

具有规模性的 Anycast 算法实现

张华健, 杨 健

(南京邮电大学 物联网学院, 江苏 南京 210003)

摘 要:为了解决因在 IPv6 网络层中引入 Anycast 通信服务而引起的互联网中的路由器路由表项的规模性问题,文中给出了利用 Anycast 单独路由和预先分配固定 Anycast 地址以及 Anycast 地址归并和拆分算法几项技术。通过理论分析这几项技术较好地解决了因为引入 Anycast 而引起的 Unicast 路由表的急剧扩大以及路由性能急剧下降的问题,较好地解决了 Anycast 实现中遇到的规模性问题并提供了灵活的选择标准。文中给出的相关技术能够解决当前 IPv6 网络中 Anycast 实现中遇到的规模性问题,为 IPv6 的应用建立了基础。

关键词:Anycast; IPv6; Unicast; 规模性

中图分类号:TP301.6

文献标识码:A

文章编号:1673-629X(2013)07-0120-03

doi:10.3969/j.issn.1673-629X.2013.07.030

Realization of Anycast Algorithm with Scale

ZHANG Hua-jian, YANG Jian

(College of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing 210003, China)

Abstract:In order to solve the scale question of the routing table entries in the router due to the introduction of IPv6 network layer Anycast communications services, give several technologies which use separate routing Anycast, pre-allocate a fixed Anycast address and Anycast address merge and split algorithm to solve the scale question. Through theoretical analysis these technologies solve the question that Unicast's router tables become big greatly and that the performance of routers declines greatly due to the introduction of Anycast. These technologies solve the scale question and gives flexible selection criteria. It gives several technologies which solve the scale question that appears in IPv6 network, and also establishes a foundation for IPv6.

Key words:Anycast; IPv6; Unicast; scale

0 引言

Anycast 是 IPv6 协议引入的一种新服务^[1,2]。RFC2723 将 IPv6 地址结构中的 Anycast 地址定义为一系列网络接口(通常属于不同的节点)的标识,其特点是:发往一个 Anycast 地址的分组将被转发到由该地址标识的“最近”^[1]的一个网络接口(“最近”的定义是基于路由协议中的距离度量,度量单位可以为时延、带宽、路由跳数等)。单一的 Anycast 地址被分配给多个节点(Anycast 成员);每次仅有一个分配 Anycast 地址的成员与发送端通信。图 1 说明了 Anycast 的这种功能。

图 1 中 Sender1 和 Sender2 都向同一个 Anycast 地址发出了请求包,但是包被网络转发到离发送者最近的组成员^[3]。

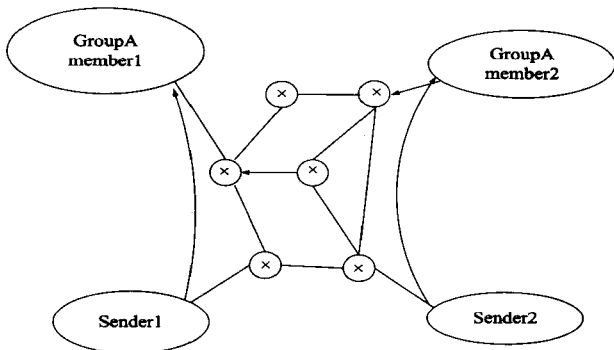


图 1 Anycast 原理图

1 Anycast 技术的应用与优势

服务器定位是 Anycast 技术应用的主要地方,分布的服务共享相同的 IP 地址,这个 IP 地址是服务的 Anycast 地址。发送端的主机可以在众多的同样功能

收稿日期:2012-03-06

修回日期:2012-08-10

网络出版时间:2013-03-05

基金项目:国家自然科学基金资助项目(60973139,61170065)

作者简介:张华健(1977-),男,河南人,硕士,工程师,研究方向为网络技术、组播技术。

网络出版地址:<http://www.cnki.net/kcms/detail/61.1450.TP.20130305.0814.005.html>

的主机中选择其中的一个,利用 Anycast 路由方法将 Anycast 需求均匀地分配到分布式服务的各个主机上,就可以达到分布式服务中的负载均衡功能^[4]。Anycast 路由系统选择了“最近”的服务,缩短了服务响应时间,同时减轻了网络负载;相同的服务在网络上形成了冗余,Anycast 路由系统可以选择负载相对轻的、带宽相对高的路径转发报文,这样就带来了两个方面的好处:

- 1)减轻 DOS 攻击对用户带来的影响。
- 2)减轻了网络拥塞给用户带来的影响。

2 Anycast 实现类型

Anycast 的实现可以分为两类:Subnet Anycast 和 Global Anycast^[5]。Subnet Anycast 是指所有的服务器主机都位于同一个网段,按这种方式实现的类型可以称为 Subnet Anycast,即集中部署的方式;Global Anycast 是指服务器主机不在同一个网段,可能在同一个节点的两个或者更多的网段,可能根本都不在同一个节点,而分散于多个节点,这些节点可能在同一个城市,也可能在多个城市,也可能多个省甚至多个国家,这种方式实现的 Anycast 可以称为 Global Anycast,亦称为分布式部署方式。

3 实现 Anycast 路由的问题

目前现存 Anycast 技术的标准定义路由协议都不清楚,因此在设计 IPv6 Anycast 路由协议时还存在一些相关的技术问题。

3.1 可量测性问题

由于 Anycast 成员不管它们实际的前缀时位置是分散的,所以针对 Anycast 地址的路由条目无法被聚集。因此针对 Anycast 地址的路由条目必须分别被存储在路由器中,当 Anycast 地址得到广泛应用时,路由表将会变得十分的拥挤。

3.2 安全问题

维持 Anycast 成员关系特别重要,对主机获得成员资格最简单的方式是它只需要广播针对相关 Anycast 地址进入路由器的路由条目。但是这种方法有时会导致严重的安全问题,即 Anycast 主机可以自由地在路由表中添加或者删除条目。

4 算法描述和实现

Anycast 地址从单播地址空间中分派出来^[6],它的范围就是单播地址类型的范围。无法确定给定的目标单播地址会不会也是 Anycast 地址。唯一知道实际情况的只有路由器以及 Anycast 组成员本身。基于以上事实出现了一些问题,因为 Anycast 地址是从单播地址

空间中划分出来的,如果 Anycast 地址分散在单播地址空间中,就不利于在路由 Anycast 数据包时路由表中 Anycast 路由项数目的降低。如果 Anycast 地址集中在一定的范围,则路由器的路由表的路由项目就可以聚集成为少数的几项。同时单播路由表和 Anycast 路由表混合在一起这样路由表的空间就变得很大,路由的效率就大大地降低了。基于以上几点原因提出了以下对应的解决方案。

4.1 划分单独的地址空间

对于 Subnet Anycast 地址,因为它的使用是集中式的,对于以 Subnet Anycast 为目的地址的路由表的项可以和别的路由表项很好地聚集起来,所以对于 Subnet Anycast 类型的地址不需要特别的处理,即把 Subnet Anycast 地址当成单播地址即可。对于 Global Anycast 类型的地址为它分配如图 2 所示的地址段,在图中地址的前 n 位为子网前缀,后 $128-n$ 为 Anycast 主机 ID。在 IPv6 网络中规定 Global Anycast 地址必须为图 2 规定的地址段中,即 Anycast 地址为某个规定的 IPv6 单播地址段。



图 2 Anycast 地址分配图

Anycast 路由算法的实现原理如图 3^[7]所示,在图中把与自治系统的某个路由器的某个接口的网段规定为 Anycast 地址区域,这样在每个自治系统中,Anycast 主机的 IP 地址不是分散在整个区域中而是集中在指定的网段中,在这个规定的网段外的路由器中的路由表中的路由项可以由整个网段中的主机路由项聚合而成(聚合算法后面会描述),这里聚合的算法采用类似动态 CIDR 算法^[8],举例如下(为了方便拿 IPv4 地址举例):有两个主机 A、B 它们的 IP 地址分别为 192.168.63.0/32、192.168.63.1/32,因为这两个地址构成的网段是 192.168.63.1/31,并且此网段内只包含这两个地址,具体原理请参考图 4。

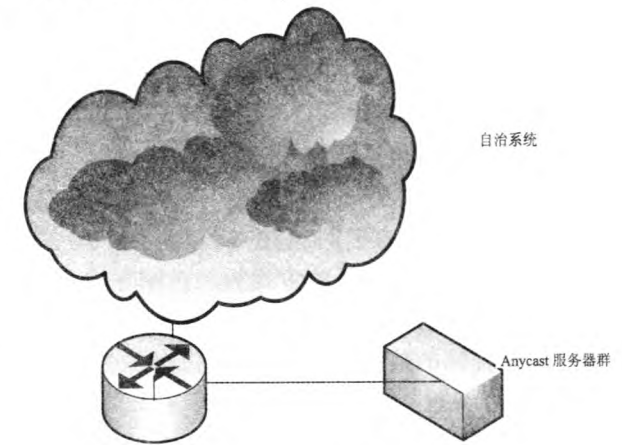


图 3 Anycast 实现原理图

```

11000000.10101000.00111111.00000000/32
    192.168.63.0/32

11000000.10101000.00111111.00000001/32
    192.168.63.1/32
    |
11000000.10101000.00111111.00000000/31
    192.168.63.0/31
    
```

图 4 Anycast 聚合示例

所以网段 192.168.63.1/31 和地址 192.168.63.0/32、192.168.63.1/32 是等价的。所以这两个地址可以聚合为 192.168.63.0/31, 这样路由器中路由表的路由项将会随着路由的聚合而变得数目更小。

4.2 单独 Anycast 路由表

把 Anycast 的路由表和 Unicast 的路由表分开, 这样在路由 Anycast 和 Unicast 时互相不影响。如果把 Anycast 和 Unicast 的路由表放在一块, 这时因为 Anycast 的路由表比较庞大, 会造成 Anycast 和 Unicast 公用的路由表规模很大, 这时在查找路由时速度很慢, 使得 Anycast 和 Unicast 的路由很慢; 如果把路由表分开, Unicast 的路由不受 Anycast 的路由的影响, 这样 Unicast 的路由可以高效的运作, 因为在网络中 Anycast 的路由量比 Unicast 的路由量小, 而且相对 Unicast 的路由对性能的需求较小, Anycast 的路由表可以比较大, 这时即使 Anycast 的路由性能比较低也是可以接受的。

4.3 地址归并算法

算法描述: 在描述算法之前引入一些相关知识, 图 5 是一棵满二叉树^[9], 这棵二叉树具有四层, 最下面的一层为叶子节点, 其余的为内节点或根节点。在这棵二叉树中, 假设根节点编号为 0, 在为内节点编号时每个节点的左子节点编号为其父节点的值加上 0 (如 B 节点的编号为 00)。同样每个节点的右子节点的编号为其父节点的值加上 1 (如 I 节点的编号为 01)。当这棵树是 128 层的满二叉树时, 可以将这棵树的叶子节点对应到整个 IPv6 网络中的计算机的 IP 地址, 而这棵树的内节点表示某个子网。如图 5 中的 C 节点表示一个具有两个地址的子网, 它的编号为 000 或者是 0000/3 或是 0001/3 (编号的方式为二进制方式)。内节点的编号方式为 i/n (其中 i 为一个地址数据, n 为一个类似子网掩码的数据, 表示地址 i 的前 n 位为一个子网编号)。可以把这棵树层数扩展到小于 128 (为了把这个地址空间映射到 IPv6 的地址空间) 的某个数。比如可以把这个值设为 100, 在这个地址的前面设定一

个长度为 28 位的固定的二进制常量即可构成一个 IPv6 地址空间。例如可设定这 28 位的常量为全 1 的地址, 剩下的地址构成了一个 IPv6 地址空间, 把这个空间定义为 Anycast 地址空间, 在互联网上 IP 数据报的地址为这个空间中的数值时, 认为这个数据报是采用的 Anycast 通信方式。算法用到的数据结构如下:

```

Struct node {
Node * next;
DWORD address; 地址空间
DWORD mask; 子网掩码
BOOL visited; 是否被使用过
};
    
```

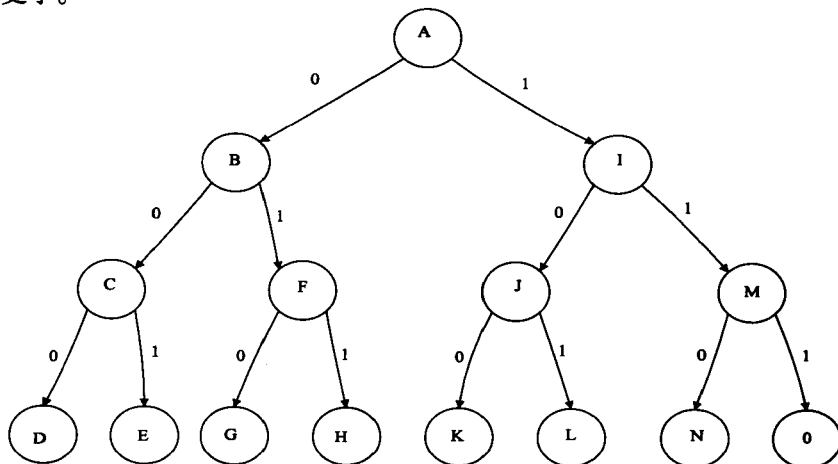


图 5 满二叉树

Anycast 地址聚合算法: 讲述本算法时举例使用的数据见图 5, 在图中我们可以用一个子网来表示多个 IP 地址方式来压缩数据量也即地址聚合, 在图中内节点 C 可以代表叶子节点 D、E, 同样 F 节点可以表示节点 G、H, 节点 B 可以表示为 C 和 F 节点, 即 C 和 F 可以聚合为 B 节点, 通过这种聚合叶子节点 D、E、G、H 可以表示为 B 节点, 这样数据量可以被大大地压缩。在图中如果说所有的叶节点都存在时, 这时的算法结果为最优, 即所有的节点最后聚合为一个根节点。最坏的情况是当叶节点全部为左叶节点或者是全部为右叶节点时, 这时 IP 的数据量为地址空间的数量的 1/2。

为了描述算法方便在实现算法时使用了 IPv4 而不是 IPv6, 下面是归并算法的伪代码。

```

node * nodes[32]; // 其中包含要聚合的节点, nodes[n] 是
nodes[n-1] 的父节点
node * temp, * p; // 两个临时节点指针
for(int i=0; i<32; i++)
{
Temp=nodes[i];
while(temp!=NULL)
{
P=temp->next;
while(p!=NULL)
{
    
```

```

If(p->visited == false)
{
    If(p 是 temp 的兄弟节点)
    {
        生成 p 和 temp 的父节点 parent;
        把 parent 节点插入到 nodes[i+1] 表示的链表中;
        把节点 p 和节点 temp 从 nodes[i] 中删除;
    }
    p = p->next;
}
Temp->visited = true;
temp = temp->next;
}

```

Anycast 地址拆分算法:讲述本算法时举例使用的数据见图 5,本算法为当 Anycast 路由器发现某个 Anycast 地址失效时要采取的动作,即把路由表中对应的路由项删除,当路由表中存在等于这个 Anycast 地址时可以简单地删除此地址即可,如果路由表中存在包含此地址的子网时,需要拆分这个子网,如要删除 G 节点这个时候路由表中存在 B 节点,需要把 B 节点拆分成 C 和 F 节点,然后把 F 节点拆分成 G 和 H 节点,接着把 G 节点删除即可。实现本算法的伪代码如下:

```

node * nodes[32]; //其中包含表示当前路由表的叶节点和网段的数据结构
node * left, * right, * parent; 三个临时的节点指针
node current; //包含要删除的节点
Int i=0; //nodes 数组的索引
for(int j=0; j<32; j++)
{
    if((nodes[j] 包含 current 节点) || (nodes[j] 包含 current 节点的父节点))
    {
        I=j;
        Break;
    }
    * parent = nodes[j];
    for(int j=I; j>0; j--)
    {
        拆分 * parent 为 * left 节点和 * right 节点;
    }
}

```

```

if(* left 是 current 节点的父节点)
{
    把 * right 插入到 nodes[j-1] 链表中;
    * parent = * left;
} else
{
    把 * left 插入到 nodes[j-1] 链表中
    * parent = * right;
}
把 current 节点从 nodes[0] 链表中删除

```

5 结束语

文中给出了 IPv6 中新的服务 Anycast,讲解了当前 Anycast 技术实现过程中的一些问题,并对 Anycast 遇到的关键问题给出了相应的解决办法,如对于 Anycast 的规模性问题,文中给出三种方法联合起来解决。文中给出的划分单独的地址空间、单独 Anycast 路由表、地址归并三种方法的联合使用能够大大地解决 Anycast 路由的性能问题,同时又不给 Unicast 和 Multicast 路由的性能带来负面影响。

参考文献:

- [1] Partridge C, Mendez T, Milliken W. Host Anycasting Service[S]. RFC 1546, 1993.
- [2] Hinden R M, Deering S E. IP Version6 Addressing Architecture[S]. RFC 2374, 1998.
- [3] 张华健,唐家益. Anycast 通信模型的研究[J]. 计算机应用与软件, 2004, 21(3): 105-107.
- [4] 陈涛,陈启买. 分布式计算机系统负载均衡研究[J]. 计算机技术与发展, 2006, 16(5): 33-35.
- [5] Johnson D. Reserved IPv6 Subnet Anycast Addresses[S]. RFC 2526, 1999.
- [6] Hinden R, Deering N S. IP Version 6 Addressing Architecture[S]. RFC 4291, 2006.
- [7] 伍孝金. IPv6 技术与应用[M]. 北京:清华大学出版社, 2010.
- [8] Stevens W R. TCP/IP Illustrated Volume1: The Protocols[M]. Beijing: China Machine Press, 2002.
- [9] 严蔚敏,吴伟民. 数据结构[M]. 北京:清华大学出版社, 1997.

(上接第 119 页)

- friendship[M]//Recommender Systems for the Social Web Intelligent Systems Reference Library. [s. l.]: Springer, 2012: 159-175.
- [13] Kai Zhou. Combining Item Rating Similarity and Item Classification Similarity for Better Recommendation Quality[C]//Advanced Materials Research. Switzerland: [s. n.], 2012: 289

-292.

- [14] 李克潮,梁正友. 适应用户兴趣变化的指数遗忘协同过滤算法[J]. 计算机工程与应用, 2011, 37(6): 226-243.
- [15] 苏萌,柏林森,周涛. 个性化:商业的未来[M]. 北京:机械工业出版社, 2012.

具有规模性的Anycast算法实现


作者：

张华健， 杨健， [ZHANG Hua-jian](#), [YANG Jian](#)

作者单位：

[南京邮电大学物联网学院, 江苏南京, 210003](#)

刊名：

[计算机技术与发展](#) 

英文刊名：

[Computer Technology and Development](#)

年，卷(期)：

2013, 23(7)

本文链接：http://d.wanfangdata.com.cn/Periodical_wjfz201307030.aspx