

讲堂 > 趣谈网络协议 > 文章详情

第9讲 | 路由协议：西出网关无故人，敢问路在何方

2018-06-06 刘超



第9讲 | 路由协议：西出网关无故人，敢问路在何方

朗读人：刘超 17'19" | 7.96M

俗话说得好，在家千日好，出门一日难。网络包一旦出了网关，就像玄奘西行一样踏上了江湖漂泊的路。

上一节我们描述的是一个相对简单的情形。出了网关之后，只有一条路可以走。但是，网络世界复杂得多，一旦出了网关，会面临着很多路由器，有很多条道路可以选。如何选择一个更快速的道路求取真经呢？这里面还有很多门道可以讲。

如何配置路由？

通过上一节的内容，你应该已经知道，路由器就是一台网络设备，它有多张网卡。当一个入口的网络包送到路由器时，它会根据一个本地的转发信息库，来决定如何正确地转发流量。这个转发信息库通常被称为**路由表**。

一张路由表中会有多条路由规则。每一条规则至少包含这三项信息。

- 目的网络：这个包想去哪儿？

- 出口设备：将包从哪个口扔出去？
- 下一跳网关：下一个路由器的地址。

通过 `route` 命令和 `ip route` 命令都可以进行查询或者配置。

例如，我们设置 `ip route add 10.176.48.0/20 via 10.173.32.1 dev eth0`，就说明要去 10.176.48.0/20 这个目标网络，要从 eth0 端口出去，经过 10.173.32.1。

上一节的例子中，网关上的路由策略就是按照这三项配置信息进行配置的。这种配置方式的一个核心思想是：**根据目的 IP 地址来配置路由**。


如何配置策略路由？

当然，在真实的复杂的网络环境中，除了可以根据目的 ip 地址配置路由外，还可以根据多个参数来配置路由，这就称为**策略路由**。

可以配置多个路由表，可以根据源 IP 地址、入口设备、TOS 等选择路由表，然后在路由表中查找路由。这样可以使得来自不同来源的包走不同的路由。

例如，我们设置：


```
1 ip rule add from 192.168.1.0/24 table 10
2 ip rule add from 192.168.2.0/24 table 20
```

 复制代码

表示从 192.168.1.0/24 这个网段来的，使用 table 10 中的路由表，而从 192.168.2.0/24 网段来的，使用 table 20 的路由表。

在一条路由规则中，也可以走多条路径。例如，在下面的路由规则中：

```
1 ip route add default scope global nexthop via 100.100.100.1 weight 1 nexthop via 200.200.200.1 w
```

 复制代码

下一跳有两个地方，分别是 100.100.100.1 和 200.200.200.1，权重分别为 1 比 2。

在什么情况下会用到如此复杂的配置呢？我来举一个现实中的例子。

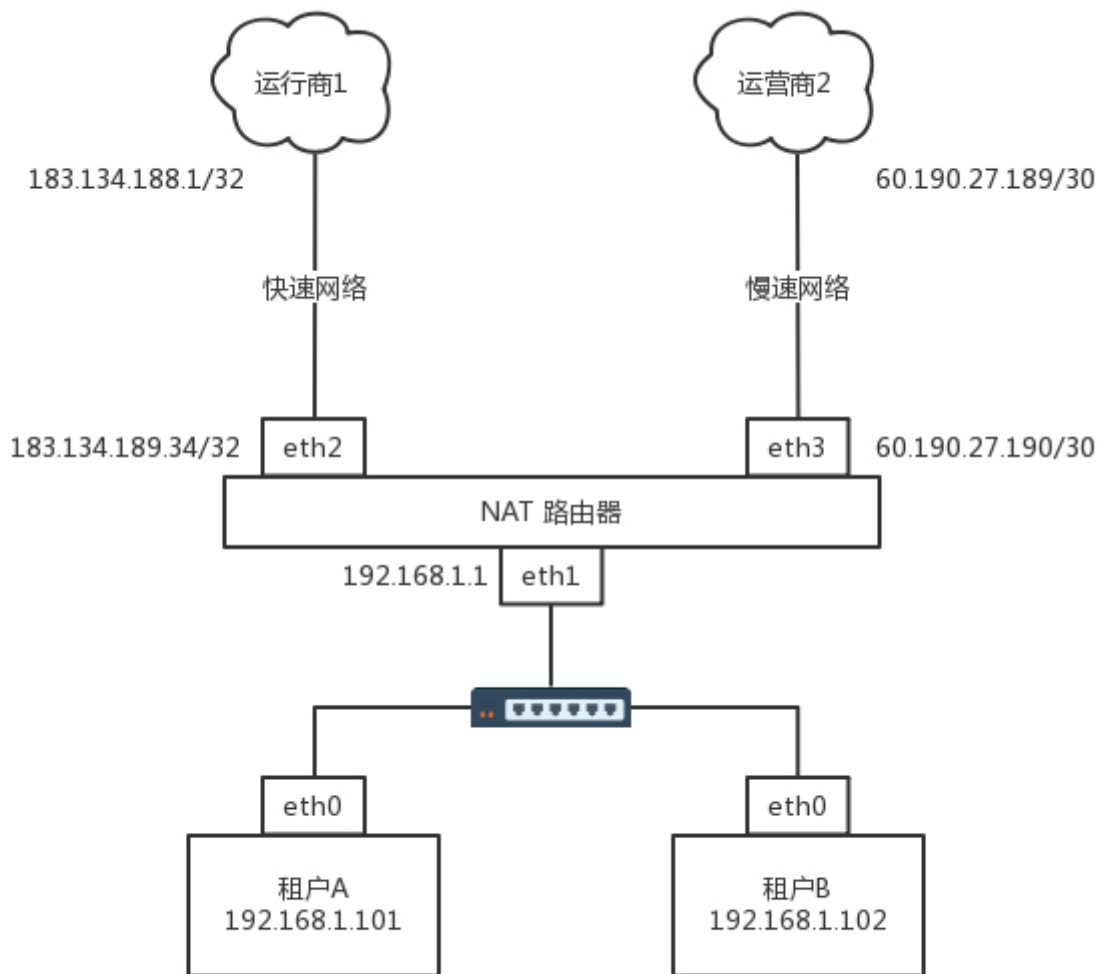
我是房东，家里从运营商那儿拉了两根网线。这两根网线分别属于两个运行商。一个带宽大一些，一个带宽小一些。这个时候，我就不能买普通的家用路由器了，得买个高级点的，可以接两个外网的。

家里的网络呢，就是普通的家用网段 192.168.1.x/24。家里有两个租户，分别把线连到路由器上。IP 地址为 192.168.1.101/24 和 192.168.1.102/24，网关都是 192.168.1.1/24，网关在路

由器上。

就像上一节说的一样，家里的网段是私有网段，出去的包需要 NAT 成公网的 IP 地址，因而路由器是一个 NAT 路由器。

两个运营商都要为这个网关配置一个公网的 IP 地址。如果你去查看你们家路由器里的网段，基本就是我图中画的样子。



运行商里面也有一个 IP 地址，在运营商网络里面的网关。不同的运营商方法不一样，有的是 /32 的，也即一个一对一连接。

例如，运营商 1 给路由器分配的地址是 183.134.189.34/32，而运营商网络里面的网关是 183.134.188.1/32。有的是 /30 的，也就是分了一个特别小的网段。运营商 2 给路由器分配的地址是 60.190.27.190/30，运营商网络里面的网关是 60.190.27.189/30。

根据这个网络拓扑图，可以将路由配置成这样：

```
1 $ ip route list table main
```

复制代码

```
2 60.190.27.189/30 dev eth3 proto kernel scope link src 60.190.27.190
3 183.134.188.1 dev eth2 proto kernel scope link src 183.134.189.34
4 192.168.1.0/24 dev eth1 proto kernel scope link src 192.168.1.1
5 127.0.0.0/8 dev lo scope link
6 default via 183.134.188.1 dev eth2
```

当路由这样配置的时候，就告诉这个路由器如下的规则：

- 如果去运营商二，就走 eth3；
- 如果去运营商一呢，就走 eth2；
- 如果访问内网，就走 eth1；
- 如果所有的规则都匹配不上，默认走运营商一，也即走快的网络。

但是问题来了，租户 A 不想多付钱，他说我就上上网页，从不看电影，凭什么收我同样贵的网费啊？没关系，咱有技术可以解决。

下面我添加一个 Table，名字叫**chao**。

```
1 # echo 200 chao >> /etc/iproute2/rt_tables
```

[复制代码](#)

添加一条规则：

```
1 # ip rule add from 192.168.1.101 table chao
2 # ip rule ls
3 0:      from all lookup local
4 32765:  from 10.0.0.10 lookup chao
5 32766:  from all lookup main
6 32767:  from all lookup default
```

[复制代码](#)

设定规则为：从 192.168.1.101 来的包都查看个 chao 这个新的路由表。

在 chao 路由表中添加规则：

```
1 # ip route add default via 60.190.27.189 dev eth3 table chao
2 # ip route flush cache
```

[复制代码](#)

默认的路由走慢的，谁让你不付钱。

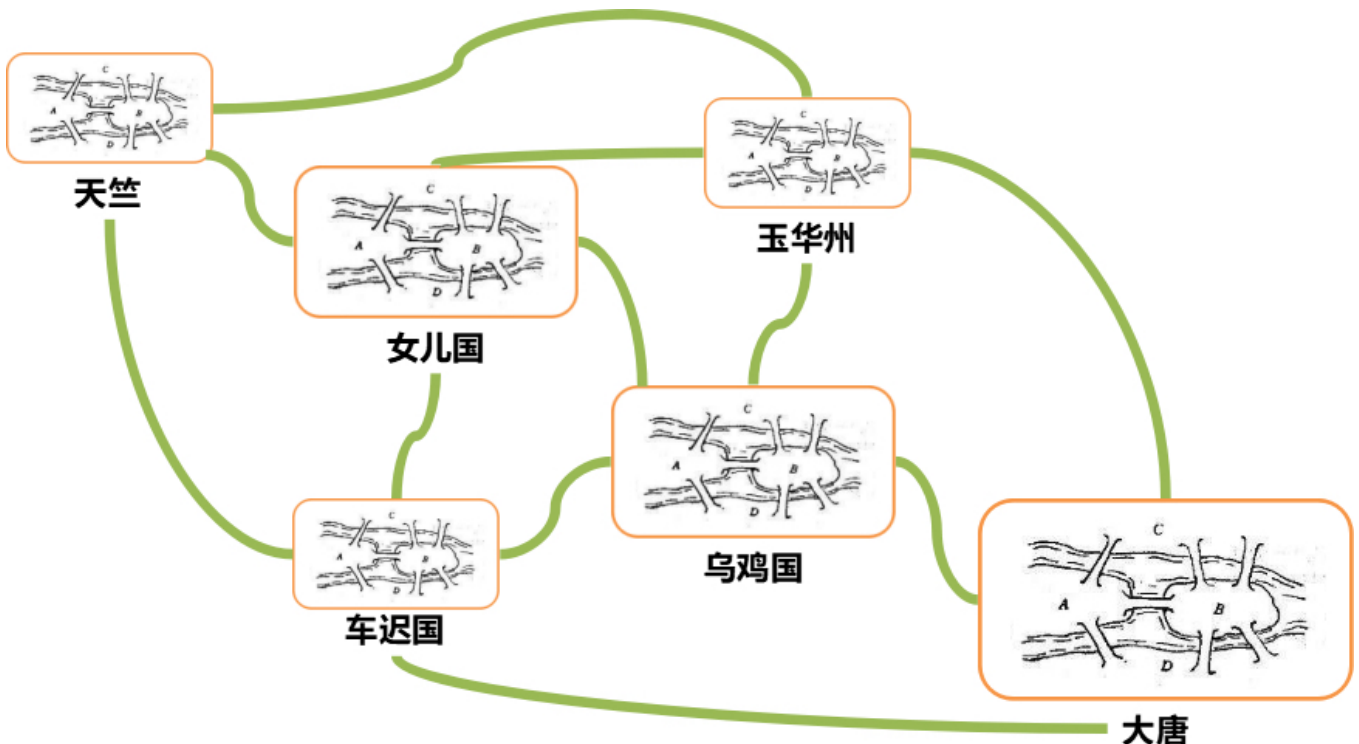
上面说的都是静态的路由，一般来说网络环境简单的时候，在自己的可控范围之内，自己捣鼓还是可以的。但是有时候网络环境复杂并且多变，如果总是用静态路由，一旦网络结构发生变化，

让网络管理员手工修改路由太复杂了，因而需要动态路由算法。

动态路由算法

使用动态路由路由器，可以根据路由协议算法生成动态路由表，随网络运行状况的变化而变化。那路由算法是什么样的呢？

我们可以想象唐僧西天取经，需要解决两大问题，一个是在每个国家如何找到正确的路，去换通关文牒、吃饭、休息；一个是在国家之间，野外行走的时候，如何找到正确的路、水源的问题。



无论是一个国家内部，还是国家之间，我们都可以将复杂的路径，抽象为一种叫作图的数据结构。至于唐僧西行取经，肯定想走得路越少越好，道路越短越好，因而这就转化成为**如何在途中找到最短路径**的问题。

咱们在大学里面学习计算机网络与数据结构的时候，知道求最短路径常用的有两种方法，一种是 Bellman-Ford 算法，一种是 Dijkstra 算法。在计算机网络中基本也是用这两种方法计算的。

1. 距离矢量路由算法

第一大类的算法称为**距离矢量路由 (distance vector routing)**。它是基于 Bellman-Ford 算法的。

这种算法的基本思路是，每个路由器都保存一个路由表，包含多行，每行对应网络中的一个路由器，每一行包含两部分信息，一个是要到目标路由器，从那条线出去，另一个是到目标路由器的距离。

由此可以看出，每个路由器都是知道全局信息的。那这个信息如何更新呢？每个路由器都知道自己和邻居之间的距离，每过几秒，每个路由器都将自己所知的到达所有的路由器的距离告知邻

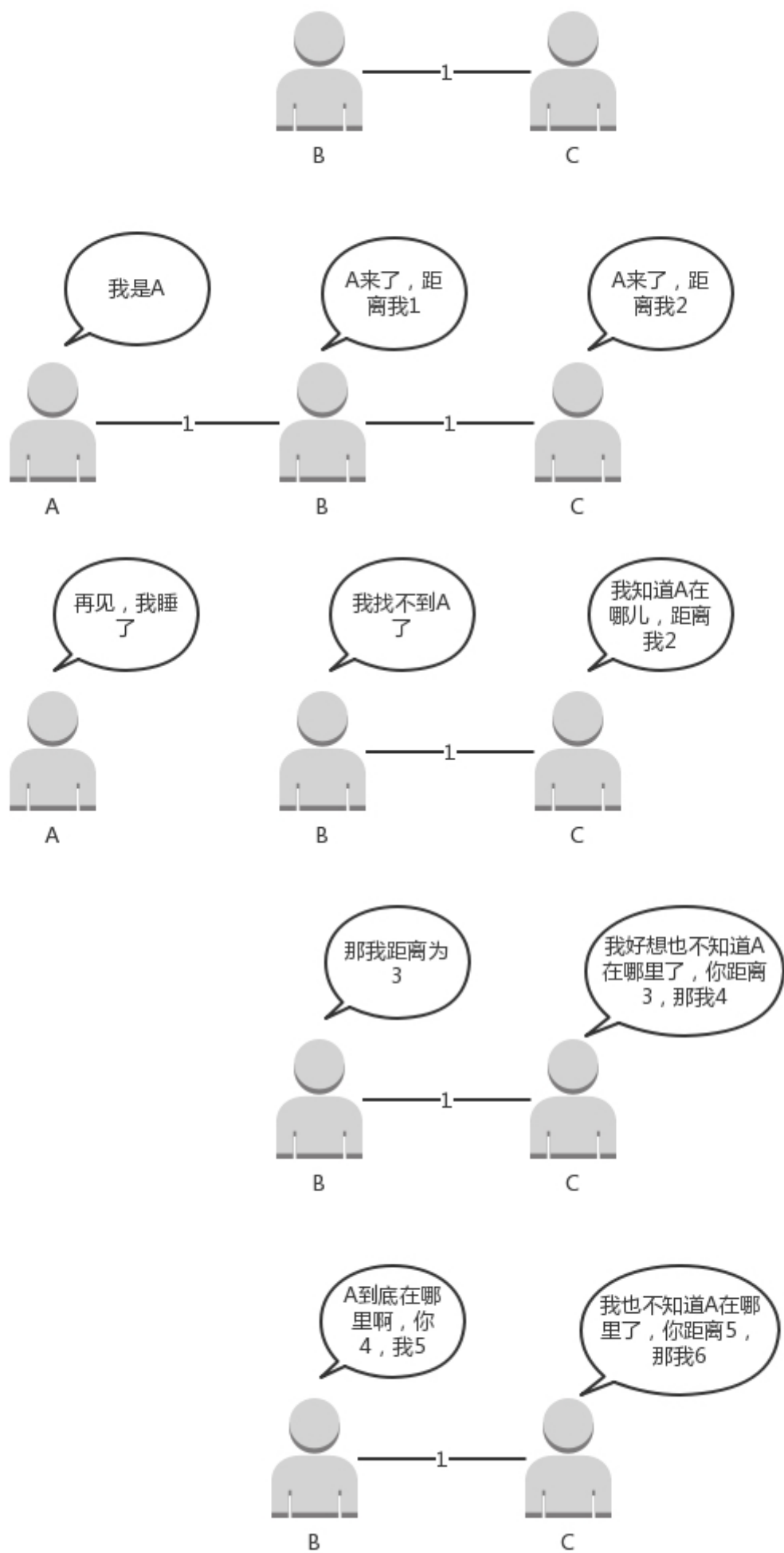
居，每个路由器也能从邻居那里得到相似的信息。

每个路由器根据新收集的信息，计算和其他路由器的距离，比如自己的一个邻居距离目标路由器的距离是 M ，而自己距离邻居是 x ，则自己距离目标路由器是 $x+M$ 。

这个算法比较简单，但是还是有问题。

第一个问题就是好消息传得快，坏消息传得慢。如果有个路由器加入了这个网络，它的邻居就能很快发现它，然后将消息广播出去。要不了多久，整个网络就都知道了。但是一旦一个路由器挂了，挂的消息是没有广播的。当每个路由器发现原来的道路到不了这个路由器的时候，感觉不到它已经挂了，而是试图通过其他的路径访问，直到试过了所有的路径，才发现这个路由器是真的挂了。

我再举个例子。



原来的网络包括两个节点，B 和 C。A 加入了网络，它的邻居 B 很快就发现 A 启动起来了。于是它将自己和 A 的距离设为 1，同样 C 也发现 A 起来了，将自己和 A 的距离设置为 2。但是如果 A 挂掉，情况就不妙了。B 本来和 A 是邻居，发现连不上 A 了，但是 C 还是能够连上，只不过距离远了点，是 2，于是将自己的距离设置为 3。殊不知 C 的距离 2 其实是基于原来自己的距离为 1 计算出来的。C 发现自己也连不上 A，并且发现 B 设置为 3，于是自己改成距离 4。依次类推，数越来越大，直到超过一个阈值，我们才能判定 A 真的挂了。

这个道理有点像有人走丢了。当你突然发现找不到这个人了。于是你去学校问，是不是在他姨家呀？找到他姨家，他姨说，是不是在他舅舅家呀？他舅舅说，是不是在他姥姥家呀？他姥姥说，是不是在学校呀？总归要问一圈，或者是超过一定的时间，大家才会认为这个人的确走丢了。如果这个人其实只是去见了一个谁都不认识的网友去了，当这个人回来的时候，只要他随便见到其中的一个亲戚，这个亲戚就会拉着他到他的家长那里，说你赶紧回家，你妈都找你一天了。

这种算法的第二个问题是，每次发送的时候，要发送整个全局路由表。网络大了，谁也受不了，所以最早的路由协议 RIP 就是这个算法。它适用于小型网络（小于 15 跳）。当网络规模都小的时候，没有问题。现在一个数据中心内部路由器数目就很多，因而不适用了。

所以上面的两个问题，限制了距离矢量路由的网络规模。

2. 链路状态路由算法

第二大类算法是**链路状态路由（link state routing）**，基于 Dijkstra 算法。

这种算法的基本思路是：当一个路由器启动的时候，首先是发现邻居，向邻居 say hello，邻居都回复。然后计算和邻居的距离，发送一个 echo，要求马上返回，除以二就是距离。然后将自己和邻居之间的链路状态包广播出去，发送到整个网络的每个路由器。这样每个路由器都能够收到它和邻居之间的关系的信息。因而，每个路由器都能在自己本地构建一个完整的图，然后针对这个图使用 Dijkstra 算法，找到两点之间的最短路径。

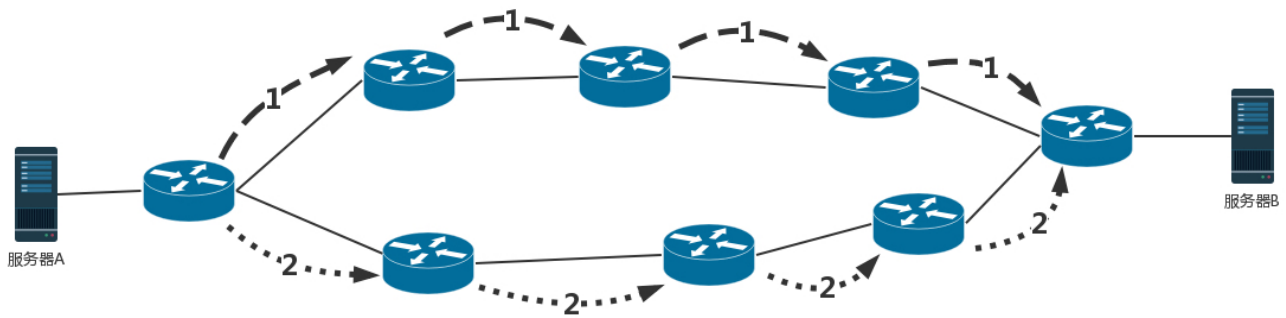
不像距离距离矢量路由协议那样，更新时发送整个路由表。链路状态路由协议只广播更新的或改变的网络拓扑，这使得更新信息更小，节省了带宽和 CPU 利用率。而且一旦一个路由器挂了，它的邻居都会广播这个消息，可以使得坏消息迅速收敛。

动态路由协议

1. 基于链路状态路由算法的 OSPF

OSPF（Open Shortest Path First，开放式最短路径优先）就是这样一个基于链路状态路由协议，广泛应用在数据中心中的协议。由于主要用在数据中心内部，用于路由决策，因而称为**内部网关协议（Interior Gateway Protocol，简称IGP）**。

内部网关协议的重点就是找到最短的路径。在一个组织内部，路径最短往往最优。当然有时候 OSPF 可以发现多个最短的路径，可以在这多个路径中进行负载均衡，这常常被称为**等价路由**。



这一点非常重要。有了等价路由，到一个地方去可以有相同的两个路线，可以分摊流量，还可以当一条路不通的时候，走另外一条路。这个在后面我们讲数据中心的网络的时候，一般应用的接入层会有负载均衡 LVS。它可以和 OSPF 一起，实现高吞吐量的接入层设计。

有了内网的路由协议，在一个国家内，唐僧可以想怎么走怎么走了，两条路选一条也行。

2. 基于距离矢量路由算法的 BGP

但是外网的路由协议，也即国家之间的，又有所不同。我们称为**外网路由协议（Border Gateway Protocol，简称BGP）**。

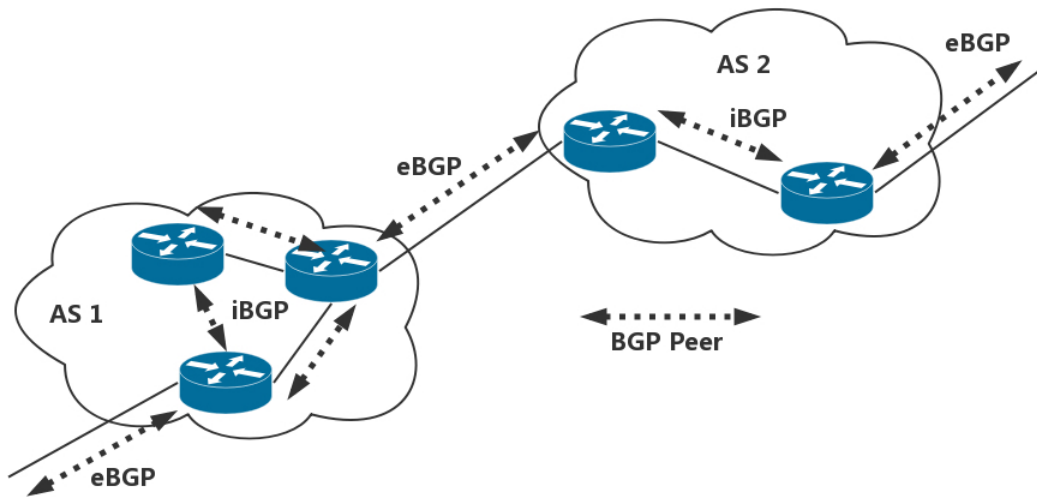
在一个国家内部，有路当然选近的走。但是国家之间，不光远近的问题，还有政策的问题。例如，唐僧去西天取经，有的路近。但是路过的国家看不惯僧人，见了僧人就抓。例如灭法国，连光头都要抓。这样的情况即便路近，也最好绕远点走。

对于网络包同样，每个数据中心都设置自己的 Policy。例如，哪些外部的 IP 可以让内部知晓，哪些内部的 IP 可以让外部知晓，哪些可以通过，哪些不能通过。这就好比，虽然从我家里到目的地最近，但是不能谁都能从我家走啊！

在网络世界，这一个个国家成为自治系统**AS（Autonomous System）**。自治系统分几种类型。

- Stub AS：对外只有一个连接。这类 AS 不会传输其他 AS 的包。例如，个人或者小公司的网络。
- Multihomed AS：可能有多个连接连到其他的 AS，但是大多拒绝帮其他的 AS 传输包。例如一些大公司的网络。
- Transit AS：有多个连接连到其他的 AS，并且可以帮助其他的 AS 传输包。例如主干网。

每个自治系统都有边界路由器，通过它和外面的世界建立联系。



BGP 又分为两类，eBGP 和 iBGP。自治系统间，边界路由器之间使用 eBGP 广播路由。内部网络也需要访问其他的自治系统。边界路由器如何将 BGP 学习到的路由导入到内部网络呢？就是通过运行 iBGP，使得内部的路由器能够找到到达外网目的地的最好的边界路由器。

BGP 协议使用的算法是**路径矢量路由协议**（path-vector protocol）。它是距离矢量路由协议的升级版。

前面说了距离矢量路由协议的缺点。其中一个收敛慢。在 BGP 里面，除了下一跳 hop 之外，还包括了自治系统 AS 的路径，从而可以避免坏消息传的慢的问题，也即上面所描述的，B 知道 C 原来能够到达 A，是因为通过自己，一旦自己都到达不了 A 了，就不用假设 C 还能到达 A 了。

另外，在路径中将一个自治系统看成一个整体，不区分自治系统内部的路由器，这样自治系统的数目是非常有限的。就像大家都能记住出去玩，从中国出发先到韩国然后到日本，只要不计算细到具体哪一站，就算是发送全局信息，也是没有问题的。

小结

好了，这一节就到这里了，我来做个总结：

- 路由分静态路由和动态路由，静态路由可以配置复杂的策略路由，控制转发策略；
- 动态路由主流算法有两种，距离矢量算法和链路状态算法。基于两种算法产生两种协议，BGP 协议和 OSPF 协议。

最后，再给你留两个思考题：

1. 路由协议要在路由器之间交换信息，这些信息的交换还需要走路由吗？不是死锁了吗？
2. 路由器之间信息的交换使用什么协议呢？报文格式是什么样呢？

欢迎你留言和我讨论。趣谈网络协议，我们下期见！



©版权归极客邦科技所有，未经许可不得转载

上一篇 第8讲 | 世界这么大，我想出网关：欧洲十国游与玄奘西行

下一篇 第10讲 | UDP协议：因性善而简单，难免碰到“城会玩”

写留言

精选留言



zcpromising

6

老师，更新前，能否解释下上一期的问题吗

2018-06-06

作者回复

问题分两个，第一个往往是这一章的延伸思考题，一般是需要一节时间才能解释清楚的，所以不适合第二期回答，所以可能会另写文章的方式回答，第二个问题往往就是下一节内容。目前一周三篇压力还是挺大的，基本周末都进去了，所以另写的文章慢慢放出，望体谅，谢谢

2018-06-06



埃罗芒阿老师

10

BGP基于TCP，OSPF基于UDP

2018-06-06



灰灰

9

都是干货，本来是来读小说的，不知不觉学了很多东西，作为一个从事系统开发的女生表示很受用。

2018-06-07

作者回复

谢谢

2018-06-07



sprinty

7

老师您好，运营商1和eth2的连接是一对一的是什么意思？/32 的话，子网掩码是255.255.255.255，eth2和运营商并不在一个网络，是怎么通信的？

2018-06-06

作者回复

p2p

2018-06-06



小田

4

#提纲

1. 静态路由实现原理

- 3项路由：目标ip，出口ip，下一跳网关IP

- 策略路由：多路由表/多路径

2. 动态路由实现原理

- 距离矢量路由算法：BGP

- 链路状态路由算法：OSPF

2018-06-07



姜戈

4

学过网络的人表示很形象，相见恨晚

2018-06-06

作者回复

谢谢

2018-06-06



林先森

3

请问下老师，都说lastmile是网络链路中延迟最大的，也是是业界最难解决的问题，能解释下lastmile为什么会有比较大的延迟吗？

2018-06-06



赵强强

3

老师，距离矢量路由算法避免不了循环更新路由吗？比如上面举例B和C，当B到不了A时，此时B把距离变更为不可达，当C向B发送路由信息表时，B应该可以发现C到A的下一跳是自己，应该不更新自己就好了，并告诉C路由失效，就可以避免无用的循环更新。请老师解答一下。

2018-06-06

作者回复

因为没有下一跳的信息，如果有这个相当于记录了路径了

2018-06-06



heliang

👍 2

老师你好，文章这部分

32765: from.10.0.0.10 lookup chao

中10.0.0.10的地址是不是应改为租户A的地址:192.168.1.101

2018-09-23

作者回复

是的，谢谢指正

2018-09-25



eason2017

👍 2

老师好，您路由协议这块可以再深入讲解一节吗？谢谢🙏

2018-08-01



一步

👍 2

没学习过网络的，表示有点蒙

只能学习到有这个东西，具体的就不行了

2018-07-02



番茄尼玛

👍 2

rip是udp协议，ospf直接发ip包。而bgp使用tcp协议，路由器之间会建立tcp连接，每60s发送一次keep-alive消息。

2018-06-22



William

👍 1

BGP用到的路径矢量路由协议是距离矢量路由协议的升级版，这个不太理解。老师有没有推荐的资料呢？

2018-11-20

作者回复

就是保存了路径，计算机网络 书籍里面都有

2018-11-20



武坤

👍 1

本文说的路由距离，是指响应的时间吗？

2018-06-15



文子

👍 1

请教下，在开篇简单路由器是有多个网卡的，记得路由器网络设置里分为lan口和wlan口设置，这两口的mac地址不同，说明lan和wlan是两个网卡，但是路由器是有多个lan口的，发

现lan口的mac地址是统一的，这是不是说明lan是一个网卡，且有多个网口；wlan是一个网卡？还有就是lan和wlan是两个网卡，他们之间的网络包传递是怎么实现的？谢谢！

2018-06-07

作者回复

是的，家用路由器虽然多个口，但是其实像我们示意图里面一样，是一个地址加交换机的形式。wlan和lan要过路由

2018-06-07



Zend

1

在公司内网的测试环境 就遇到 多个部署微服务的物理机网段不一样，但http服务走浏览器访问能通，但机器之间ping不通，这是为什么啊

2018-06-06



docker

1

顶

2018-06-06



拿笔小星

0

不是从事网络相关工作的，纯JAVA开发，感觉越到后面越吃力！老师，对于我这种几乎的小白，有什么建议吗？

2018-11-25



疯琴

0

收获不小，谢谢，请问igp和ibgp是什么区别？

2018-11-23



阿吉

0

9动静路由

手动配置路由表或策略

命令：route,IP route

例：两个租客不同路由策略

路由的三个关键信息

目的地址，出口地址，下一跳地址

路由器的基本算法都是建立在图论之上

静态路由

路由路径基本不变

动态路由

路由路径根据提前设好的策略自动调整

distance vector routing

距离矢量路由，基于bellman-ford算法(图论)

路由表：所有相邻路由信息

缺点：

有路由器挂了发现很慢

定期更新整张路由表，效率不高

相关协议算法：

RIP(早期算法，适合较小的网络，小于15跳)

BGP(border gateway protocol)

link state routing

链路状态路由，基于Dijkstra 算法(图论)。

特点：变量更新

路由均衡

相关协议算法：

IGP(Interior gateway protocol)

OSPF(open shortest path first)

等价路由

负载均衡(LVS)

影响routing policy 其他因素

每个数据中心或子网对内和对外都会有些限制，

对内部而言有些外网IP不可访问

(天朝防火墙)

对外部而言有些内网IP是不可访问量的

(数据中心内部IP不对外直接访问)

AS(autonomous system)自治系统

stub as 单路对外链接，不转发 个人小公司

multihomed as 多路对外链接 不转发 小公司

transit as 转发 骨干网

2018-11-14