**DNS协议解析**

DNS 的来由   
如果您为您的机器设定过 internet 连线﹐那么您一定接触过 DNS 了﹐但 DNS 又是什么东东呢﹖说穿了﹐DNS 是用来帮助记忆网路位址的﹐完全是为了迁就人类的记忆思维而设的。   
  
DNS 的全称是 Domain Name System(或 Service) ﹐当您连上一个网址﹐在URL打上﹕www.google.com 的时候﹐可以说就是使用了 DNS 的服务了。但如果您知道这个 www.google.com 的 IP 位址﹐直接输入 216.239.53.101 也同样可以到达这个网址。其实﹐电脑使用的只是 IP 位址而已(最终也是 0 和 1 啦)﹐这个 www.google.com 只是让人们容易记忆而设的。因为我们人类﹐对一些比较有意义的文字记忆(如﹕www.google.com)﹐比记忆那些毫无头绪的号码(如﹕216.239.53.101)﹐往往容易得多。DNS 的作用就是为我们在文字和 IP 之间担当了翻译﹐而免除了强记号码的痛苦。   
假如您的电话有名字记忆功能﹐您只需知道对方的名字﹐就可以拨号给友人了﹐我们可以说﹐这电话也具备如 DNS 的功能了呢﹗但是﹐我们在网路中使用的 DNS 系统﹐就是这么简单吗﹖非也﹐复杂得很呢﹗下面﹐就让我们一起去探索一下 DNS 的奥秘﹕   
在早期的 IP 网路世界里面﹐每台电脑都只用 IP 位址来表示﹐不久人们就发现这样很难记忆﹐于是﹐一些 UNIX 的管理者﹐就建立一个 HOSTS 对应表﹐将 IP 和主机名字对应起来﹐这样﹐用户只需输入电脑名字﹐就可以代替 IP 来进行沟通了。如果你安装了 Linux 系统﹐在 /etc 下面就可以找到这个 hosts 档案﹔在 NT 的系统里﹐你也可以在 \winnt\system32\drivers\etc 下面找到它。不过这个 HOSTS 档是要由管理者手工维护的﹐最大的问题是无法适用于大型网路﹐而且更新也是件非常头痛的事情。这就是 DNS 大派用场的时候了。   
DNS 的结构   
DNS 是一个分层级的分散式名称对应系统﹐有点像电脑的目录树结构﹕在最顶端的是一个“root”﹐然后其下分为好几个基本类别名称﹐如﹕com﹑org﹑edu 等﹔再下面是组织名称﹐如﹕ibm﹑microsoft﹑intel 等﹔继而是主机名称﹐如﹕www﹑mail﹑ftp 等。因为当初 internet 是从美国发展起的﹐所以当时并没有国域名称﹐但随着后来 internet 的蓬勃发展﹐DNS 也加进了诸如 tw﹑hk﹑cn 等国域名称。所以一个完整的 dns 名称就好象是这样的﹕www.xyz.com.tw﹐而整个名称对应的就是一个(或多个) IP 位址了。   
在早期的设计下﹐root 下面只有六个组织类别﹕   
不过﹐自从组织类别名称开放以后﹐各种各样五花八门的名称也相继涌现出来了﹐但无论如何﹐取名的规则最好尽量适合网站性质。除了原来的类别资料由美国本土的 NIC(Network Information Center) 管理之外﹐其它在国域以下的类别分别由该国的 NIC 管理(比方说台湾的 DNS 将授权给 twnic 来管理)。这样的结构看起来就像这样﹕   
在结构中﹐各组织的 DNS 经过申请后由该组织或其委托主机管理(通常当您申请注册一个 domain 域名称的时候﹐都要指定两台 DNS 主机负责该域名的 DNS 管理)。   
DNS 的运作   
在我们设定 IP 网路环境的时候﹐都要告诉每台主机关于 DNS 伺服器的位址(我们可以手动的在每一台主机上面设置﹐也可以使用 DHCP 来指定)。但这设定的义意何在呢？从前面的介绍我们或可知道：其目的就是请 DNS 帮忙解析主机名称与 IP 位址啦。在这个设定过程中，DNS 被称为 resolver (也就是负责解析的 DNS Server)，而被设定主机，则只是单纯的 DNS Client 了，也就是提出解析请求的主机。   
下面让我们看看 DNS 是怎样运作的﹕   
1. 当被询问到有关本域名之内的主机名称的时候﹐DNS 伺服器会直接做出回答﹔   
2. 客户端向伺服器提出查询项目﹔   
3. 当被询问到有关本域名之内的主机名称的时候﹐DNS 伺服器会直接做出回答﹔   
4. 如果所查询的主机名称属于其它域名的话﹐会检查快取记忆体(Cache)﹐看看有没有相关资料﹔   
5. 如果没有发现﹐则会转向 root 伺服器查询﹔   
6. 然后 root 伺服器会将该域名之下一层授权(authoritative)伺服器的位址告知(可能会超过一台)﹔   
7. 本地伺服器然后会向其中的一台伺服器查询﹐并将这些伺服器名单存到记忆体中﹐以备将来之需(省却再向 root 查询的步骤)﹔   
8. 远方伺服器回应查询﹔   
9. 若该回应并非最后一层的答案，则继续往下一层查询，直到获的客户端所查询的结果为止﹔   
10. 将查询结果回应给客户端﹐并同时将结果储存一个备份在自己的快取记忆里面﹔   
11. 如果在存放时间尚未过时之前再接到相同的查询﹐则以存放于快取记忆里面的资料来做回应。   
从这个过程我们可以看出﹐没有任何一台 DNS 主机会包含所有域名的 DNS 资料﹐资料都是分散在全部的 DNS 伺服器中﹐而 NIC 只需知道各 DNS 伺服器位址就可以了。 

网络基础 02 DNS 协议   
一、域名和资源记录的定义   
1、Name space definitions   
2、资源记录定义(RR definitions)   
    2.1 格式   
         后面分析报文的时候详细解释。   
    2.2 类型值(TYPE values)   
         类型主要用在资源记录中，注意下面的值是QTYPE的一个子集。   
        类型           值和含义   
         A               1 a host address   
         NS              2 an authoritative name server   
         MD              3 a mail destination (Obsolete - use MX)   
         MF              4 a mail forwarder (Obsolete - use MX)   
         CNAME           5 the canonical name for an alias   
         SOA             6 marks the start of a zone of authority   
         MB              7 a mailbox domain name (EXPERIMENTAL)   
         MG              8 a mail group member (EXPERIMENTAL)   
         MR              9 a mail rename domain name (EXPERIMENTAL)   
         NULL            10 a null RR (EXPERIMENTAL)   
         WKS             11 a well known service description   
         PTR             12 a domain name pointer   
         HINFO           13 host information   
         MINFO           14 mailbox or mail list information   
         MX              15 mail exchange   
         TXT             16 text strings   
    2.3 查询类型(QTYPE values)   
         查询类型出现在问题字段中，查询类型是类型的一个超集，所有的类型都是可用的查询类型，其他查询类型如下：   
         AXFR            252 A request for a transfer of an entire zone   
         MAILB           253 A request for mailbox-related records (MB, MG or MR)   
         MAILA           254 A request for mail agent RRs (Obsolete - see MX)   
         \*               255 A request for all records   
    2.4 类(CLASS values)   
         IN              1 the Internet   
         CS              2 the CSNET class (Obsolete - used only for examples in some obsolete RFCs)   
         CH              3 the CHAOS class   
         HS              4 Hesiod [Dyer 87]   
    2.5 查询类(QCLASS values)   
         查询类是类的一个超集   
         \*               255 any class   
   3、Standard RRs   
     3.1 CNAME RDATA format   
    3.2 HINFO RDATA format   
    3.3 MB RDATA format (EXPERIMENTAL)   
    3.4 MD RDATA format (Obsolete)   
    3.5 MF RDATA format (Obsolete)   
    3.6 MG RDATA format (EXPERIMENTAL)   
    3.7 MINFO RDATA format (EXPERIMENTAL)   
    3.8 MR RDATA format (EXPERIMENTAL)   
    3.9 MX RDATA format   
    3.10 NULL RDATA format (EXPERIMENTAL)   
    3.11 NS RDATA format   
    3.12 PTR RDATA format   
    3.13 SOA RDATA format   
    3.14 TXT RDATA format   
   4、ARPA Internet specific RRs   
    4.1 A RDATA format   
    4.2 WKS RDATA format   
5、IN-ADDR.ARPA domain   
6、Defining new types, classes, and special namespaces 

二、报文   
1、报文格式(Format)   
    dns请求和应答都是用相同的报文格式，分成5个段（有的报文段在不同的情况下可能为空），如下：   
    +---------------------+   
    |        Header       | 报文头   
    +---------------------+   
    |       Question      | 查询的问题   
    +---------------------+   
    |        Answer       | 应答   
    +---------------------+   
    |      Authority      | 授权应答   
    +---------------------+   
    |      Additional     | 附加信息   
    +---------------------+   
    Header段是必须存在的，它定义了报文是请求还是应答，也定义了其他段是否需要存在，以及是标准查询还是其他。   
    Question段描述了查询的问题，包括查询类型(QTYPE)，查询类(QCLASS)，以及查询的域名(QNAME)。剩下的3个段包含相同的格式:一系列可能为空的资源记录(RRs)。Answer段包含回答问题的RRs；授权段包含授权域名服务器的RRs；附加段包含和请求相关的，但是不是必须回答的RRs。   
    1.1 Header的格式   
        报文头包含如下字段：   
                                    1 1 1 1 1 1   
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |                      ID                       |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |QR|   Opcode |AA|TC|RD|RA|   Z    |   RCODE   |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |                    QDCOUNT                    |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |                    ANCOUNT                    |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |                    NSCOUNT                    |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |                    ARCOUNT                    |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
        各字段分别解释如下：   
        ID      请求客户端设置的16位标示，服务器给出应答的时候会带相同的标示字段回来，这样请求客户端就可以区分不同的请求应答了。   
        QR      1个比特位用来区分是请求（0）还是应答（1）。   
        OPCODE 4个比特位用来设置查询的种类，应答的时候会带相同值，可用的值如下：   
                0               标准查询 (QUERY)   
                1               反向查询 (IQUERY)   
                2               服务器状态查询 (STATUS)   
                3-15            保留值，暂时未使用   
        AA      授权应答(Authoritative Answer) - 这个比特位在应答的时候才有意义，指出给出应答的服务器是查询域名的授权解析服务器。   
                注意因为别名的存在，应答可能存在多个主域名，这个AA位对应请求名，或者应答中的第一个主域名。  
        TC      截断(TrunCation) - 用来指出报文比允许的长度还要长，导致被截断。   
        RD      期望递归(Recursion Desired) - 这个比特位被请求设置，应答的时候使用的相同的值返回。如果设置了RD，就建议域名服务器进行递归解析，递归查询的支持是可选的。   
        RA      支持递归(Recursion Available) - 这个比特位在应答中设置或取消，用来代表服务器是否支持递归查询。   
        Z       保留值，暂时未使用。在所有的请求和应答报文中必须置为0。   
        RCODE   应答码(Response code) - 这4个比特位在应答报文中设置，代表的含义如下：   
                0               没有错误。   
                1               报文格式错误(Format error) - 服务器不能理解请求的报文。   
                2               服务器失败(Server failure) - 因为服务器的原因导致没办法处理这个请求。   
                3               名字错误(Name Error) - 只有对授权域名解析服务器有意义，指出解析的域名不存在。   
                4               没有实现(Not Implemented) - 域名服务器不支持查询类型。   
                5               拒绝(Refused) - 服务器由于设置的策略拒绝给出应答。比如，服务器不希望对某些请求者给出应答，或者服务器不希望进行某些操作（比如区域传送zone transfer）。   
                6-15            保留值，暂时未使用。   
        QDCOUNT 无符号16位整数表示报文请求段中的问题记录数。   
        ANCOUNT 无符号16位整数表示报文回答段中的回答记录数。   
        NSCOUNT 无符号16位整数表示报文授权段中的授权记录数。   
        ARCOUNT 无符号16位整数表示报文附加段中的附加记录数。   
    1.2 Question的格式   
        在大多数查询中，Question段包含着问题(question)，比如，指定问什么。这个段包含QDCOUNT(usually 1)个问题，每个问题为下面的格式：   
                                    1 1 1 1 1 1   
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |                                               |   
    /                     QNAME                     /   
    /                                               /   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |                     QTYPE                     |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |                     QCLASS                    |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
        字段含义如下   
        QNAME   域名被编码为一些labels序列，每个labels包含一个字节表示后续字符串长度，以及这个字符串，以0长度和空字符串来表示域名结束。注意这个字段可能为奇数字节，不需要进行边界填充对齐。   
        QTYPE   2个字节表示查询类型，.取值可以为任何可用的类型值，以及通配码来表示所有的资源记录。   
        QCLASS 2个字节表示查询的协议类，比如，IN代表Internet。   
    1.3 资源记录格式(Resource record)   
        应答，授权，附加段都共用相同的格式：多个资源记录，资源记录的个数由报文头段中对应的几个数值确定，每个资源记录格式如下：   
                                    1 1 1 1 1 1   
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |                                               |   
    /                                               /   
    /                      NAME                     /   
    |                                               |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |                      TYPE                     |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |                     CLASS                     |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |                      TTL                      |   
    |                                               |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    |                   RDLENGTH                    |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--|   
    /                     RDATA                     /   
    /                                               /   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
        各字段含义如下：   
        NAME    资源记录包含的域名   
        TYPE    2个字节表示资源记录的类型，指出RDATA数据的含义   
        CLASS   2个字节表示RDATA的类   
        TTL     4字节无符号整数表示资源记录可以缓存的时间。0代表只能被传输，但是不能被缓存。   
        RDLENGTH        2个字节无符号整数表示RDATA的长度   
        RDATA   不定长字符串来表示记录，格式根TYPE和CLASS有关。比如，TYPE是A，CLASS 是 IN，那么RDATA就是一个4个字节的ARPA网络地址。   
    1.4 报文压缩   
        为了减小报文，域名系统使用一种压缩方法来消除报文中域名的重复。使用这种方法，后面重复出现的域名或者labels被替换为指向之前出现位置的指针。   
        指针占用2个字节，格式如下：   
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    | 1 1|                OFFSET                   |   
    +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
        前两个比特位都为1。因为lablels限制为不多于63个字节，所以label的前两位一定为0，这样就可以让指针与label进行区分。(10 和 01 组合保留，以便日后使用) 。偏移值(OFFSET)表示从报文开始的字节指针。偏移量为0表示ID字段的第一个字节。   
        压缩方法让报文中的域名成为：   
        - 以0结尾的labels序列   
        - 一个指针   
        - 指针结尾的labels序列   
        指针只能在域名不是特殊格式的时候使用，否则域名服务器或解析器需要知道资源记录的格式。目前还没有这种情况，但是以后可能会出现。   
        如果报文中的域名需要计算长度，并且使用了压缩算法，那么应该使用压缩后的长度，而不是压缩前的长度。   
        程序可以自由选择是否使用指针，虽然这回降低报文的容量，而且很容易产生截断。不过所有的程序都应该能够理解收到的报文中包含的指针。   
        比如，一个报文需要使用域名F.ISI.ARPA，FOO.F.ISI.ARPA，ARPA，以及根。忽略报文中的其他字段，应该编码为：   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    20 |           1           |           F           |   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    22 |           3           |           I           |   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    24 |           S           |           I           |   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    26 |           4           |           A           |   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    28 |           R           |           P           |   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    30 |           A           |           0           |   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    40 |           3           |           F           |   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    42 |           O           |           O           |   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    44 | 1 1|                20                       |   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    64 | 1 1|                26                       |   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
    92 |           0           |                       |   
       +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   
        偏移20的是域名F.ISI.ARPA。域名FOO.F.ISI.ARPA偏移40; 这样表示FOO的label后面跟着一个指向之前F.ISI.ARPA的指针。域名ARPA偏移64，使用一个指针指向F.ISI.ARPA的ARPA。注意可以用这个指针是因为ARPA是从偏移位置20开始的labels序列中的最后一个label。根域名在位置92定义为一个0，没有labels。   
2、传输(Transport)   
    DNS假设报文以数据报，或者从虚链路上以字节流进行传输。虚链路可以用来任何的DNS的传输，数据报可以减少代价提高传输性能。区域刷新必须使用虚链路，因为需要一个可靠的传输。   
    因特网中DNS支持端口53的TCP[RFC-793]和端口53的UDP [RFC-768]传输。   
    2.1 使用UDP   
        消息通过UDP的53端口进行传输。   
         UDP传输的消息严格要求限制在512字节内(不包括IP和UDP头)。长报文被截断，同时置报文头的TC标志位。   
         UDP不能用于区域传输，主要用在标准的域名查询。报文通过UDP可能会丢失，所以重传机制是需要的，请求和应答可能在网络中或者服务器处理的时候被重新排序，所以解析客户端不能依赖请求的发送顺序。   
        UDP的最优重传策略会因为网络的性能，客户的需要而不同，但是下面是推荐的：   
        - 客户端在对一台固定的服务器重试之前，尝试一下其他的服务器。   
        - 如果可能的话，重传的时间间隔需要建立在统计分析数据的基础上，太快的重试可能因为量太大导致服务器响应慢。建议的重试时间为2-5秒。   
    2.2 使用TCP   
         通过TCP发送的报文使用53端口，报文的前面有个字节表示后面报文的长度，长度不包括自己占用的2个字节，这个长度使得底层收取完整的报文后在交给上层处理。   
         很多连接管理策略如下:   
         - 服务器不能阻塞其他传输TCP数据的请求。   
         - 服务器需要支持多连接   
         - 服务器要等客户端主动关闭连接，除非所有的数据都已经传输完了。   
         - 如果服务器想关闭没有通讯的连接来释放资源，那么需要等待大约2分钟的时间。特别是要等SOA和AXFR(刷新操作中)在一个连接上传输完。服务器关闭连接的时候可以单方面的关闭，或者直接reset掉连接。   
三、实例   
1、请求解析www.baidu.com.   
     在linux下使用tcpdump port 53抓包，同时使用dig进行解析测试，得到结果如下：   
         ; (1 server found)   
         ;; global options: +cmd   
         ;; Got answer:   
         ;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1169   
         ;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 4, ADDITIONAL: 0   
         ;; QUESTION SECTION:   
         ;www.baidu.com.    IN A   
         ;; ANSWER SECTION:   
        www.baidu.com.   1200 IN CNAME www.a.shifen.com.   
        www.a.shifen.com. 600 IN A 121.14.88.76   
        www.a.shifen.com. 600 IN A 121.14.89.10   
         ;; AUTHORITY SECTION:   
         a.shifen.com.   86411 IN NS ns5.a.shifen.com.   
         a.shifen.com.   86411 IN NS ns6.a.shifen.com.   
         a.shifen.com.   86411 IN NS ns1.a.shifen.com.   
         a.shifen.com.   86411 IN NS ns3.a.shifen.com.   
    1.1 请求报文   
0x0000: 4500 003b f8cf 0000 4011 f9ae xxxx xxxx E..;....@......r   
0x0010:   xxxx xxxx 92b8 0035 0027 23ed 0491 0100 ...q...5.'#.....   
0x0020: 0001 0000 0000 0000 0377 7777 0562 6169 .........www.bai   
0x0030: 6475 0363 6f6d 0000 0100 01              du.com.....   
        0491：报文ID，也就是十进制的1169   
        0100：标志，置了RD字段，也就是期望递归的请求   
        0001 0000 0000 0000：分别为问题数，应答数，授权记录数，附加记录数，也就是1个问题   
        0377 7777 0562 6169 6475 0363 6f6d 00：也就是www.baidu.com的编码   
        00 0100 01：查询类型和查询类都为1，也就是internet的A记录查询   
    1.2 应答报文   
0x0000: 4500 00be 0016 4000 4011 b1e5 xxxx xxxx E.....@.@......q   
0x0010:   xxxx xxxx 0035 92b8 00aa 33e1 0491 8180 ...r.5....3.....   
0x0020: 0001 0003 0004 0000 0377 7777 0562 6169 .........www.bai   
0x0030: 6475 0363 6f6d 0000 0100 01c0 0c00 0500 du.com..........   
0x0040: 0100 0004 b000 0f03 7777 7701 6106 7368 ........www.a.sh   
0x0050: 6966 656e c016 c02b 0001 0001 0000 0258 ifen...+.......X   
0x0060: 0004 790e 584c c02b 0001 0001 0000 0258 ..y.XL.+.......X   
0x0070: 0004 790e 590a c02f 0002 0001 0001 518b ..y.Y../......Q.   
0x0080: 0006 036e 7335 c02f c02f 0002 0001 0001 ...ns5././......   
0x0090: 518b 0006 036e 7336 c02f c02f 0002 0001 Q....ns6././....   
0x00a0: 0001 518b 0006 036e 7331 c02f c02f 0002 ..Q....ns1././..   
0x00b0: 0001 0001 518b 0006 036e 7333 c02f       ....Q....ns3./   
         注意8180，也就是二进制的 1 0000 0 0 1 1 000 0000 ，说明是应答，置了RD和RA位   
         黄色背景为压缩编码，比如c016就代表第22个字节，也就是com。