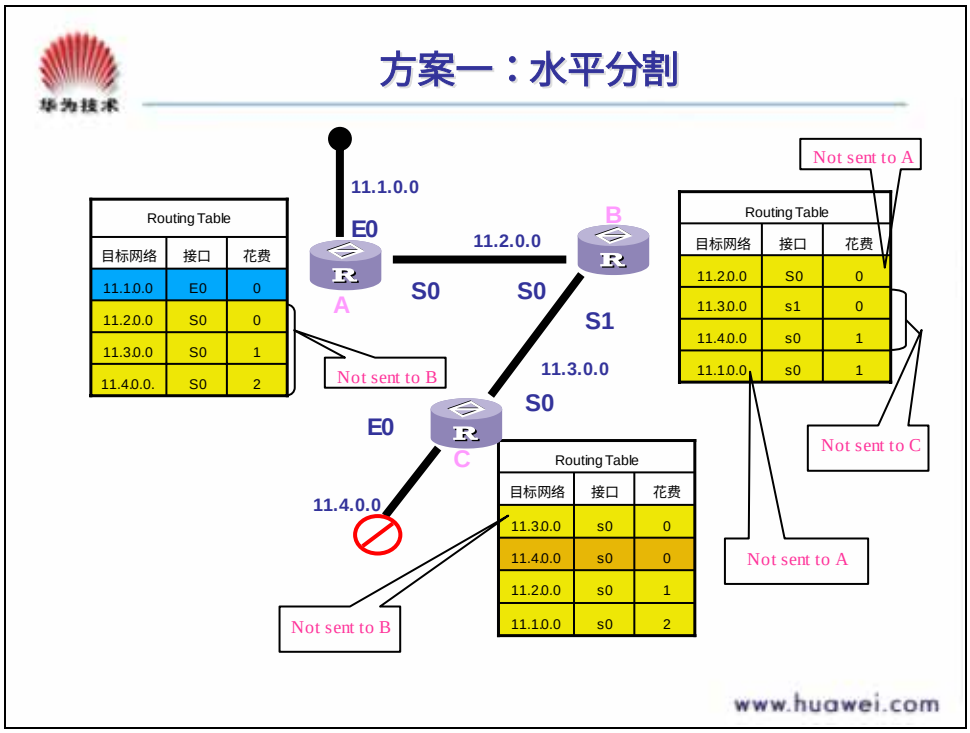
7.5.2 环路补救方案



如上所述，发生路由环路时，路由器去往网络 11.4.0.0 的跳数会不断的增大，网络无法收敛。为解决这个问题，我们给跳数定义一个最大值，在 RIP 路由协议中，允许跳数最大值为 16。在图中，当跳数到达最大值时，网络 11.4.0.0 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 11.4.0.0 的路由。

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权值无限增大的问题，同时也校正了错误的路由信息。但是，在最大权值到达之前，路由环路还是会存在。也就是说，以上解决方案只是补救措施，不能避免环路产生，只能减轻路由环路产生的危害。路由协议的设计者们又提供了诸如水平分割、触发更新等多种避免环路产生几率的方案。

7.5.3 环路避免方案



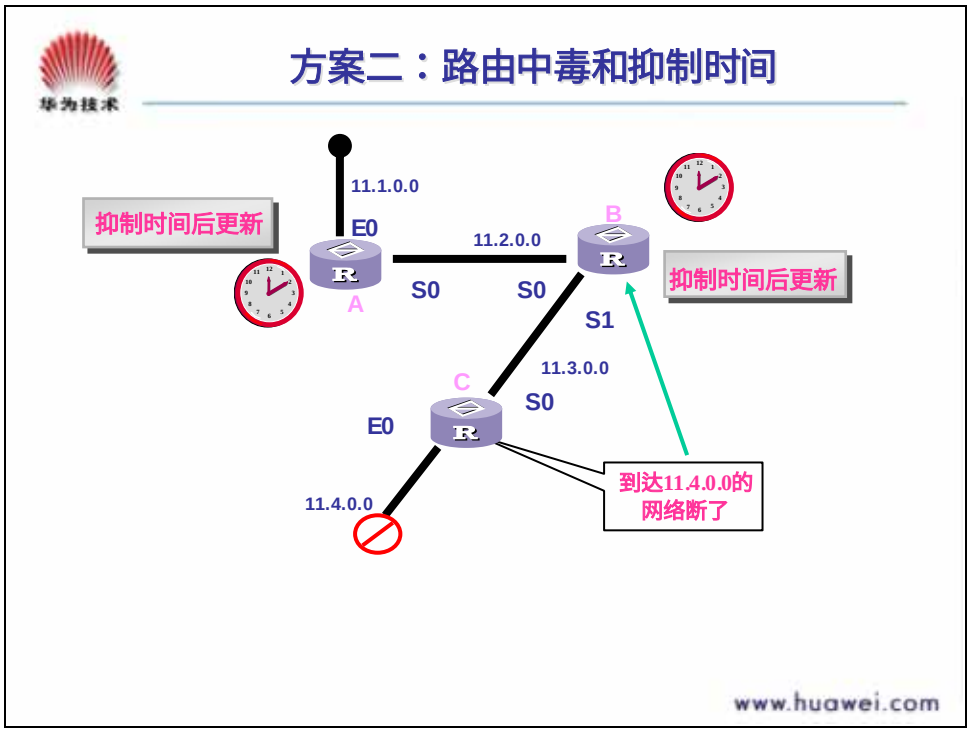
水平分割是在距离矢量路由协议中最常用的避免环路发生的解决方案之一。分析产生路由环路的原因，其中一条就是因为路由器将从某个邻居学到的路由信息又告诉了这个邻居。水平分割的思想就是在路由信息传送过程中，不再把路由信息发送给接收此路由信息的接口上。如上图所示：

- 路由器 C 告诉路由器 B 去往网络 11.4.0.0 的路由，路由器 B 会把此路由信息传递给路由器 A。同时，也会再传回给路由器 C。网络 11.4.0.0 没有崩溃时，路由器 C 不会接受路由器 B 传递来的去往网络 11.4.0.0 的路由信息。因为，路由器 C 有花费更小的路由。

- 如果路由器 C 到达网络 11.4.0.0 的路由崩溃了，路由器 C 就会接受路由器 B 传递来的去往网络 11.4.0.0 的路由信息，尽管这条路由信息已经是错误路由了（因为随着路由器 C 去往网络 11.4.0.0 的路由崩溃，路由器 B 从路由器 C 学到的去往网络 11.4.0.0 路由也就错误了）。但是路由器 C 并不知道这一点。

这样，路由器 B 认为 可以通过路由器 C 去往网络 11.4.0.0 ，路由器 C 认为 可以通过路由器 B 去往网络 11.4.0.0 ，就形成了环路。

- 水平分割方法就是解决这样问题的，水平分割不允许路由器将路由更新信息再次传回到传出该路由信息的端口。上图中，路由器 B 从路由器 C 那里学习到了去往网络 11.4.0.0 的路由。水平分割规定：路由器 B 不再把去往网络 11.4.0.0 的路由信息传回给路由器 C，从而在一定程度上避免了环路的产生。



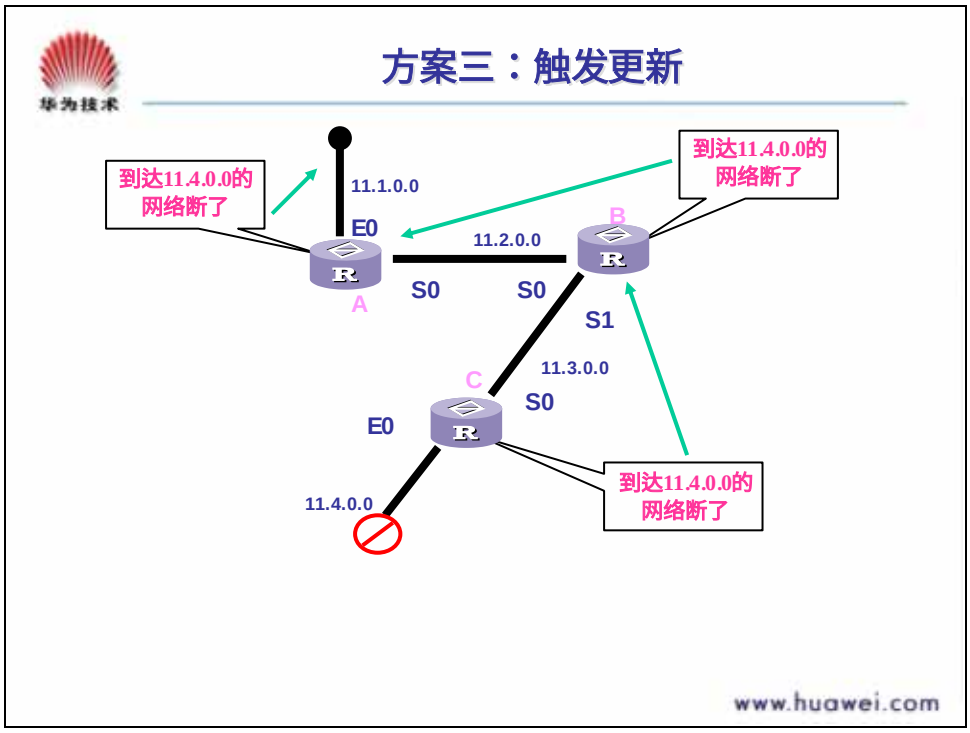
路由中毒和抑制时间结合起来，也可以在一定程度上避免路由环路产生，同时也可以抑制因复位接口等原因，引起的网络动荡。这种方法在网络故障或接口复位时，使相应路由中毒，同时启动抑制时间，控制 路由器在抑制时间内不要轻易更新自己的路由表。从而，避免环路产生、抑制网络动荡。

如上图所示：

- 当网络 11.4.0.0 发生故障时，路由器 C 使自己路由表中的此路由项中毒，也就是在路由表中使到达网络 11.4.0.0 的路径开销是无穷大（也就是不可达），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（或同一方向）又接收到此路由可达的更新信息时，路由器就将网络标识为可达，并删除抑制时间。

- 如果接收到其他的相邻路由器的更新信息，且新的权值比以前的权值好，则路由器就将更新路由表，接受这一更优的路由，并删除抑制时间。

- 在抑制时间结束之前的任何时刻，如果从其他的相邻路由器接收到路径可用的更新信息时，但新的权值没有以前的权值好，则不接收此更新路由。如果在抑制时间过后，路由器仍能收到该更新路由信息，则路由器将更新路由表。



如图，网络 11.4.0.0 不可达了，路由器 C 最先得到这一信息。通常，更新路由信息会定时发送给相邻路由器。例如，RIP 协议每隔 30 秒发送一次。但如果在路由器 C 等待更新周期到来的时候，路由器 B 的更新报文传到了路由器 C，路由器 C 就会学到路由器 B 的去往网络 11.4.0.0 的错误路由。这样就会形成路由环路。如果路由器 C 发现网络故障之后，不再等待更新周期到来，就立即发送路由更新信息，则可以避免产生上述问题。这就是触发更新机制。

触发更新机制是在路由信息产生某些改变时，立即发送给相邻路由器一种称为触发更新的信息。路由器检测到网络拓扑变化，立即依次发送触发更新信息给相邻路由器，如果每个路由器都这样做，这个更新会很快传播到整个网络。

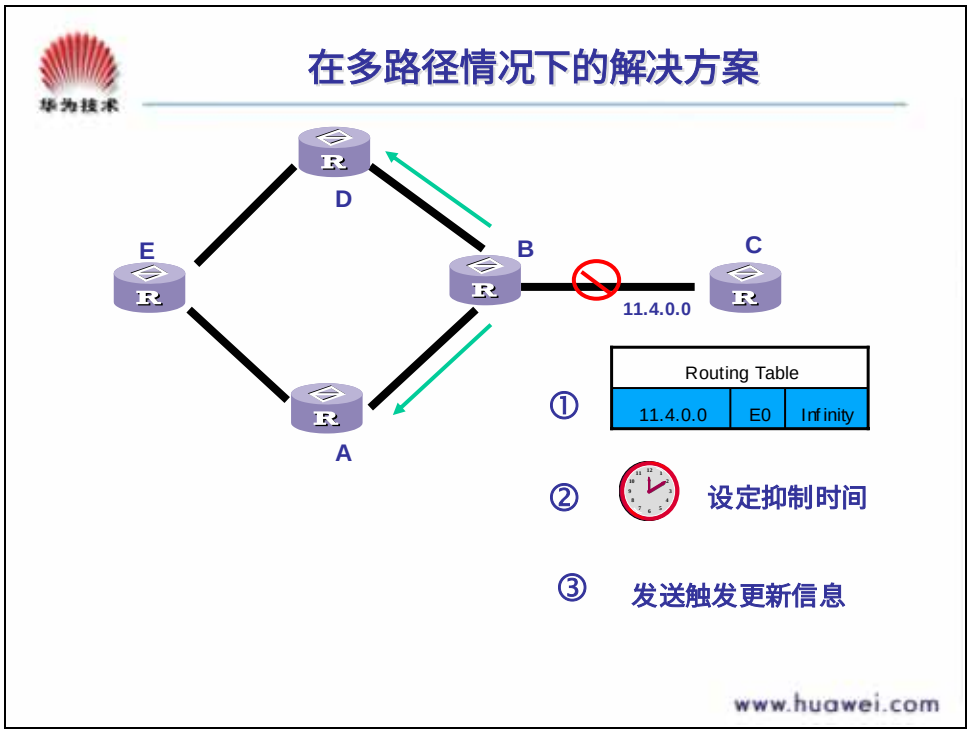
在图中，路由器 C 立即通告网络 11.4.0.0 不可达信息，路由器 B 接收到这个信息，就从 S0 口发出网络 11.4.0.0 不可达信息，依次路由器 A 从 E0 口通告此信息。

从上述叙述可以看出，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：

- 包含有更新信息的数据包可能会被丢掉或损坏。
- 如果触发更新信息还没有来得急发送，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。

为解决以上的问题，我们将抑制时间和触发更新相结合，就可以解决上述问题。抑制时间方法有一个规则就是，当到某一目的网络的路径出现故障，在一定时间内，路由器不轻易接收到这一目的网络的路径信息。因此，将抑制时间和触发更新相结合就可以确保了触发信息有足够的时间在网络中传播。

7.5.4 在多路径情况下的解决方案



在下面的例子中，路由器之间有多条路径到达对方，图中，路由器 A，D，E 都有两条路径到达网络 11.4.0.0 。

当网络 11.4.0.0 发生故障时，会有下面的情形发生；

⑴ 路由中毒——当路由器 B 检测到网络 11.4.0.0 故障时，路由器 B 使所有连接该网络的路径中毒，使到此网络的跳数为最大数值。

⑵ 设定抑制时间——旦路由器 B 使连接网络 11.4.0.0 的路径中毒，则它会设定一个抑制时间。

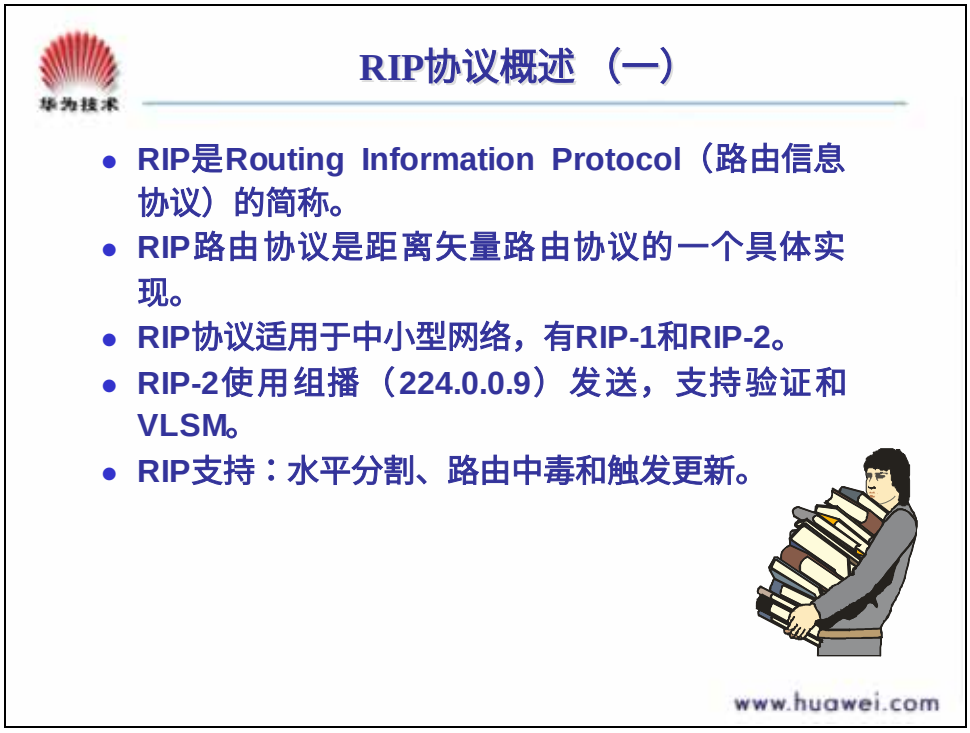
⑶ 发送触发更新信息——路由器 B 向路由器 A、D 发送触发更新信息，指出网络 11.4.0.0 故障。新的路由信息在其它网络间传输，使得其余路由器再重复步骤 2、3。路由器 A、D 接收到触发更新信息以后，在抑制时间内禁止更新的路径信息。接下来，路由器 A 和 D 再向路由器 E 发送网络 11.4.0.0 故障的触发更新信息。

路由器 E 接收到触发更新信息后，设定自己的抑制时间，一直处于等待状态，直到出现下面的情形：

- 抑制时间结束。出现这种情况，路由器 E 确定网络 11.4.0.0 不可达。
- 接收到网络状态改变的信息。出现这种情况，路由器 E 更新路由表。
- 接收到具有更好权值的路径更新信息。出现这种情况，路由器 E 更新路由表。

7.6 RIP路由协议

7.6.1 RIP 协议概述



RIP 是 Routing Information Protocol （路由信息协议）的简称。它是一种相对简单的动态路由协议，但在实际使用中有着广泛的应用。RIP 是一种基于 D-V 算法的路由协议，它通过 UDP 交换路由信息，每隔 30 秒向外发送一次更新报文。如果路由器经过 180 秒没有收到来自对端的路由更新报文，则将所有来自此路由器的路由信息标志为不可达，若在其后 120 秒内仍未收到更新报文，就将该条路由从路由表中删除。

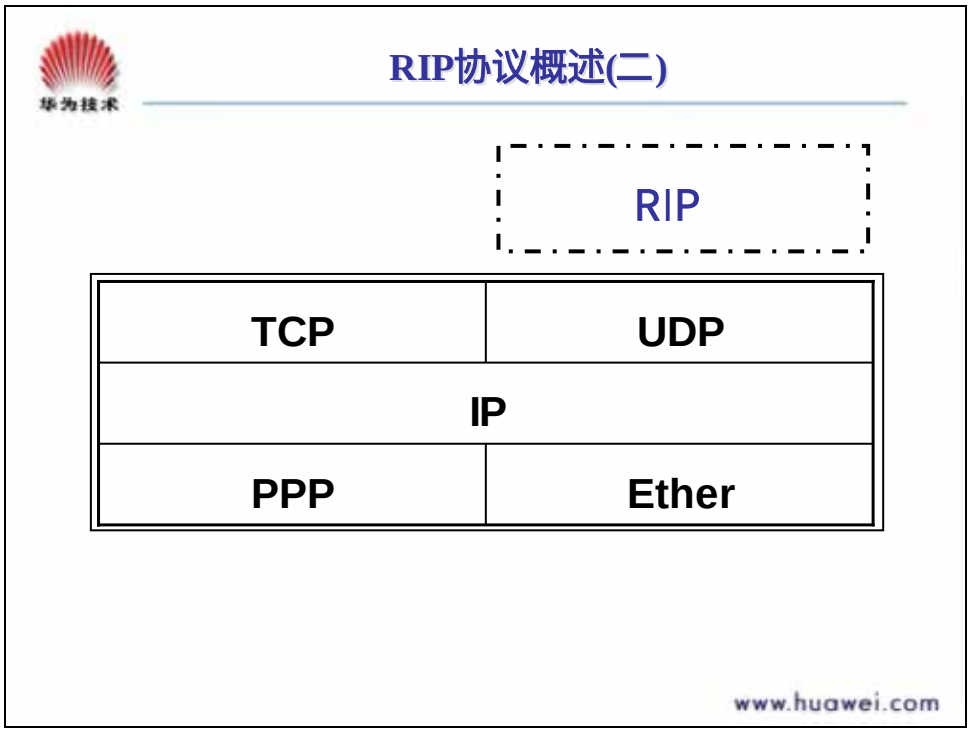
RIP 使用跳数（Hop Count）来衡量到达目的网络的距离，称为路由权（Routing Metric）。在 RIP 中，路由器到与它直接相连网络的跳数为 0，通过一个路由器可达的网络的跳数为 1，其余依此类推。为限制收敛时间，RIP 规定 metric 取值 0~15 之间的整数，大于或等于 16 的跳数被定义为无穷大，即目的网络或主机不可达。

为提高性能，防止产生路由环路，RIP 支持水平分割（Split Horizon）与路由中毒（Poison Reverse），并在路由中毒时采用触发更新（Triggered Update）。另外，RIP 协议还允许引入其它路由协议所得到的路由。

RIP 包括 RIP-1 和 RIP-2 两个版本，RIP-1 不支持变长子网掩码(VLSM)，RIP-2 支持变长子网掩码(VLSM)，同时 RIP-2 支持明文认证和 MD5 密文认证。

RIP-1 使用广播发送报文，RIP-2 有两种传送方式：广播方式和组播方式，缺省将采用组播发送报文，RIP-2 的组播地址为 224.0.0.9。组播发送报文的好处是在同一网络中那些没有运行 RIP 的网段可以避免接收 RIP 的广播报文；另外，组播

发送报文还可以使运行 RIP-1 的网段避免错误地接收和处理 RIP-2 中带有子网掩码的路由。

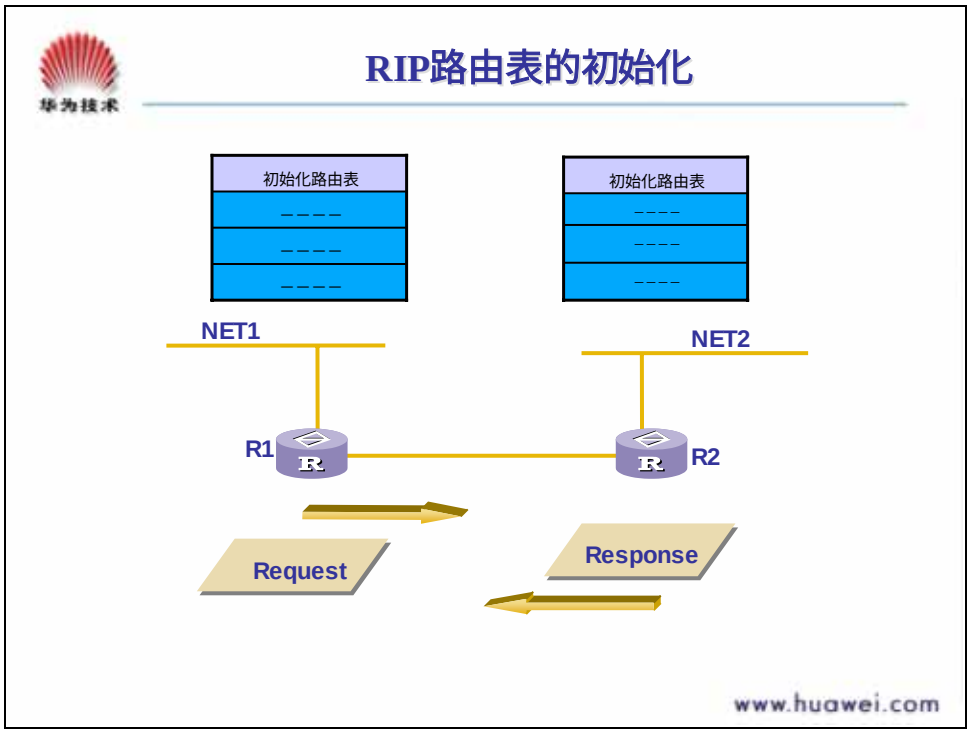


RIP 协议是最早使用的 IGP 之一，RIP 协议被设计用于使用同种技术的中小型网络，因此适应于大多数的校园网和使用速率变化不是很大的区域性网络。对于更复杂的环境，一般不使用 RIP 协议。

在实现时，RIP 作为一个系统长驻进程存在于路由器中，它负责从网络中的其它路由器接收路由信息，从而对本地 IP 层路由表作动态的维护，保证 IP 层发送报文时选择正确的路由，同时广播本路由器的路由信息，通知相邻路由器作相应的修改。

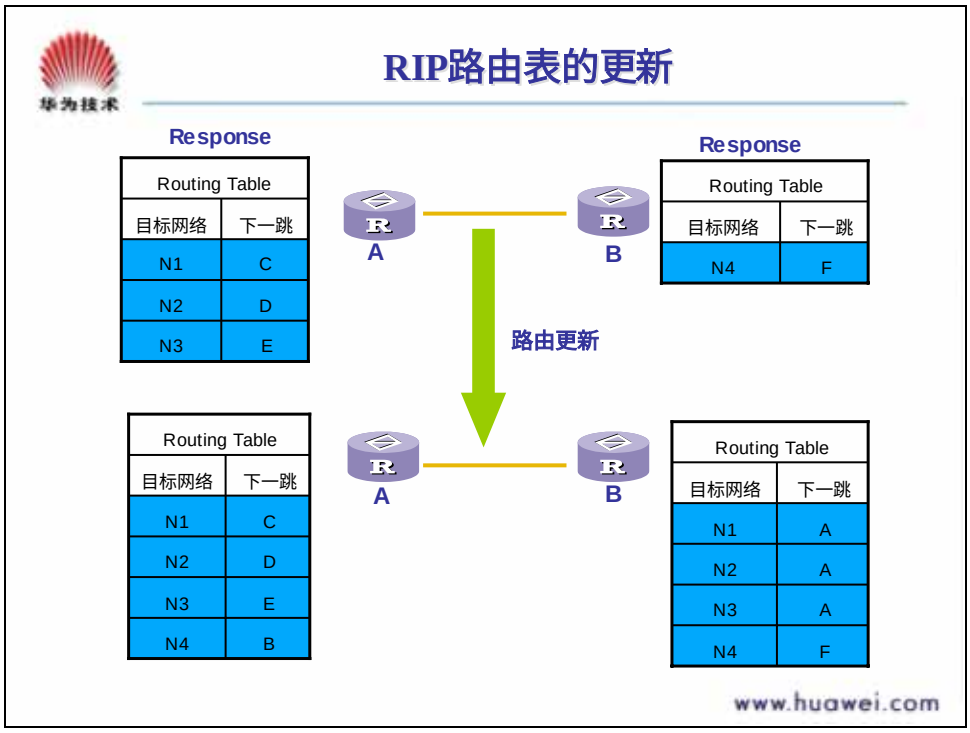
RIP 协议处于 UDP 协议的上层，RIP 所接收的路由信息都封装在 UDP 的数据报中，RIP 在 520 号端口上接收来自远程路由器的路由修改信息，并对本地的路由表做相应的修改，同时通知其它路由器。通过这种方式，达到全局路由的同步。

7.6.2 RIP 协议的实现



- RIP 启动时的初始路由表仅包含本路由器的一些直连接口路由。
- RIP 协议启动后向各接口广播一个 Request 报 文。
- 邻居路由器的 RIP 协议从某接口收到 Request 报文后，根据自己的路由表，形成 Response 报文向该接口对应的网络广播。
- RIP 接收邻居路由器回复的包含邻居路由器路由表的 Response 报文，形成自己的路由表。

RIP 根据 D-V 算法的特点，将协议的参加者分为主动机和被动机两种。主动机主动向外广播路由刷新报文，被动机被动地接收路由刷新报文。一般情况下，主机作为被动机，路由器则既是主动机又是被动机，即在向外广播路由刷新报文的同时，接收来自其它主动机的 D-V 报文，并进行路由刷新。



RIP 协议以 30 秒为周期用 Response 报文广播自己的路由表。

收到邻居发送而来的 Response 报文后，RIP 协议计算报文中的路由项的度量值，比较其与本地路由表路由项度量值的差别，更新自己的路由表。

报文中路由项度量值的计算： metric' = MIN（metric + cost, 16），metric 为报文中携带的度量值信息，cost 为接收报文的网络的度量值开销，缺省为 1（1 跳），16 代表不可达。

RIP 路由表的更新原则：

对本路由表中已有的路由项，当发送报文的网关相同时，不论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）；

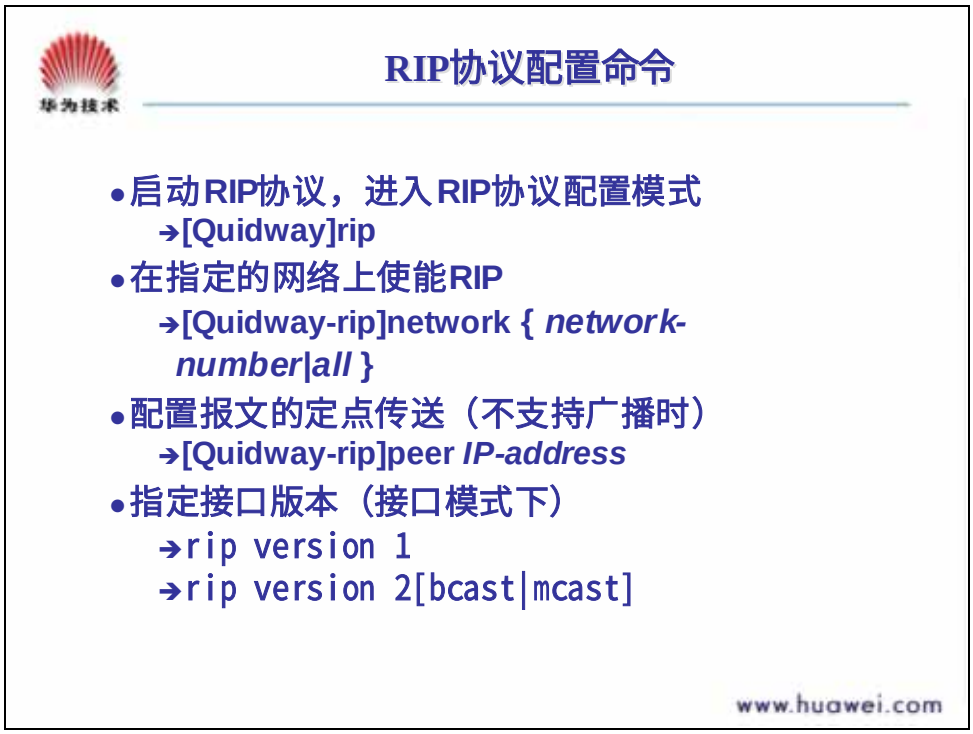
对本路由表中已有的路由项，当发送报文的网关不同时，只在度量值减少时，更新该路由项；

对本路由表中不存在的路由项，在度量值小于不可达（16）时，在路由表中增加该路由项；

路由表中的每一路由项都对应一老化定时器，当路由项在 180 秒内没有任何更新时，定时器超时，该路由项的度量值变为不可达（16）。

某路由项的度量值变为不可达后，以该度量值在 Response 报文中发布四次（120 秒），之后从路由表中清除。

7.6.3 RIP 协议配置命令



在各项配置任务中，必须先启动 RIP、使能 RIP 网络后，才能配置其它的功能特性。而配置与接口相关的功能特性不受 RIP 是否使能的限制。需要注意的是，在关闭 RIP 后，原来的接口参数也同时失效。

在全局配置模式下用 rip 命令启动 RIP 协议并进入 RIP 协议配置模式。

- RIP 任务启动后还必须指定其工作网段，RIP 只在指定网段上的接口工作；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，也不将它的接口路由转发出去，就好象这个接口不存在一样。

network-number 为使能或不使能的网络的地址，可为各个接口 IP 网络的地址。当对某一地址使用命令 Network 时，效果是使能该地址的网段的接口。例如：network 129.102.1.1，用 display current-configuration 和 display rip 命令看到的均是 network 129.102.0.0。

- RIP 是一个广播发送报文的协议，为与非广播网络交换路由信息，就必须采用定点传送的方式。 通常的情况下，我们不建议用户使用该命令，因为对端并不需要一次收到两份相同的报文。需要要注意的是：**neighbor** 在发送报文时也要受 **rip work** 、**rip output** 、**rip input** 和 **network** 等的限制。

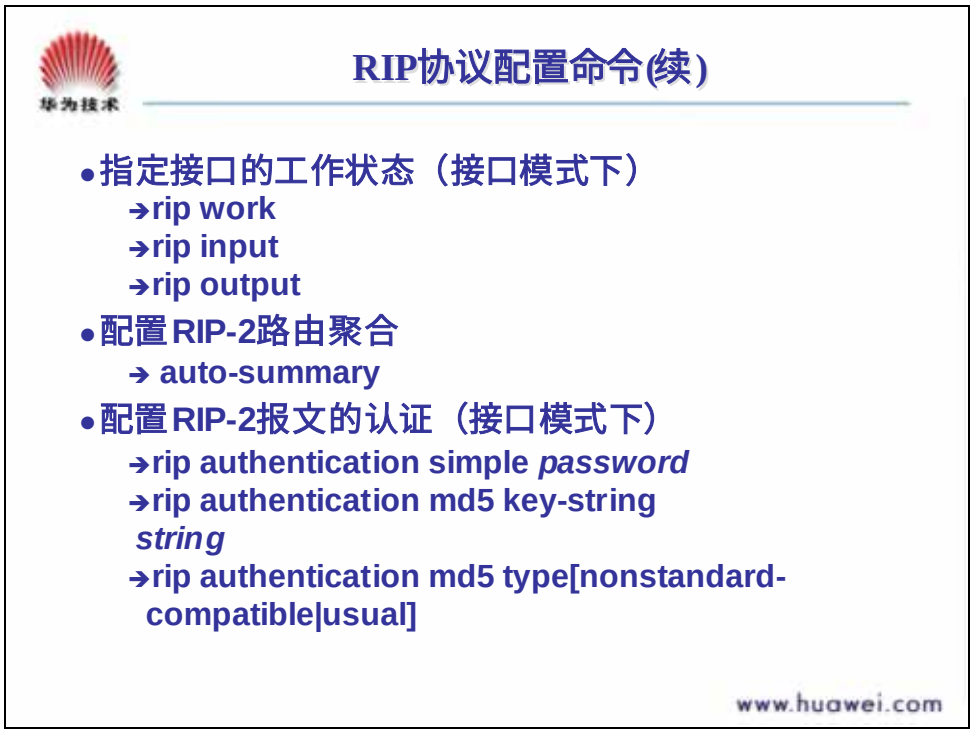
- 可指定接口所处理 RIP 报文的版本。

需要注意的是：RIP-1 采用广播形式发送报文； RIP-2 有两种传送方式，广播方式和多播方式，缺省将采用多播发送报文。 RIP-2 中多播地址为 224.0.0.9。多播发送报文的好处是在同一网络中那些未运行 RIP 的主机可以避免接收 RIP 的广

播报文。另外，多播发送报文还可以使运行 RIP-1 的主机避免错误地接收和处理 RIP-2 中带有子网掩码的路由。

当接口运行 RIP-1 时，只接收与发送 RIP-1 与 RIP-2 广播报文，不接收 RIP-2 多播报文。当接口运行在 RIP-2 广播方式时，只接收与发送 RIP-1 与 RIP-2 广播报文，不接收 RIP-2 多播报文； 当接口运行在 RIP-2 多播方式时，只接收和发送 RIP-2 多播报文；不接收 RIP-1 与 RIP-2 广播报文。

缺省情况下，接口运行 RIP-1 报文，即只能接收与发送 RIP-1 报文。



- 可指定 RIP 在接口上的工作状态，如接口上是否运行 RIP，即是否在接口发送和接收 RIP 刷新报文；还可单独指定接口是否发送或者接收更新报文。

在缺省情况下，一个接口既可接收 RIP 更新报文，也可发送 RIP 更新报文。

undo rip work 命令的功能与 no network 命令功能相近，但它们并不完全相同。相同点在于，使用任一命令的接口都不再收发 RIP 路由；区别在于： 在 undo rip work 情况下，其它接口对使用该命令的接口的路由仍然转发，而在 undo network 的情况下，其它接口对使用该命令的接口的路由不再转发，见到的效果就象少了一个接口。

另外， rip work 从功能上等价于 rip input 与 rip output 两个命令。

- 路由聚合是指：同一自然网段内的不同子网的路由在向外（其它网段）发送时聚合成一条自然掩码的路由发送。路由聚合减少了路由表中的 路由信息量，也减少了路由交换的信息量。

RIP-1 只发送自然掩码的路由，即总是以路由聚合形式向外发送路由，关闭路由聚合对 RIP-1 将不起作用。RIP-2 支持无类别路由，当需要将子网的路由广播出去时，可关闭 RIP-2 的路由聚合功能。

缺省情况下，允许 RIP-2 进行路由聚合。

- RIP-1 不支持报文认证，但当接口运行 RIP-2 时， 可进行报文的认证。

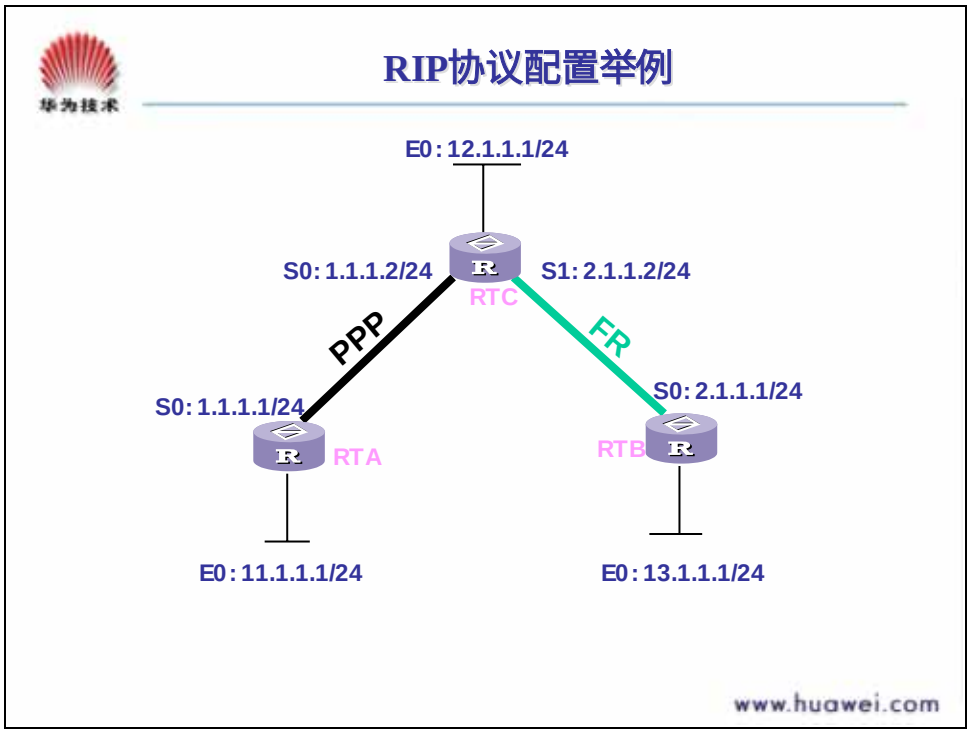
RIP-2 支持两种认证方式：明文认证 Simple 和 MD5 密文认证。MD5 密文认证的报文格式有两种：一种遵循 RFC1723（RIP Version 2 Carrying Additional Information）规定；另一种遵循 RFC2083（RIP-2 MD5 Authentication）规定。

Cisco-compatible 路由器只支持后一种格式，Quidway 系列路由器对两种格式的 MD5 认证报文都提供支持。

明文认证不能提供安全保障，未经加密的认证字将随报文一同传送，所以明文认证不能用于安全性要求较高的情况。

缺省的情况下，接口采用 MD5 认证，若未指定 MD5 认证报文格式的类型，将采用后一种报文格式类型（usual）。

7.6.4 RIP 协议配置举例



如胶片所示，RTA 和 RTB 之间链路层封装 PPP 协议，RTB 和 RTC 之间链路层封装 FR 协议，所有路由器启动 RIP 路由协议。RTA 和 RTB 之间做 MD5 验证。主要配置如下：

RTA:

[RTA] **rip**

[RTA-rip] **network all**

[RTA-Ethernet0]rip version 2 broadcast

[RTA-Serial0]rip version 2 broadcast

[RTA-Serial0]rip authentication-mode md5 key-string quidway // MD5

RTB:

[RTB] **rip**

[RTB-rip] **network all**

[RTB-rip] **peer 2.1.1.1** // 配置报文指定发送

[RTB-Ethernet0rip version 2 broadcast

[RTB-Serial0]rip version 2 broadcast

[RTB-Serial0]rip authentication-mode md5 key-string quidway // MD5

[RTB-Serial1]rip version 2 broadcast

[RTB-Serial1]link-protocol fr

RTC:

[RTC] **fr switching** // 使能帧中继交换

[RTC] **rip**

[RTC-rip] **network all**

[RTC-rip] **peer 2.1.1.1**

[RTB-Ethernet0]rip version 2 broadcast

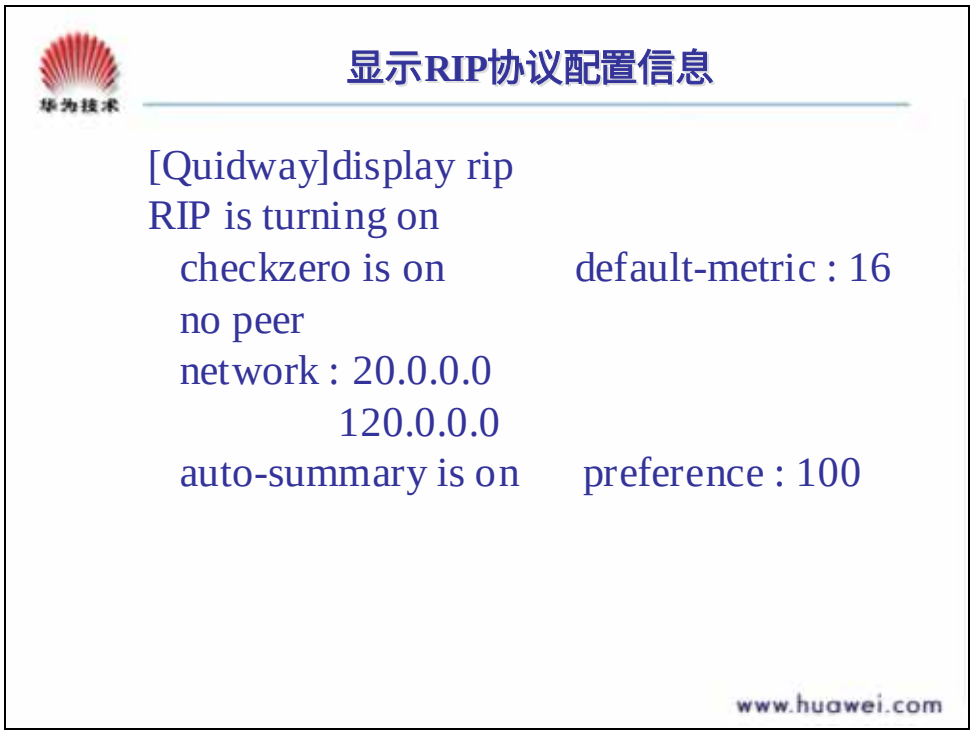
[RTC-Serial1]rip version 2 broadcast

[RTC-Serial1]link-protocol fr

[RTC-Serial1] **fr interface-type DCE** // 封装帧中继接口类型

[RTC-Serial1] **fr dlci 20** // 分配 DLCI

7.6.5 显示 RIP 协议配置信息



用 display rip 显示当前 RIP 协议的运行状态：

RIP is turning on

checkzero is on default-metric : 16 // 校验和开关打开，缺省路由权为 16；

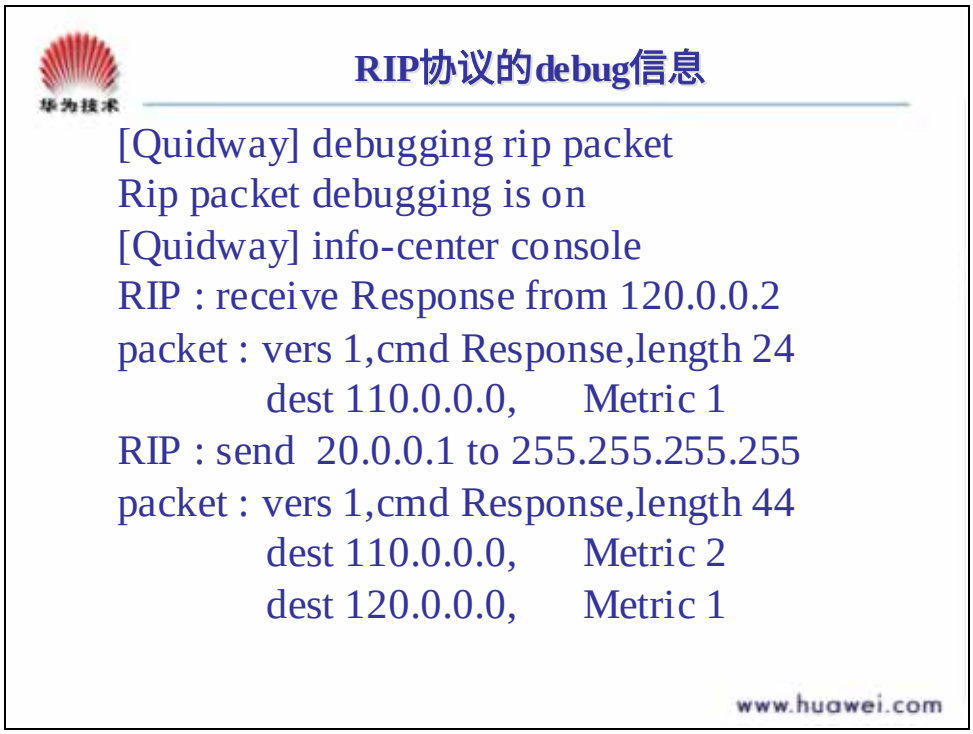
no peer // 没有指定定点传送地址；

network : 20.0.0.0

120.0.0.0 // 在 20.0.0.0 与 120.0.0.0 网段上使用 RIP 协议；

summary is on preference : 100 //自动聚合路由，RIP 路由的 preference 为 100；

7.6.6 RIP 协议的 debug 信息



使用 debugging rip packet 打开 RIP 协议的调试开关，同时用 info-center console 输出调试信息：

RIP : receive Response from 120.0.0.2 （serial0）

packet : vers 1,cmd Response,length 24

dest 110.0.0.0, Metric 1

路由器从 120.0.0.2 收到一个 Response 报文，包含信息：版本号、报文长度，报文的主体是一条路由信息：目标地址（dest）是 110.0.0.0， 度量值（ Metric）是 1 。

RIP : send 20.0.0.1 to 255.255.255.255

packet : vers 1,cmd Response,length 44

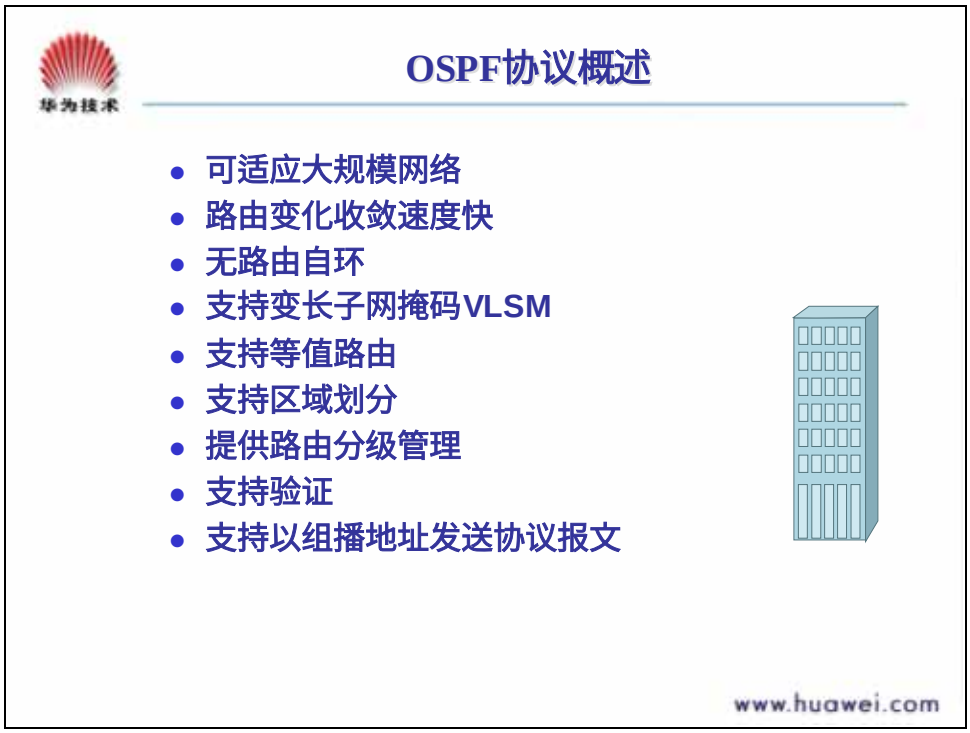
dest 110.0.0.0, Metric 2

dest 120.0.0.0, Metric 1

路由器发送一个 Response 报文，包含信息：版本号、报文长度，报文的主体是两条路由信息：dest 110.0.0.0, Metric 2 和 dest 120.0.0.0, Metric 1。

7.7 OSPF简介

7.7.1 OSPF 协议概述

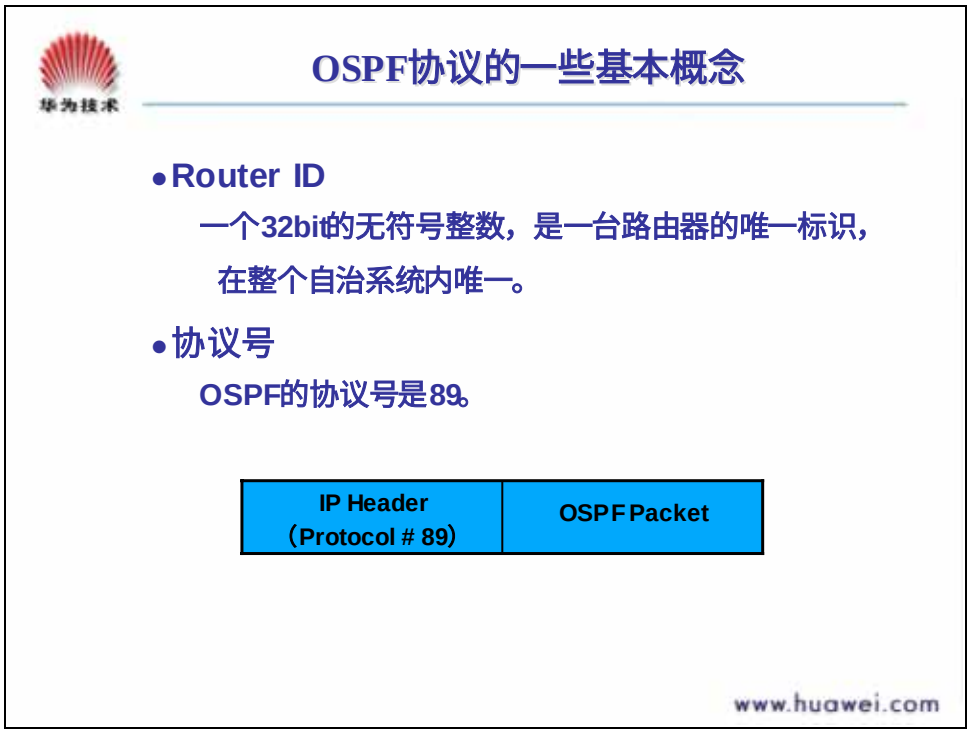


OSPF 是 Open Shortest Path First（即“开放最短路由优先协议”）的缩写。它是 IETF （Internet Engineering Task Force）组织开发的一个基于链路状态的自治系统内部路由协议。在 IP 网络上，它通过收集和传递自治系统的链路状态来动态地发现并传播路由。当前 OSPF 协议使用的是第二版，最新的 RFC 是 2328。OSPF 协议具有如下特点：

- 适应范围 —— OSPF 支持各种规模的网络，最多可支持几百台路由器。
- 快速收敛 —— 如果网络的拓扑结构发生变化，OSPF 立即发送更新报文，使这一变化在自治系统中同步。
- 无自环 —— 由于 OSPF 通过收集到的链路状态用最短路径树算法计算路由，故从算法本身保证了不会生成自环路由。
- 子网掩码 —— 由于 OSPF 在描述路由时携带网段的掩码信息，所以 OSPF 协议不受自然掩码的限制，对 VLSM 提供很好的支持。
- 区域划分 —— OSPF 协议允许自治系统的网络被划分成区域来管理，区域间传送的路由信息被进一步抽象，从而减少了占用网络的带宽。
- 等值路由 —— OSPF 支持到同一目的地址的多条等值路由。
- 路由分级 —— OSPF 使用 4 类不同的路由，按优先顺序来说分别是：区域内路由、区域间路由、第一类外部路由、第二类外部路由。

- 支持验证 —— 它支持基于接口的报文验证以保证路由计算的安全性。
- 组播发送 —— OSPF 在有组播发送能力的链路层上以组播地址发送协议报文，即达到了广播的作用，又最大程度的减少了对其他网络设备的干扰。

7.7.2 一些基本概念



Router ID：

一台路由器如果要运行 OSPF 协议，必须存在 Router ID，可以使用命令：

[Quidway]router id 1.2.3.4

手工配置。如果没有配置，则系统会优先选择 Lookback 接口的 IP 地址作为 ID（因为 Lookback 接口总是处于 UP 状态），如果没有配置 Lookback 接口，OSPF 会从当前接口的 IP 地址中自动选一个 IP 地址作为 ID。

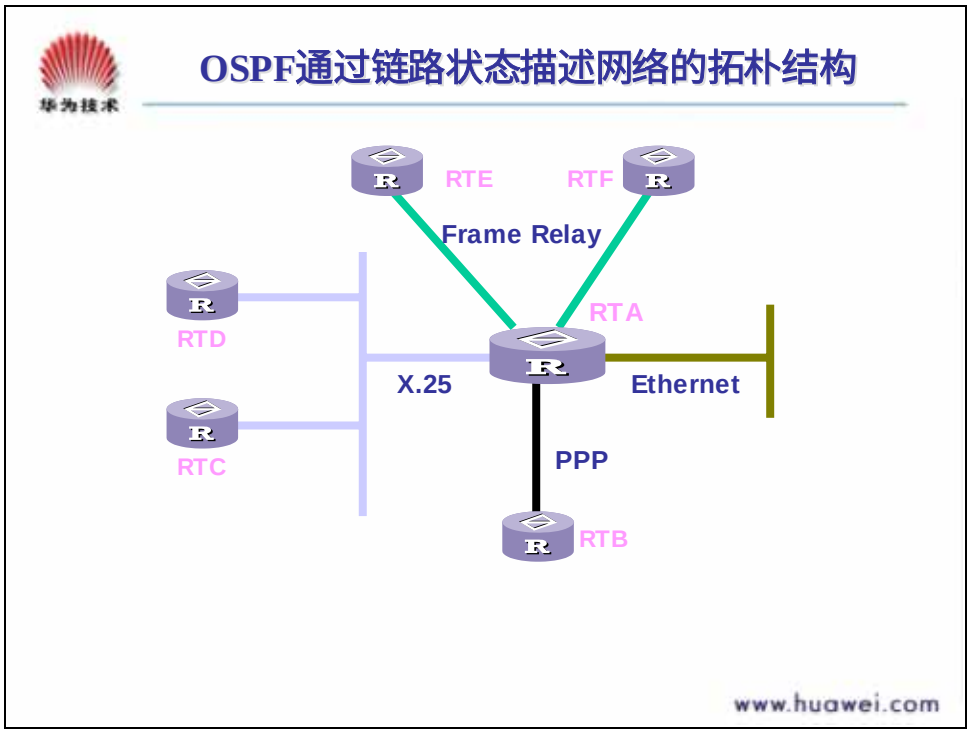
如果一台路由器的 Router ID 在运行中改变，必须重启 OSPF 协议或重启路由器才能使新的 Router ID 生效。

如果想查看本路由器的 Router ID，执行命令 display ospf。

协议号：

OSPF 协议用 IP 报文直接封装协议报文，协议号是 89。

7.7.3 链路状态



OSPF 协议计算路由是以本路由器周边网络的拓扑结构为基础的。每台路由器将自己周边的网络拓扑描述出来，传递给其他所有的路由器。

OSPF 将不同的网络拓扑抽象为以下四种类型：

- 该接口所连的网段中只有本路由器自己。（stub networks）
- 该接口通过点到点的网络与一台路由器相连。（point-to-point ）
- 该接口通过广播或 NBMA 的网络与多台路由器相连。（broadcast or NBMA networks）
- 该接口通过点到多点的网络与多台路由器相连。（point-to-multipoint）

NBMA 与点到多点的区别：

在 OSPF 协议中 NBMA 和点到多点都是指非广播多点可达的网络，但 NBMA 网络必须满足全连通（full meshed）的要求，即任意两点都可以不经转发而使报文直达对端。否则，我们称该网络是点到多点网络。

如上图所示，RTA 周围的链路状态情况可归纳为以下四种：

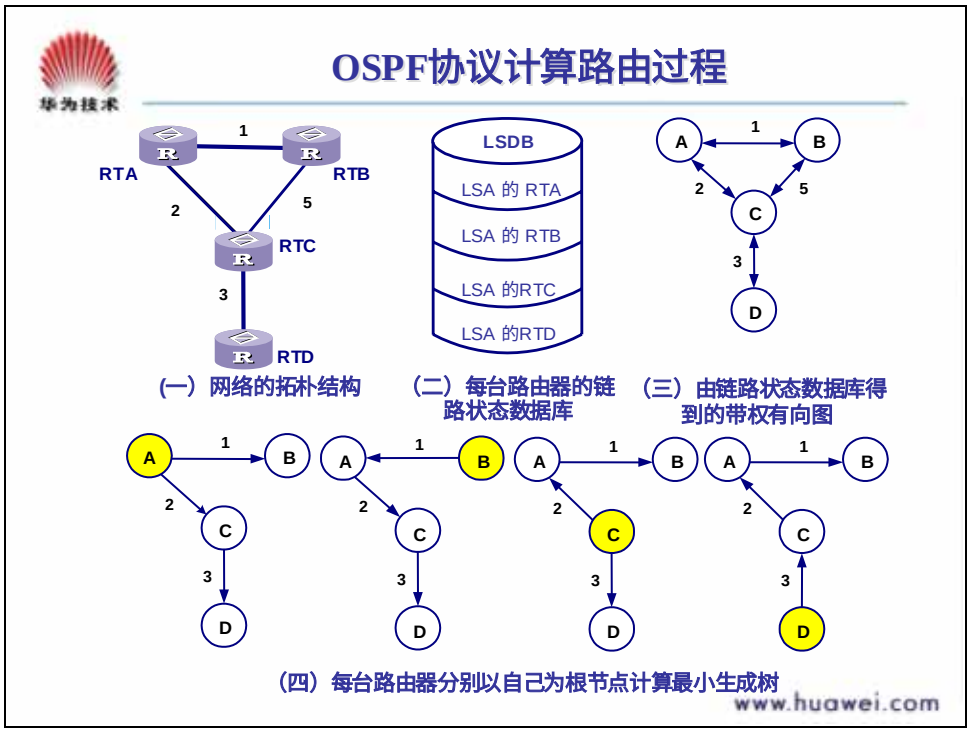
1）通过 PPP 协议与另一台路由器 RTB 直接相连 ；

2）通过一个 X.25 网络与 RTC 和 RTD 相连（该网络是全连通的）；

3）通过一个 Frame Relay 网络与 RTE 和 RTF 相连（该网络不是全连通的，RTE 与 RTF 不直接相连）；

4）直接连接着一个局域网。

7.7.4 计算路由



上图中描述了通过 OSPF 协议计算路由的过程。

（1）由四台路由器组成的网络，连线旁边的数字表示从一台路由器到另一台路由器所需要的花费。为简化问题，我们假定两台路由器相互之间发送报文所需花费是相同的。

（2）每台路由器都根据自己周围的网络拓扑结构生成一条 LSA（链路状态广播），并通过相互之间发送协议报文将这条 LSA 发送给网络中其它的所有路由器。这样每台路由器都收到了其它路由器的 LSA，所有的 LSA 放在一起称作 LSDB（链路状态数据库）。显然，4 台路由器的 LSDB 都是相同的。

（3）由于一条 LSA 是对一台路由器周围网络拓扑结构的描述，那么 LSDB 则是对整个网络的拓扑结构的描述。路由器很容易将 LSDB 转换成一张带权的有向图，这张图便是对整个网络拓扑结构的真实反映。显然，4 台路由器得到的是一张完全相同的图。

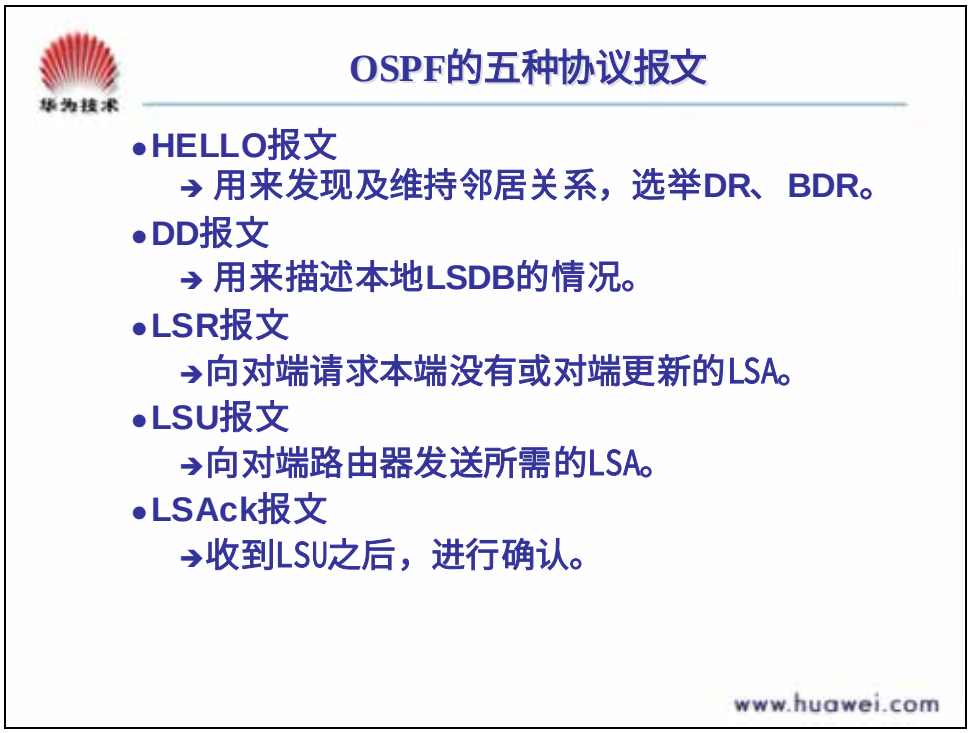
（4）接下来每台路由器在图中以自己为根节点，使用 SPF 算法计算出一棵最短路径树，由这棵树得到了到网络中各个节点的路由表。显然，4 台路由器各自得到的路由表是不同的。这样每台路由器都计算出了到其它路由器的路由。

由上面的分析可知：OSPF 协议计算出路由主要有以下三个主要步骤：

- 描述本路由器周边的网络拓扑结构，并生成 LSA。
- 将自己生成的 LSA 在自治系统中传播。并同时收集所有的其他路由器生成的 LSA。

- 根据收集的所有的 LSA 计算路由。

7.7.5 OSPF 的协议报文



OSPF 的报文类型一共有五种：

HELLO 报文（Hello Packet）：

最常用的一种报文，周期性的发送给本路由器的邻居。内容包括一些定时器的数值，DR，BDR，以及自己已知的邻居。

DD 报文（Database Description Packet）：

两台路由器进行数据库同步时，用 DD 报文来描述自己的 LSDB，内容包括 LSDB 中每一条 LSA 的摘要（摘要是指 LSA 的 HEAD，通过该 HEAD 可以唯一标识一条 LSA）。这样做是为了减少路由器之间传递信息的量，因为 LSA 的 HEAD 只占一条 LSA 的整个数据量的一小部分，根据 HEAD，对端路由器就可以判断出是否已经有了这条 LSA。

LSR 报文（Link State Request Packet）：

两台路由器互相交换过 DD 报文之后，知道对端的路由器有哪些 LSA 是本地的 LSDB 所缺少的或是对端更新的 LSA，这时需要发送 LSR 报文向对方请求所需的 LSA。内容包括所需要的 LSA 的摘要。

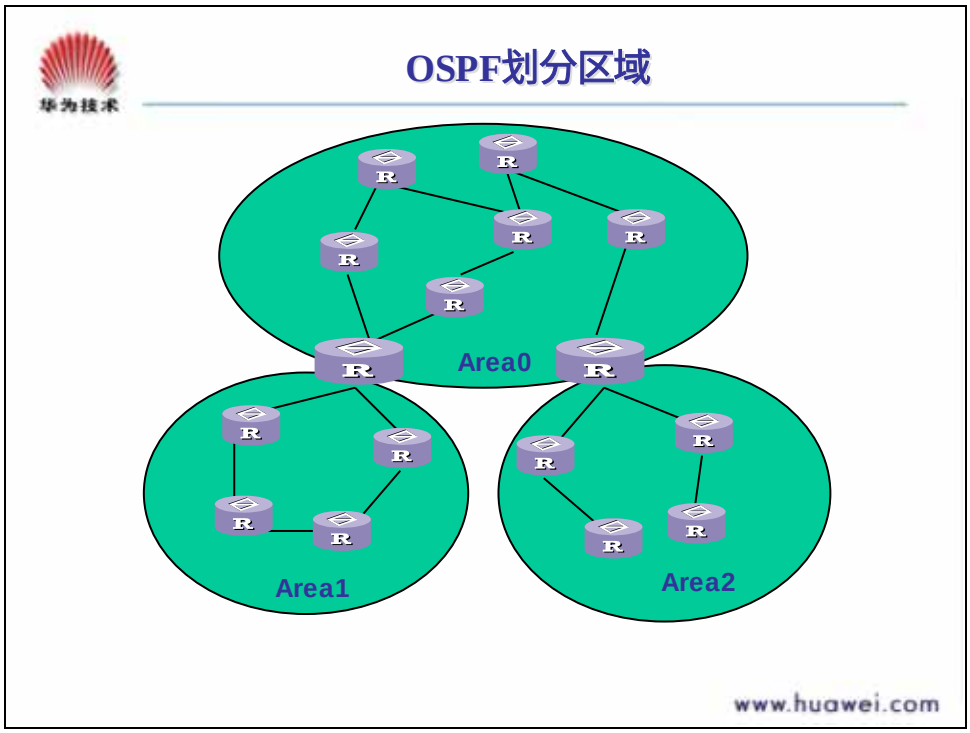
LSU 报文（Link State Update Packet）：

用来向对端路由器发送所需要的 LSA，内容是多条 LSA（全部内容）的集合。

LSAck 报文（Link State Acknowledgment Packet）

用来对接收到的 LSU 报文进行确认。内容是需要确认的 LSA 的 HEAD（一个报文可对多个 LSA 进行确认）。

7.7.6 区域划分



为什么需要划分区域：

随着网络规模日益扩大，网络中的路由器数量不断增加。当一个巨型网络中的路由器都运行 OSPF 路由协议时，就会遇到如下问题：

- 每台路由器都保留着整个网络中其他所有路由器生成的 LSA，这些 LSA 的集合组成 LSDB，路由器数量的增多会导致 LSDB 非常庞大，这会占用大量的存储空间。
- LSDB 的庞大会增加运行 SPF 算法的复杂度，导致 CPU 负担很重。
- 由于 LSDB 很大，两台路由器之间达到 LSDB 同步会需要很长时间。
- 网络规模增大之后，拓扑结构发生变化的概率也增大，网络会经常处于“动荡”之中，为了同步这种变化，网络中会有大量的 OSPF 协议报文在传递，降低了网络的带宽利用率。更糟糕的是：每一次变化都会导致网络中所有的路由器重新进行路由计算。

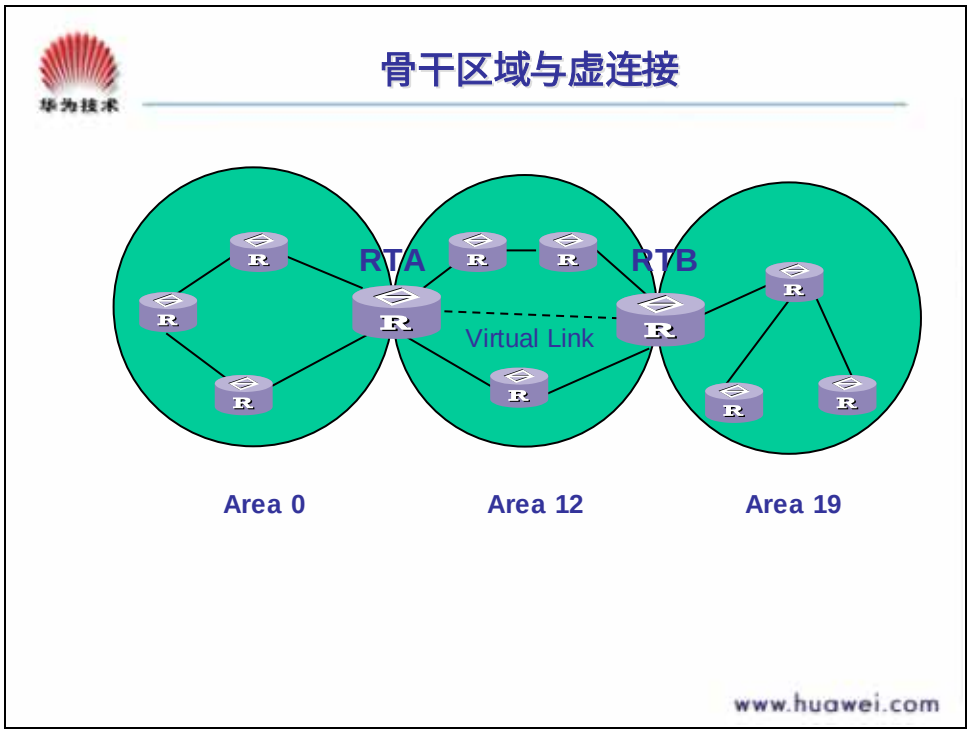
解决上述问题的关键主要有两点：减少 LSA 的数量；屏蔽网络变化波及的范围。

OSPF 协议通过将自治系统划分成不同的区域（Area）来解决上述问题。区域是在逻辑上将路由器划分为不同的组。区域的边界是路由器，这样会有一些路由器属于不同的区域，（这样的路由器称作区域边界路由器——ABR），而一个网段只能属于一个区域。

划分成区域之后，给 OSPF 协议的处理带来了很大的变化。

- 每一个网段必须属于一个区域，或者说每个运行 OSPF 协议的接口必须指名属于某一个特定的区域，区域用区域号（Area ID）来标识。区域号是一个从 0 开始的 32 位整数。
- 不同的区域之间通过 ABR 来传递路由信息。

7.7.7 骨干区域与虚连接



为何需要骨干区域：

OSPF 划分区域之后，并非所有的区域都是平等的关系。其中有一个区域是与众不同的，它的区域号（Area ID）是 0，通常被称为骨干区域（Backbone Area）。

由于划分区域之后，区域之间是通过 ABR 将一个区域内的已计算出的路由封装成 Type3 类的 LSA 发送到另一个区域之中来传递路由信息。需要注意的是：此时的 LSA 中包含的已不再是链路状态信息，而是纯粹的路由信息了。或者说，此时的 OSPF 是基于 D-V 算法，而不是基于链路状态算法的了。这就涉及到一个很重要的问题：路由自环。因为 D-V 算法无法保证消除路由自环。如果无法解决这个问题，则区域概念的提出就是失败的。

通过分析 D-V 算法中路由环的产生的原因可知，自环的产生主要是因为生成该条路由信息的路由器没有加入生成者的信息，即每一条路由信息都无法知道最初是由谁所生成。OSPF 协议在生成 LSA 时首先将自己的 Router ID 加入到 LSA 中，但是如果该路由信息传递超过两个区域后，就会丧失最初的生成者的信息。

解决的方法是：所有 ABR 将本区域内的路由信息封装成 LSA 后，统一的发送给一个特定的区域，再由该区域将这些信息转发给其他区域。在这个特定区域内，每一条 LSA 都确切的知道生成者信息。在其他区域内所有的到区域外的路由都会发送到这个特定区域中，所以就不会产生路由自环。这个“特定区域”就是骨干区域。由上面的分析可知：所有的区域必须和骨干区域相连，也就是说，每一个 ABR 连接的区域中至少有一个是骨干区域。而且骨干区域自身也必须是连通的。

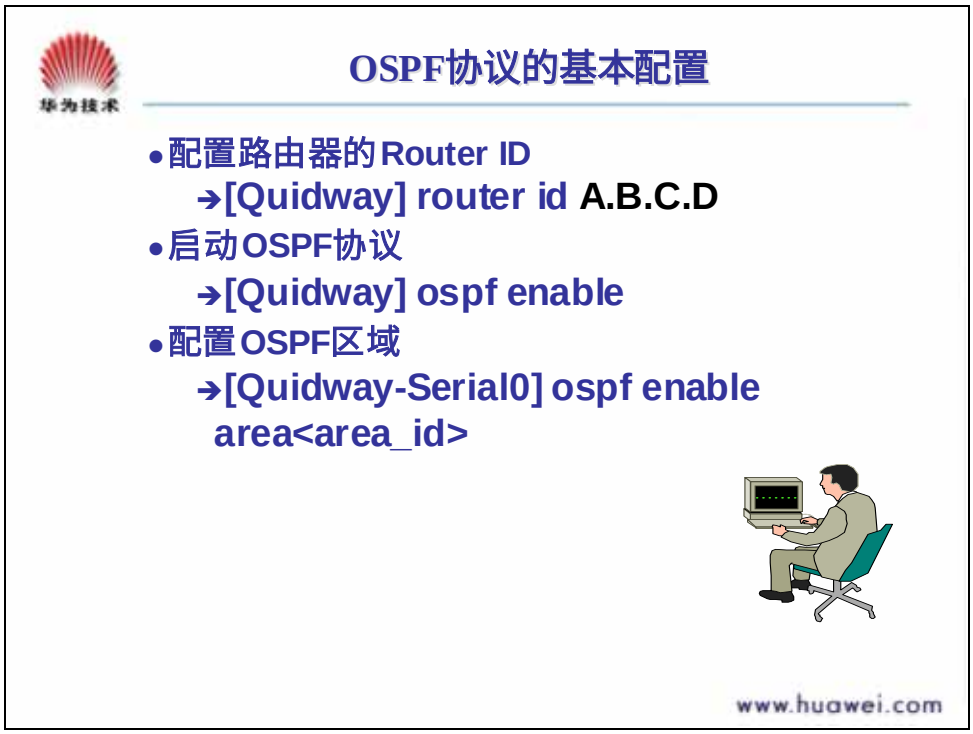
虚连接：

由于网络的拓扑结构复杂，有时无法满足每个区域必须和骨干区域直接相连的要求，例如图中的 Area 19。 为解决此问题，OSPF 提出了虚连接的概念。虚连接是指在两台 ABR 之间，穿过一个非骨干区域（转换区域——transit Area），建立的一条逻辑上的连接通道。可以理解为两台 ABR 之间存在一个点对点的连接。“逻辑通道”是指两台 ABR 之间的多台运行 OSPF 的路由器只是起到一个转发报文的作用（由于协议报文的目的地址不是这些路由器，所以这些报文对于他们是透明的，只是当作普通的 IP 报文来转发），两台 ABR 之间直接传递路由信息。这里的路由信息是指由 ABR 生成的 type3 的 LSA，区域内的路由器同步方式没有因此改变。

注意：

如果自治系统被划分成一个以上的区域，则必须有一个区域是骨干区域，并且保证其它区域与骨干区域直接相连或逻辑上相连，且骨干区域自身也必须是连通的。

7.7.8 OSPF 协议的基本配置



配置路由器的 Router ID：

Router ID 是每一台路由器在自治系统中的唯一标识，OSPF 协议能够正常运行的前提条件是该路由器已经存在一个 Router ID。通常是由图中的命令手工配置，Router ID 是一个 32bit 的整数，配置时应输入类似 IP 地址的点分十进制格式。如果用户没有手工指定 Router ID，系统会自动从当前 UP 的接口的 IP 地址中选一个最小的。自动选举的 Router ID 会随着 IP 地址的变化而改变，这样会干扰协议的正常运行。**所以强烈建议：手工指定 Router ID**。但需要注意的一点是：手工指定 Router ID 时必须保证自治系统中没有两台路由器的 Router ID 是相同的。通常的做法是将 Router ID 设置成与本路由器的某个接口（如以太网）的 IP 地址相同，因为 IP 地址是全网唯一的。

启动 OSPF 协议：

一台路由器如果要运行 OSPF 协议，必须首先在全局配置模式下启动该协议。

配置 OSPF 区域：

必须为每一个要运行 OSPF 协议的接口指定一个区域。

在接口模式下用：ospf enable area <area_id> 配置该接口属于某个区域。

例如：某路由器有一个接口 S0，欲将其配置属于区域 2 。配置如下 ：

[Quidway-Serial0]ospf enable area 2