网络工具（一）

本文将简单介绍几个网络工具及基本概念，内容比较浅显，有些命令需要到实际工作环境中才能发挥其作用。这篇文章会提到三个网络命令:ping，nslookup和dig。这是网络工具中非常常用的。

先来说ping，ping的最主要作用是检测网络连通性，ping程序的客户机（发送者）通过ICMP协议包发送到ping服务器（接收并回显者），然后通过得到的应到来判断网络连通性以及链间往返时间RTT。被ping的服务器与我们通常所知的服务器并不完全相同，虽然它的功能也是对客户请求进行应答，但这里ping服务器本身是内核中对TCP/IP协议的实现，所以处理过程是发生在内核态下的。

当我们输入下ping以及被ping的地址之后，客户机发送类型码为0的ICMP回显应答数据包到ping服务器，用户机数据包在发送过程中在TCP/IP协议栈中（自顶向下）分别用到了网络层、数据链路层和物理层的服务。这里有一点必须明确的是ICMP协议并没有使用到传输层的TCP或UDP协议，仅仅使用了IP层的服务。根据我们通常的认知模型来看，ICMP应该是介于网络层和传输层之间的一层协议，但是因为在划分的时候并没有在二者这间划分一个子层，所以大部分情况下我们还是把ICMP协议划分到网络层中，只不过它的位置稍稍比IP协议偏上一些。

当ping服务器收到发来的ICMP数据包之后会检查接收到的数据类型，然后校验传输数据是否出错。确认无误后ping服务器发送类型码为8的请求回显数据包传送到我们的客户机。这里对于ping的具体过程不再叙述，以后会出一篇文章专门介绍如何自己实现ping程序。

接下来我们介绍ping的相关命令。有关ping的使用方式，我们在命令行下输入man ping不仅可以得到ping的基本语法、参数使用及功能，还能得到关于ping的基本信息。

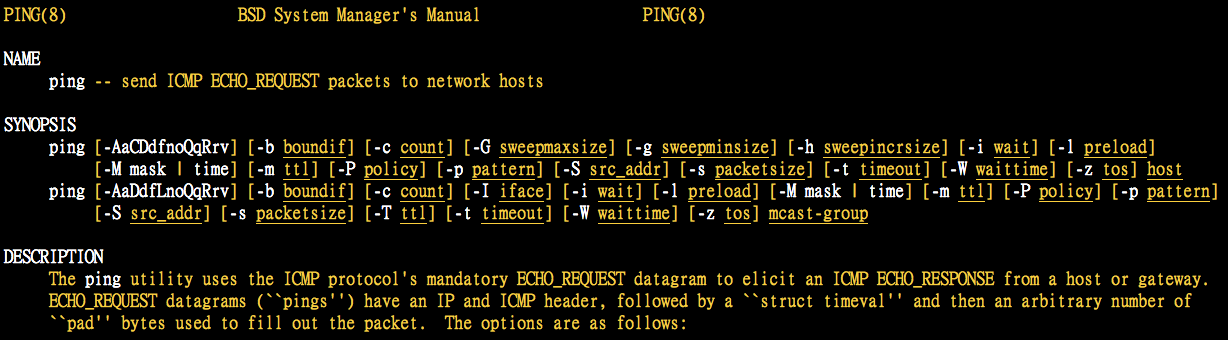


图1：ping基本信息

最常用用法就是不加任何参数去ping某个网站，如下：ping baidu.c

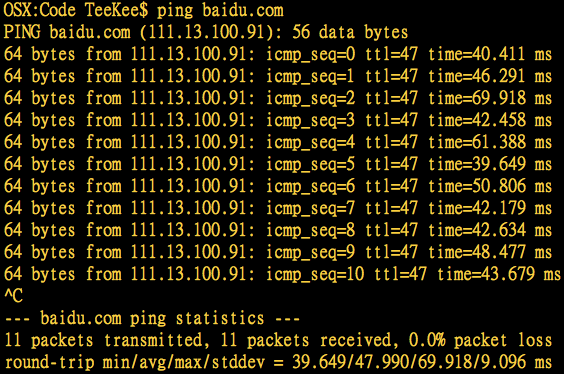


图2：ping结果

这里我们需要注意一些反馈回来的信息。首先我们发送了数据量为56Bytes的ICMP数据包到地址为111.13.100.91的百度服务器（这个地址会因条件不同二不同，不在本文讨论范围）。之后得到了64Bytes的回馈。除此之外，每个回馈都有唯一的icmp\_seq来对每一条请求进行回复，这里ttl表示的是经过的路由器跳数。在ping程序中，通过在ICMP报文数据中存放发送请求的时间值来计算往返时间，当应答返回时，用当前时间减去ICMP报文中的时间值，即是往返时间（很显然，这里设置报文中时间值的也是客户机本身）。倒数第二行返回的是整体统计，包括总共发送了多少包，收到了多少，丢失了多少，丢失率大小。最下面一行返回的是ICMP包最大、最小、平均RTT时间，至于stddev（linux下是mdev）表示的是ICMP包的RTT偏离平均值的程度。

通过以上信息，我们可以得到：如果有返回，只要不是显示超时都说明网络是连通的。通过stddev（mdev）和丢包率可以判断网络的网络的畅通程度，通过最大、最小、平均RTT值可以初步判断服务器和客户机的距离。

有几个常用值可以作为参考：ping本机或本机房的电脑通常RTT < 0.1ms，同城在1ms左右，不同城但同属南/北方大致在20ms，如果是50ms左右大致是南方（北方）ping北方（南方）的服务器，超过200ms的基本就架设仔国外的服务器了。

参数比较多，这里就不再截图了，只列举几个：

-c 控制发送的ICMP数据包数量

-q 只显示随后两行丢包、RTT信息

-s 设置数据包大小

-i 设置发送间隔（默认1s，修改需切换到root）

-f 潮水模式，快速ping

简单介绍了ping的使用方法，接下来我们需要介绍的是一个关于DNS的命令：nslookup。上面ping的例子其实已经出现了DNS的影子，在图2中，我们ping的是baidu.com，最终得到的结果是111.13.100.91，这里已经完成了从域名到IP地址的转换，而这一过程正是接下来需要讨论的。

关于域名解析的基本概念这里就不在仔细说了，我将会结合nslookup命令解释DNS的过程。

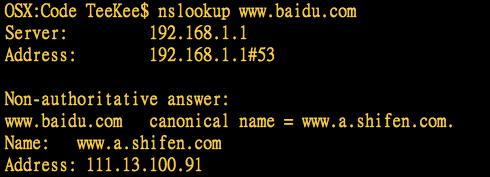


图3：nslookup

首先得到看到的是域名为192.168.1.1的Server行，这里指的是我们请求DNS时候辅助我们完成域名解析过程的DNS服务器，第二行中的53代表端口号，正好也说明我们仔进行DNS请求时是先将域名解析请求发到该服务器上的，这个地址本身是一个NAT地址，所以没有太多意义。

DNS通常有两种基本模式，递归查询和迭代查询。

在解释二者区别之前，首先必须要明确的一点是，无论通过什么方式的DNS查询，最终反馈IP地址给客户机的一定是这里地址为192.168.1.1的服务器。但是这个从域名到IP地址映射的数据并不是存储在这个服务器里的。试想如果DNS服务器都要存储下所有主机的域名和IP地址的话，那这个数据量也太可怕了。所以DNS实际上是最典型的分布式数据库系统，我们所需要的数据最终通过多台计算机的协同作业完成整个解析过程。

比如学院网站：<http://sdcs.sysu.edu.cn/> ，我们在解析这个网站（假设递归解析）时，首先将请求由客户机发送到刚刚提到的那台本地DNS服务器，DNS服务器得到请求后将会向根域名服务器（root-servers.net）发送请求，根域名服务器请求.cn的DNS服务器的地址（顶级域名服务器），接下来.cn的DNS服务器又会去请求.edu.cn服务器的DNS服务器，一直通过这种方式直到找到保存有sdcs.sysu.edu.cn的IP地址的DNS服务器（权威域名服务器）。当取得IP地址之后，此服务器又会向发起请求的.sysu.edu.cn（上一级）服务器发回反馈，然后按照之前请求相反的路径将数据发送到根域名服务器，之后再发给最先接收客户请求的DNS服务器（192.168.1.1），并由其返回给客户机。

迭代请求方式在本地DNS服务器接收到请求之后，第一步依旧是向根DNS服务器发起请求，但是这次根DNS服务器不需要再向之前一样给.cn服务器发起请求并等待最终回馈了，而是直接将.cn服务器的地址告知DNS服务器，再由DNS服务器向.edu.cn的DNS服务器发起请求，一直迭代上面的执行过程直到sdcs.sysu.edu.cn服务器返回IP地址给我们的DNS服务器为止，最终由其将结果发回客户机。

考虑到全球仅有13台根域名服务器，如果每次DNS请求都通过根域名服务器来响应最终IP地址（递归式）的话，那么根域名服务器将会负载及其严重，为了解决这个问题，我们有两种方法来解决。

一是通过将递归查询和迭代查询结合的方式，本地DNS服务器和根域名服务器之间通过迭代的方式，每次根域名服务器在接受到本地DNS服务器发来的请求之后都只返回顶级域名服务器的地址（通常是.cn、.com、.net之类），之后的操作通过递归的方式来解决。另一个最有效的方法是设置本地DNS缓存，缓存模型就是在本地保存下最常使用的域名和IP地址的映射，当客户机发起DNS请求时，先查找本地DNS服务其中是否已经缓存了其中的数据，如果存在则无需再去请求根域名服务器，这样不仅可以大大减轻其他DNS服务器的负载也能提高DNS的响应速度。

不过设置DNS缓存也带来了一个问题就是，如果我们所查询的站点在缓存有效期内迁移了，而本地DNS的缓存依旧会返回旧的数据，这样我们就会被定位到错误的IP地址产生错误。

我们在图三中就可以看出这里返回的IP地址实际上是缓存的数据而不是实时通过上面提到的递归或者迭代的方式完成的DNS请求。确定是不是缓存数据的方法很简单，最下面一行有提示Non-authoritative answer就能说明本地DNS服务器返回的是缓存数据。

实际上，DNS服务器返回给客户机的资源不仅仅只有IP地址，其返回的资源实际是一个五元组{域名，生存期限，类别，类型，值}，其中域名顾名思义就是我们所需要查询的域名，生存期限和DNS缓存相关，是该记录在DNS服务器中的生存时间，类别没有太多变化通常字段都是IN表示internet。类型字段有八种，分别是SOA（授权起始）、A（IP地址）、MX（邮件交换）、NS（域名服务器）、CNAME（别名）、PTR（指针）、HINFO（主机描述信息）、TXT（文本信息），而五元组中最后一项的值和前一部分的类型相关，当然了，比较常见的还是A、MX、CNAME。

我将简单展示如何显示各类型的信息，因为反馈的信息会有部分重合，这里八种类型只展示SOA、MX的用法并简单分析其返回信息。

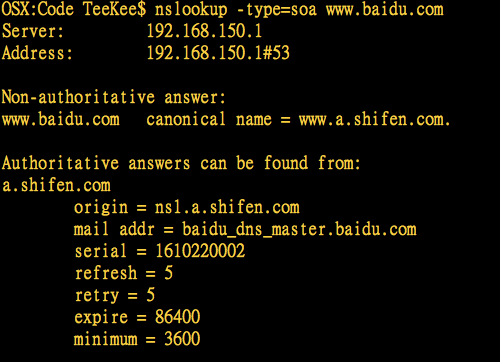


图4：SOA类型显示信息

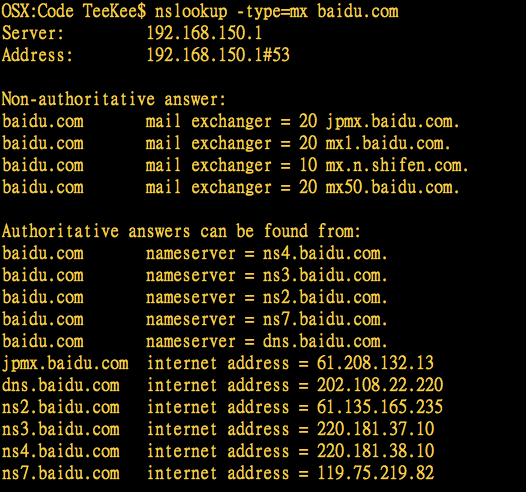


图5：MX类型显示信息

图4显示了[www.baidu.com](http://www.baidu.com)的别名[www.a.shifen.com](http://www.a.shifen.com)。mail addr表示管理员邮箱地址，serial表示版本号次，minimum则表示缓存的时效。

图5显示在MX类型下baidu的四个邮箱服务器的名字，并且给出了baidu五个权威服务器的名字及对应的IP地址。

另外，nslookup还有一种交互模式，我们可以在交互模式下完成一些基本设置。直接键入nslookup就会进入交互模式，在交互模式下我们可以通过输入server + DNS服务器地址来改变我们计算机本地DNS服务器地址。也可以通过键入set all来查看当前DNS的基本配置。

相比于nslookup的功能，命令dig提供的功能更加强大。当输入dig .的时候，会显示根域名服务器的A记录。如果我们不去指定特定的DNS服务器，那么dig命令将会依据我们/etc/resolv.conf配置文件中设置的DNS服务器作为本地DNS服务器，然后对我们设置需要查询的结果进行查询。

我们也可以手动更改本地DNS服务器来为我们服务，比如使用google的8.8.8.8用来作为对baidu.com进行解析的DNS服务器。如下：

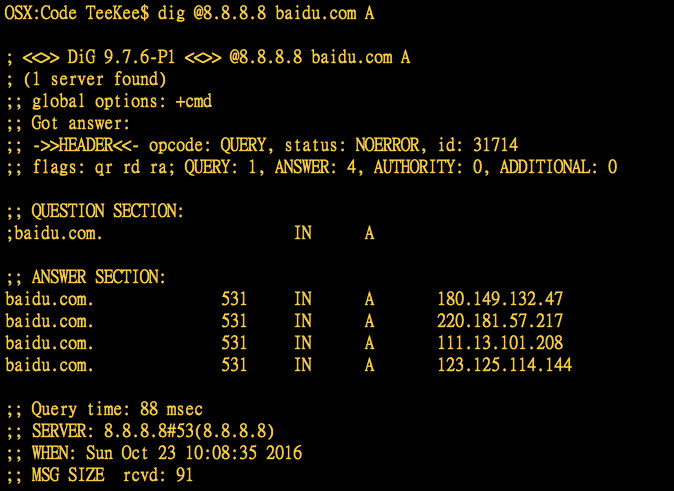
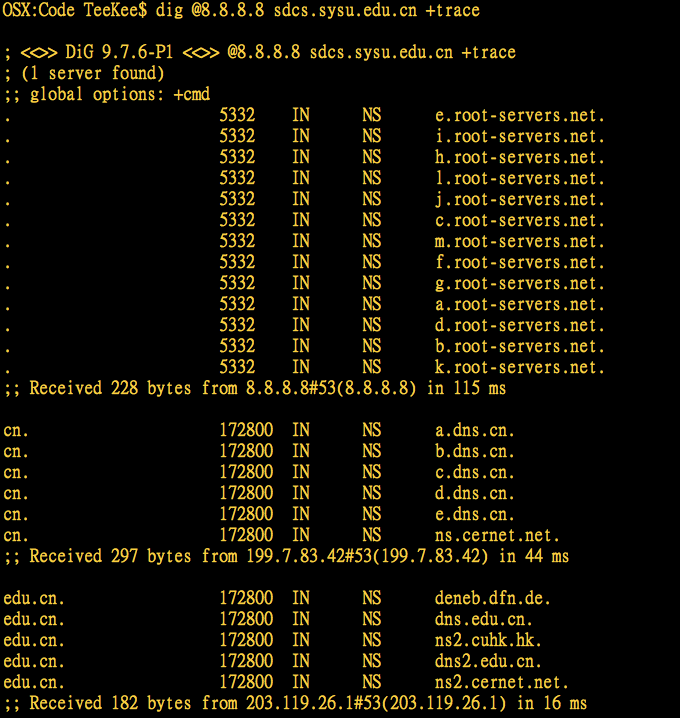


图6：dig更换谷歌dns服务器服务

这次使用的不再是我们/etc/resolv.conf配置中指定的DNS服务器了，而是使用谷歌的开放服务器8.8.8.8提供的服务，从反馈的结果来看，baidu.com被解析后的IP地址与之前通过nslookup得到的并不相同，可见百度在域名解析这一层做了负载均衡，使得大家在访问其网站时可以被解析到不同的主机。

所有查询选项中，对于dig来说最重要的就是+trace选项了。



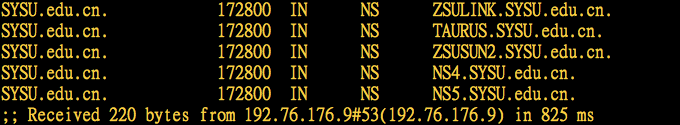


图7：dig +trace示例

第一步首先从本地DNS服务器查找到根域名服务器列表，之后选择了IP地址为199.7.83.44的根域名服务器来查找cn.域的DNS列表，之后选择了203.119.26.1（列出的13个根域名服务器之一）为IP的顶级域名服务器来查找edu.cn.域的DNS列表，接下来选择了IP地址为203.119.26.1的二级域名服务器来查找sysu.edu.cn.域的DNS列表。最后，将会由IP地址为192.76.176.9的权威域名服务器（包含完整的sdcs.sysu.edu.cn到IP地址的映射）来返回最终的查询结果。该结果由本地域名服务器（8.8.8.8）返回给客户机，完成整个DNS请求。

更多的使用方法可以通过man + 命令名的方式来查询，这里更多的会讲述其中的一些原理性知识。