网络工作组 山羊草
征求意见：1035 ISI
1987年11月

废弃：RFC 882,883,973

域名 - 实施和规范

1．本备忘录的状态

此RFC描述了域系统和协议的详细信息，并假定读者熟悉配套RFC"域名 - 概念和设施"[RFC-1034]中讨论的概念。

域系统是功能和数据类型的混合，它们是官方协议，功能和数据类型仍然是实验性的。 由于域系统是有意可扩展的，因此除了官方协议之外，系统的某些部分应始终预期新的数据类型和实验行为。
官方协议部分包括标准查询，响应和因特网RR类数据格式（例如，主机地址）。
自上一个RFC集以来，一些定义已经改变，因此一些先前的定义已经过时。

这些RFC中明确标记了实验或过时的功能，应谨慎使用此类信息。

特别提醒读者不要依赖于哪些价值观
在例子中出现的是当前或完整的，因为它们的目的主要是教学法。 本备忘录的发布 是
是无限的。

目录

2. 介绍

2.1. 概观

域名的目标是提供一种命名资源的机制，使名称可以在不同的主机，网络，协议族，互联网和管理组织中使用。

从用户的角度来看，域名作为本地代理的参数很有用，称为解析器，它检索与域名相关的信息。因此，用户可能要求提供与特定域名相关联的主机地址或邮件信息。要使用户能够请求特定类型的信息，请使用域名将适当的查询类型传递给解析程序。对于用户，域树是单个信息空间;解析器负责隐藏用户名称服务器之间的数据分布。

从解析器的角度来看，构成域空间的数据库分布在各种名称服务器中。域空间的不同部分存储在不同的名称服务器中，尽管特定数据项将冗余地存储在两个或多个名称服务器中。解析器首先了解至少一个名称服务器。当解析器处理用户查询时，它向已知的名称服务器询问该信息;作为回报，解析器要么接收所需信息，要么转介到另一个名称服务器。使用这些引用，解析器将学习其他名称服务器的身份和内容。解析器负责处理域空间的分配，并通过查询其他服务器中的冗余数据库来处理名称服务器故障的影响。

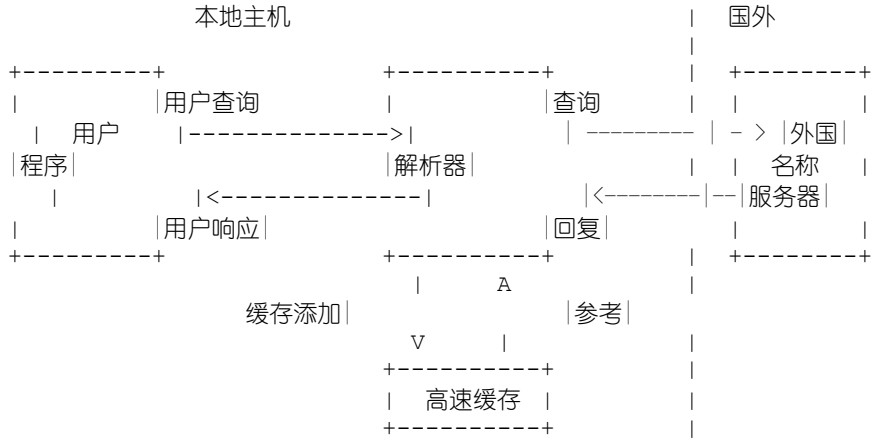
名称服务器管理两种数据。第一类数据集合称为区域;每个区域是域空间的特定“修剪”子树的完整数据库。这些数据称为权威数据。名称服务器会定期检查以确保其区域是最新的，如果不是，则获取更新区域的新副本

来自本地或其他名称服务器中存储的主文件。 第二种数据是由本地解析器获取的缓存数据。
此数据可能不完整，但在重复访问非本地数据时可提高检索过程的性能。 缓存数据最终会被超时机制丢弃。

此功能结构隔离了解析器中的用户界面，故障恢复和分发问题，并隔离了名称服务器中的数据库更新和刷新问题。

2.2. 常见配置

主机可以通过多种方式参与域名系统，具体取决于主机是运行从域系统检索信息的程序，是否应答来自其他主机的查询的名称服务器，或两种功能的各种组合。 最简单，也许是最典型的配置如下所示：

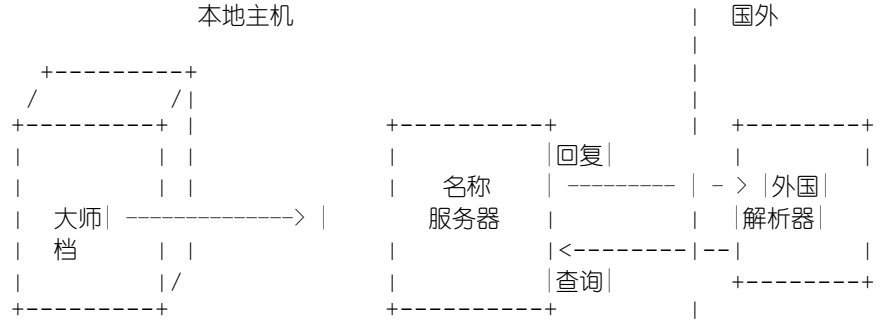


用户程序通过解析器与域名空间交互；用户查询和用户响应的格式特定于主机及其操作系统。
用户查询通常是操作系统调用，解析器及其缓存将是主机操作系统的一部分。
能力较弱的主机可以选择将解析器实现为与每个需要其服务的程序链接的子例程。
解析器通过对外部名称服务器和本地缓存的查询来回答用户查询所获得的信息。

注意，解析器可能必须对若干不同的外部名称服务器进行多次查询以回答特定的用户查询，因此用户查询的解析可能涉及若干网络访问和任意时间量。 对外部名称服务器的查询和相应的响应具有描述的标准格式

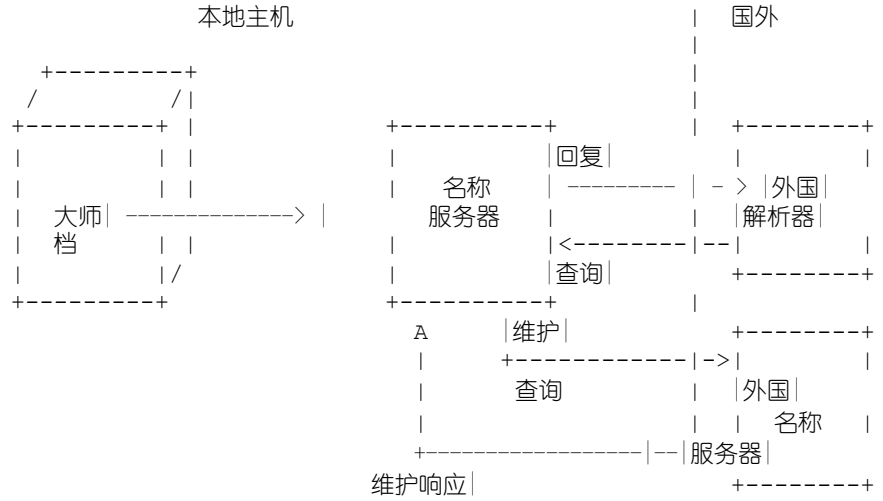
在这份备忘录中，可能是数据报。

根据其功能，名称服务器可以是专用机器上的独立程序，也可以是大型共享主机上的进程或进程。一个简单的配置可能是：



这里，主名称服务器通过从其本地文件系统读取主文件来获取有关一个或多个区域的信息，并回答有关从外部解析器到达的区域的查询。

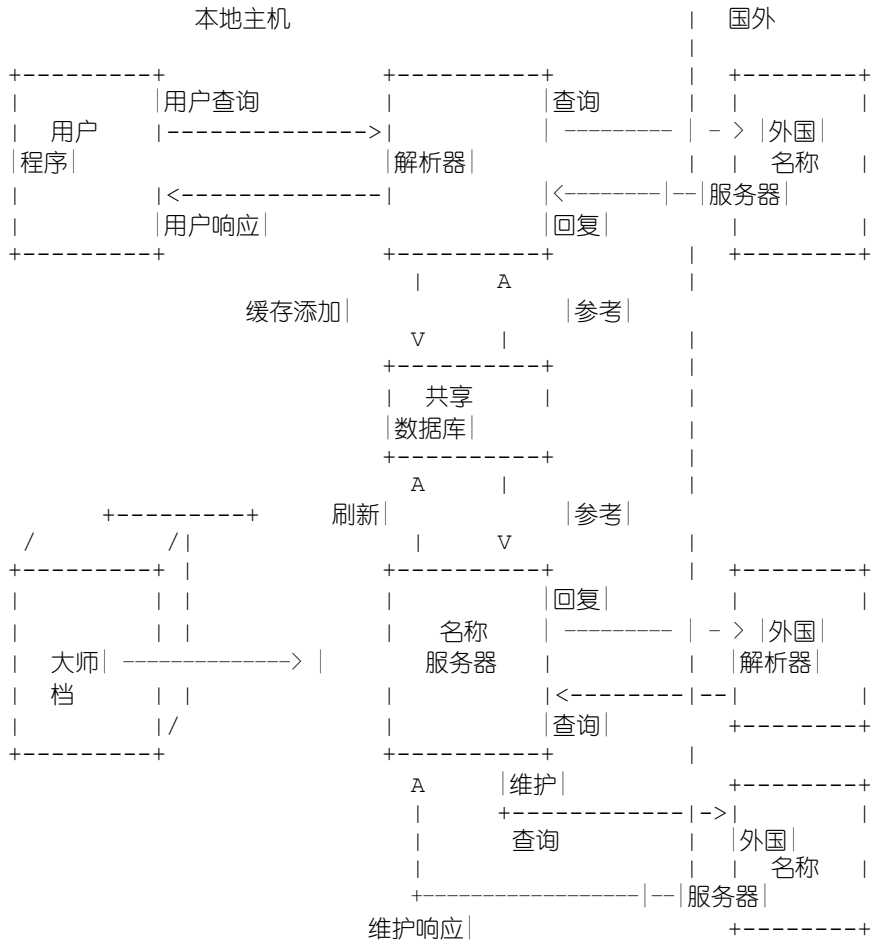
DNS要求所有区域由多个名称服务器冗余支持。 指定的辅助服务器可以使用DNS的区域传输协议获取区域并从主服务器检查更新。 此配置如下所示：



在此配置中，名称服务器定期向外部名称服务器建立虚拟电路以获取区域的副本或检查现有副本是否未更改。 发送的消息

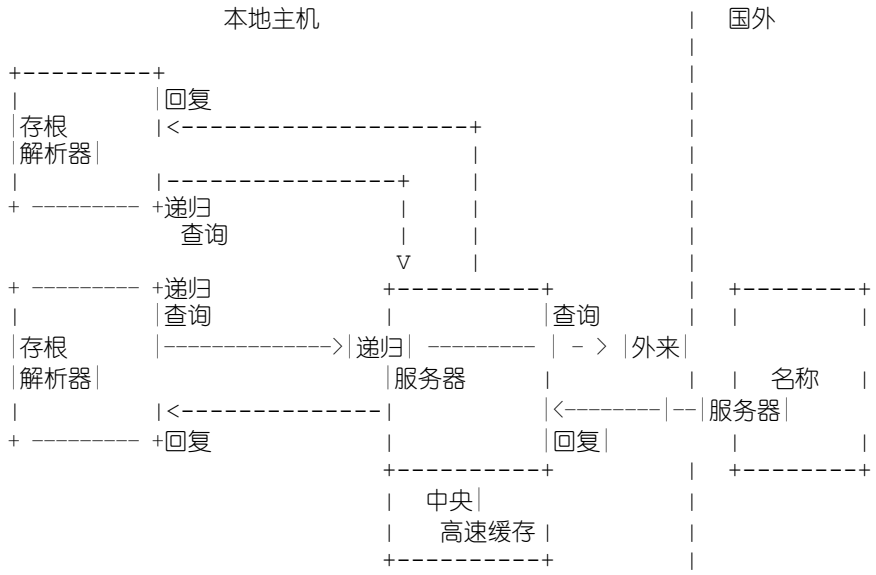
这些维护活动遵循与查询和响应相同的形式，但消息序列有些不同。

支持域名系统所有方面的主机中的信息流如下所示：



共享数据库保存本地名称服务器和解析程序的域空间数据。 共享数据库的内容通常是由名称服务器的定期刷新操作维护的权威数据和来自先前解析器请求的缓存数据的混合。 域数据的结构以及名称服务器和解析器之间同步的必要性意味着该数据库的一般特征，但实际格式取决于本地实现者。

还可以定制信息流，以便一组主机一起行动以优化活动。 有时这样做可以卸载功能较少的主机，这样他们就不必实现完整的解析器。
这适用于希望最小化所需新网络代码量的PC或主机。 该方案还可以允许一组主机可以共享少量高速缓存，而不是维护大量单独的高速缓存，前提是集中式高速缓存具有更高的命中率。 在任何一种情况下，解析器都被存储器解析器替换，存根解析器作为位于递归服务器中的解析器的前端，位于已知执行该服务的一个或多个名称服务器中：



在任何情况下，请注意尽可能复制域组件以确保可靠性。

2.3. 约定

域系统有几个约定处理低级但基本的问题。虽然实现者可以在他自己的系统中自由地违反这些约定，但他必须在从其他主机观察到的所有行为中遵守这些约定。

2.3.1. 首选名称语法

DNS规范尝试在构造域名的规则中尽可能通用。 我们的想法是，任何现有对象的名称都可以表示为具有最小更改的域名。

但是，在为对象分配域名时，谨慎用户将选择满足域系统规则和对象的任何现有规则的名称，无论这些规则是由现有程序发布还是隐含。

例如，在命名邮件域时，用户应满足此备忘录的规则和RFC-822中的规则。 创建新主机名时，应遵循HOSTS.TXT的旧规则。 这可以避免将旧软件转换为使用域名时出现的问题。

以下语法将导致许多使用域名的应用程序（例如，邮件，TELNET）的问题减少。

<域>: : =子域>" "

<子域>: : <subdomain>"。"<标签>

<标签>: : =字母> [[<LDHSTR>] <让掘进>

<LDH STR>: : 让我们挖HYP> < LDH STR>

让我们挖HYP>: : "-"

<让挖掘>: : =字母><数字>

<letter> :: =大写的52个字母A到Z中的任何一个和小写的a到z

<digit> :: =十位数0到9中的任何一个

请注意，虽然域名中允许使用大写和小写字母，但案例不附带任何重要性。 也就是说，具有相同拼写但不同情况的两个名称将被视为相同。

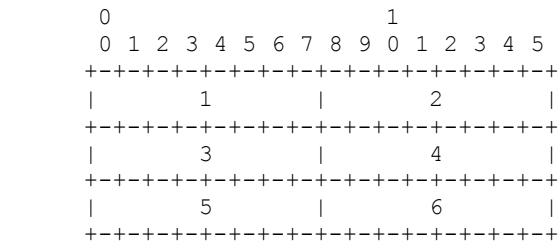
标签必须遵循ARPANET主机名的规则。 它们必须以字母开头，以字母或数字结尾，并且内部字符仅包含字母，数字和连字符。 长度也有一些限制。 标签不得超过63个字符。

例如，以下字符串标识Internet中的主机：A.ISI.EDU XX.LCS.MIT.EDU SRI-NIC.ARPA

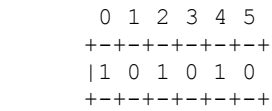
2.3.2. 数据传输顺序

本文档中描述的标题和数据的传输顺序被解析为八位字节级别。 每当图表显示一个

八位字节组，这些八位字节的传输顺序是用英语读取它们的正常顺序。例如，在下图中，八位字节按编号顺序传输。



每当八位字节表示数字量时，图中最左边的位是高位或最高位。 也就是说，标记为0的位是最高位。 例如，下图表示值170（十进制）。



类似地，每当多个八位字节字段表示数字量时，整个字段的最左位是最高有效位。 当发送多个八位字节数量时，首先发送最重要的八位字节。

2.3.3. 人物案例

对于作为官方协议一部分的DNS的所有部分，字符串（例如，标签，域名等）之间的所有比较都以不区分大小写的方式完成。 目前，该规则在整个域系统中毫无例外地生效。 但是，超出当前使用范围的未来添加可能需要在名称中使用完整的二进制八位字节功能，因此应避免尝试以7位ASCII存储域名或使用特殊字节来终止标签等。

当数据进入域系统时，应尽可能保留其原始案例。 在某些情况下，这是不可能做到的。 例如，如果两个RR存储在数据库中，一个在xy，一个在XY，它们实际上存储在数据库中的相同位置，因此只保留一个外壳。 基本规则是只有在数据用于定义数据库中的结构时才能丢弃大小写，并且在不区分大小写的情况下比较两个名称是相同的。

必须最小化案例敏感数据的丢失。因此，虽然xy和XY的数据都可以存储在单个位置xy或XY下，但数据可用于ax和BX永远不会存储在Ax，AX，bx或bX下通常，这会保留域名的第一个标签的情况，但会强制内部节点标签的标准化。

如果系统管理员的系统区分大小写，则将数据输入域数据库的系统管理员应注意以案例一致的方式表示他们提供给域系统的数据。域系统中的数据分发系统将确保保持一致的表示。

2.3.4. 尺寸限制

DNS中的各种对象和参数具有大小限制。它们列在下面。有些可以很容易地改变，有些则更为基础。

标签	63个八位组或更少
名	255个八位字节或更少
TTL	有符号32位数的正值。UDP消息512个八位字节或更少

3. 域名空间和rr定义

3.1. 名称空间定义

消息中的域名以标签序列表示。每个标签表示为一个八位字节长度字段，后跟该八位字节数。由于每个域名都以根的空标签结束，因此域名以零长度字节终止。每个长度八位字节的高位两位必须为零，并且长度字段的剩余六位将标签限制为63个八位字节或更少。

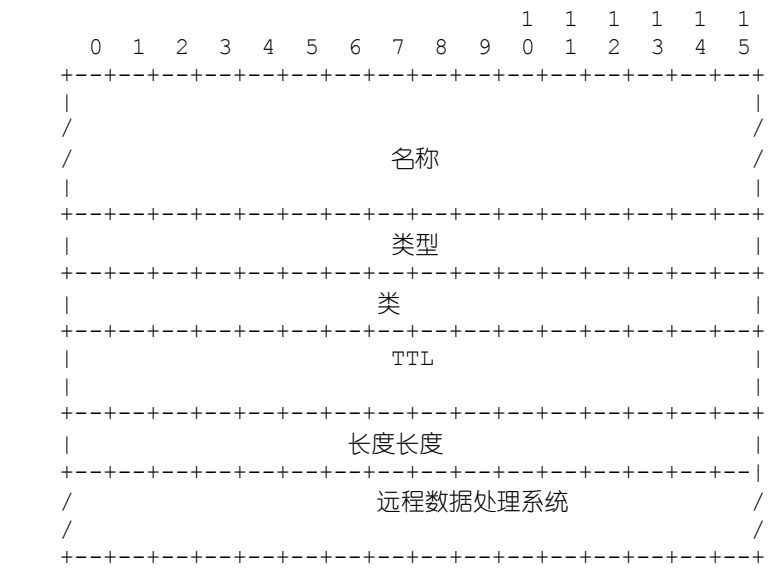
为了简化实现，域名的总长度（即，标签八位字节和标签长度八位字节）被限制为255个八位字节或更少。

虽然标签可以包含构成标签的八位字节中的任何8位值，但强烈建议标签遵循本备忘录中其他地方描述的首选语法，该语法与现有主机命名约定兼容。名称服务器和解析器必须以不区分大小写的方式比较标签（即A = a），假设ASCII为零奇偶校验。非字母代码必须完全匹配。

3.2. RR定义

3.2.1. 格式

所有RR都具有相同的顶级格式，如下所示：



哪里：

名称 所有者名称，即该资源记录所属的节点的名称。

类型 两个八位字节包含一个RR TYPE代码。类两个八位字节包含一个RR

CLASS代码。

TTL 一个32位有符号整数，指定在再次查询信息源之前可以缓存资源记录的时间间隔。 零值被解释为RR只能用于正在进行的事务，并且不应该被缓存。 例如，SOA记录总是以零TTL分布以禁止缓存。 零值也可用于极易变数据。

长度长度 无符号16位整数，指定RDATA字段的八位字节长度。

远程数据处理系统 一个可变长度的八位字节串，用于描述资源。 此信息的格式根据资源记录的TYPE和CLASS而有所不同。

3.2.2. TYPE值

TYPE字段用于资源记录。 请注意，这些类型是QTYPE的子集。

类型	价值和意义
A	1个主机地址
纳秒	2是权威名称服务器
分子动力学	3邮件目的地（已过时 - 使用MX）
中频	4邮件转发器（已废弃 - 使用MX）
cNeNe	5别名的规范名称
SOA	6标志着权威区域的开始
MB	7邮箱域名（实验）
毫克	8个邮件组成员（实验）
先生	9邮件重命名域名（实验）
空值	10无效RR（实验）
周	11众所周知的服务描述
PTR	12一个域名指针
海文	13主机信息
明复	14个邮箱或邮件列表信息
MX	15邮件交换
文本	16个文本字符串

3.2.3. QTYPE值

QTYPE字段显示在查询的问题部分中。 QTYPES是TYPE的超集，因此所有TYPE都是有效的QTYPE。 此外，还定义了以下QTYPE：

阿克斯弗尔 252要求转移整个区域

邮递员 253请求邮箱相关记录（MB，MG或MR）MAILA 254邮件代理RR的请求（已过时 - 请参阅MX）

* 255对所有记录的请求

3.2.4. CLASS值

CLASS字段显示在资源记录中。 定义了以下CLASS助记符和值：

在 1互联网

反恐精英 2 CSNET类（已废弃 - 仅用于某些过时RFC中的示例）

中国 3 CHAOS课程

房 协 4 Hesiod [代尔87]

3.2.5. QCLASS值

QCLASS字段显示在查询的问题部分中。 QCLASS值是CLASS值的超集;每个CLASS都是有效的QCLASS。 除CLASS值外，还定义了以下QCLASS：

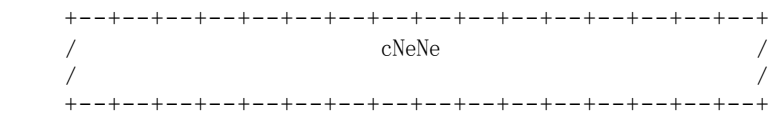
* 255任何类

3.3. 标准RR

预计在所有类中至少可能发生以下RR定义。特别是，NS，SOA，CNAME和PTR将在所有类中使用，并且在所有类中具有相同的格式。由于它们的RDATA格式是已知的，因此可以压缩这些RR的RDATA部分中的所有域名。

<domain-name>是表示为一系列标签的域名，以长度为零的标签终止。 <character-string>是单个长度的八位字节，后跟该字符数。 <character-string>被视为二进制信息，最长可达256个字符（包括长度八位字节）。

3.3.1. CNAME RDATA格式

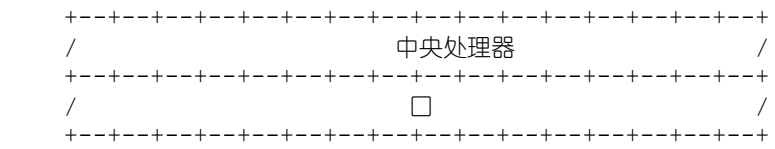


哪里:

cNeNe <domain-name>，指定所有者的规范或主要名称。所有者名称是别名。

CNAME RR不会导致其他部分处理，但在某些情况下，名称服务器可能会选择以规范名称重新启动查询。有关详细信息，请参阅[RFC-1034]中的名称服务器逻辑描述。

3.3.2. HINFO RDATA格式



哪里:

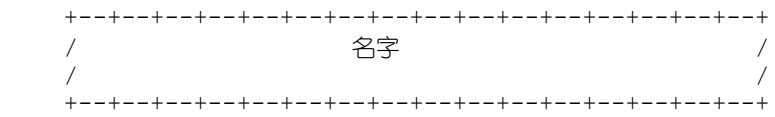
中央处理器 一个<character-string>，它指定CPU类型。

□ 一个<character-string>，它指定操作系统类型。

CPU和OS的标准值可在[RFC-1010]中找到。

HINFO记录用于获取有关主机的一般信息。主要用于FTP等协议，在机器或相同类型的操作系统之间进行通信时可以使用特殊程序。

3.3.3. MB RDATA格式 (实验)

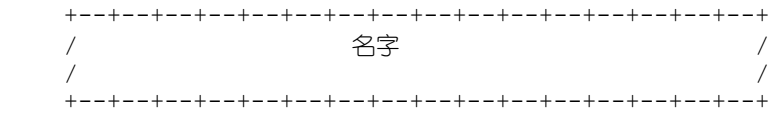


哪里:

名字 <domain-name>，指定具有指定邮箱的主机。

MB记录导致额外的部分处理，其查找与MADNAME相对应的A类型RR。

3.3.4. MD RDATA格式（已淘汰）



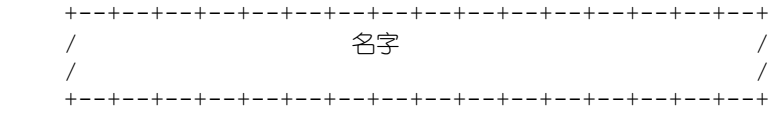
哪里：

名字 一个<domain-name>，它指定一个主机，该主机具有该域的邮件代理，该邮件代理应该能够为该域传递邮件。

MD记录会导致额外的部分处理，查找与MADNAME对应的A类型记录。

MD已经过时了。 有关新方案的详细信息，请参阅MX和[RFC-974]的定义。 处理主文件中的MD RR的建议策略是拒绝它们，或将它们转换为优先级为0的MX RR。

3.3.5. MF RDATA格式（已淘汰）



哪里：

名字 一个<domain-name>，它指定一个主机，该主机具有该域的邮件代理，该邮件代理将接受邮件以转发到域。

MF记录导致额外的部分处理，其查找与MADNAME相对应的A类型记录。

MF已经过时了。 有关新方案的详细信息，请参阅MX和[RFC-974]的定义。 处理主文件中的MD RR的建议策略是拒绝它们，或将它们转换为优先级为10的MX RR。

3.3.6. MG RDATA格式（实验）

哪里：

MGMNEX <domain-name>，指定邮箱，该邮箱是域名指定的邮件组的成员。

MG记录不会导致其他部分处理。

3.3.7. MINFO RDATA格式（实验）

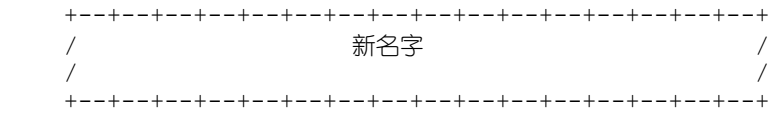
哪里：

RMIPBX <domain-name>，指定负责邮件列表或邮箱的邮箱。 如果此域名命名为root，则MINFO RR的所有者自行负责。 请注意，许多现有邮件列表对邮件列表X的RMAILBX字段使用邮箱X请求，例如，Msgroup的Msgroup-request。 该字段提供了更通用的机制。

埃米布 一个<domain-name>，它指定一个邮箱，用于接收与MINFO RR所有者指定的邮件列表或邮箱相关的错误消息（类似于已提出的ERRORS-TO：字段）。 如果此域名命名为root，则应将错误返回给邮件的发件人。

MINFO记录不会导致额外的部分处理。 虽然这些记录可以与简单邮箱关联，但它们通常与邮件列表一起使用。

3.3.8. MR RDATA格式 (实验)

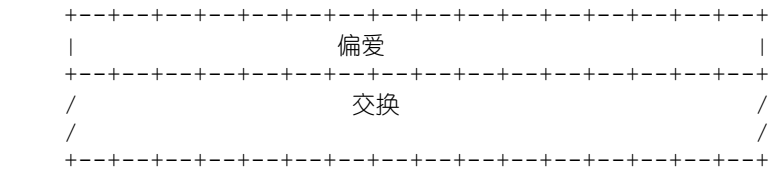


哪里:

新名字 一个<domain-name>，它指定一个邮箱，该邮箱是指定邮箱的正确重命名。

MR记录不会导致其他部分处理。 MR的主要用途是作为转移到不同邮箱的用户的转发条目。

3.3.9. MX RDATA格式



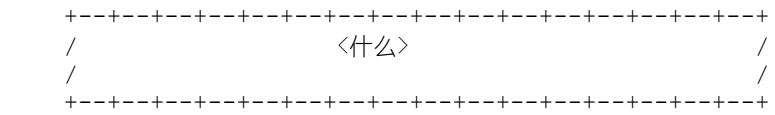
哪里:

偏爱 一个16位整数，指定在同一所有者中给予此RR的优先级。较低的值是优选的。

交换 <domain-name>，指定愿意充当所有者名称的邮件交换的主机。

MX记录导致EXCHANGE指定的主机的类型A附加部分处理。 在[RFC-974]中详细解释了MX RR的使用。

3.3.10. NULL RDATA格式 (实验)



任何东西都可以在RDATA字段中，只要它是65535个八位字节或更少。

NULL记录不会导致其他段处理。 主文件中不允许使用NULL RR。 在DNS的某些实验扩展中，NULL用作占位符。

3.3.11. NS RDATA格式

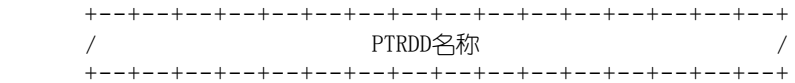
哪里:

NSDND <domain-name>，指定应对指定的类和域具有权威性的主机。

NS记录导致通常的附加区段处理以定位类型A记录，并且当在引用中使用时，特殊搜索它们驻留的区域以获得粘合信息。

NS RR声明命名主机应该具有从指定类的所有者名称开始的区域。 请注意，该类可能不指示应该用于与主机通信的协议族，尽管它通常是一个强烈的提示。 例如，通常使用IN类协议查询作为Internet (IN) 或Hesiod (HS) 类信息的名称服务器的主机。

3.3.12. PTR RDATA格式

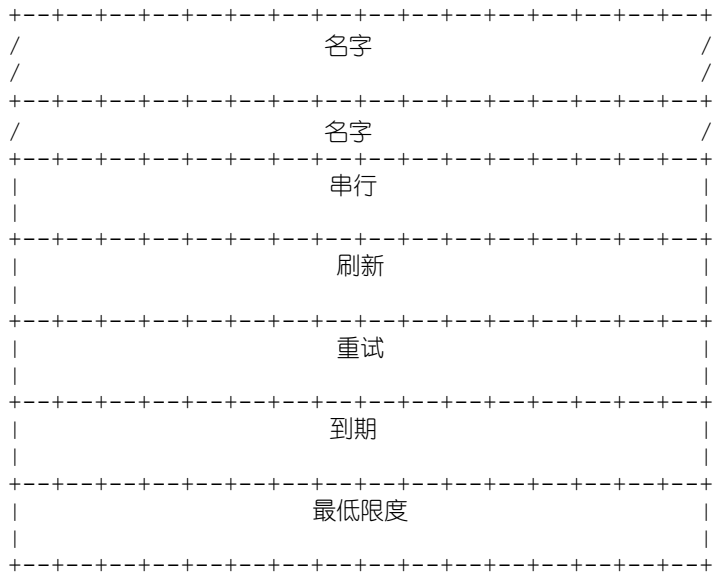


哪里:

PTRDD名称 <domain-name>，指向域名空间中的某个位置。

PTR记录不会导致额外的部分处理。 这些RR在特殊域中用于指向域空间中的其他位置。这些记录是简单数据，并不意味着任何类似于CNAME执行的特殊处理，CNAME标识别名。 有关示例，请参阅IN-ADDR.ARPA域的说明。

3.3.13. SOA RDATA格式



哪里:

名字 名称服务器的<domain-name>，它是此区域的原始数据或主要数据源。

名字 <domain-name>，指定负责此区域的人员的邮箱。

串行 区域原始副本的无符号32位版本号。区域传输保留此值。该值包装并应使用序列空间算法进行比较。

刷新 应刷新区域之前的32位时间间隔。

重试 应重试在刷新失败之前应经过的32位时间间隔。

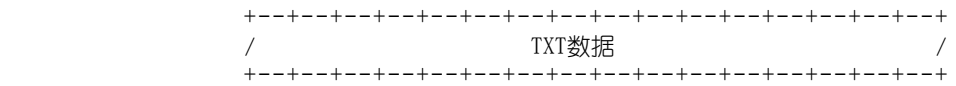
到期 一个32位时间值，指定在区域不再具有权威性之前可以经过的时间间隔的上限。

最低限度 应使用此区域中的任何RR导出的无符号32位最小\TTL字段。

SOA记录不会导致其他部分处理。所有时间都以秒为单位。

这些字段中的大多数仅适用于名称服务器维护操作。 但是，MINIMUM用于从区域检索RR的所有查询操作。 每当在对查询的响应中发送RR时，TTL字段被设置为RR中的TTL字段和适当SOA中的MINIMUM字段的最大值。 因此，MINIMUM是区域中所有RR的TTL字段的下限。 请注意，当将RR复制到响应中时，应使用MINIMUM，而不是在从主文件加载区域或通过区域传输加载时。 这个问题的原因是允许未来的动态更新工具以已知的语义更改SOA RR。

3.3.14. TXT RDATA格式



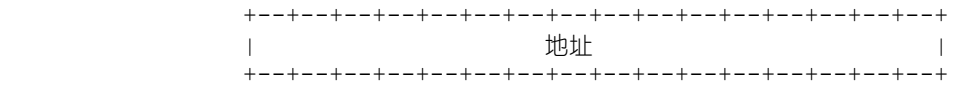
哪里:

TXT数据 一个或多个<character-string>。

TXT RR用于保存描述性文本。 文本的语义取决于找到它的域。

3.4. 特定于互联网的RR

3.4.1. RDATA格式



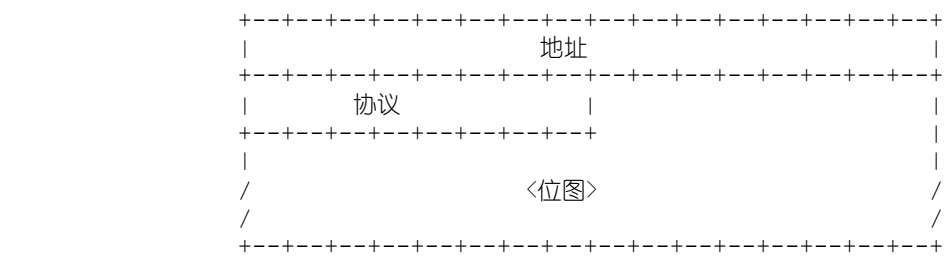
哪里:

地址 一个32位的Internet地址。

具有多个Internet地址的主机将具有多个A记录。

记录不会导致其他部分处理。 主文件中A行的RDATA部分是因特网地址，表示为由点分隔的四个十进制数，没有任何嵌入空格（例如，“10.2.0.52″或“192.0.5.6″）。

3.4.2. WKS RDATA格式



哪里:

地址 一个32位的Internet地址PROTOCOL

一个8位IP协议号

〈位图〉 可变长度位图。 位图必须是8位长的倍数。

WKS记录用于描述特定互联网地址上特定协议支持的众所周知的服务。 PROTOCOL字段指定IP协议号，并且位图在指定协议的每个端口具有一位。 第一位对应端口0，第二位对应端口1，等等。 如果位图不包括感兴趣的协议的位，则该位被假定为零。 端口和协议的适当值和助记符在[RFC-1010]中指定。

例如，如果PROTOCOL = TCP（6），则第26位对应于TCP端口25（smtp）。 如果设置此位，SMTP服务器应该在TCP端口25上侦听;如果为零，则指定的地址不支持SMTP服务。

WKS RR的目的是为TCP和UDP的服务器提供可用性信息。 如果服务器同时支持TCP和UDP，或者具有多个Internet地址，则使用多个WKS RR。

WKS RRs不会导致其他部分处理。

在主文件中，端口和协议都使用助记符或十进制数表示。

3.5. IN-ADDR.ARPA域名

Internet使用特殊域来支持网关位置和Internet地址以进行主机映射。其他类可以在其他域中采用类似的策略。该域的目的是提供一种保证方法来执行主机地址到主机名映射，并便于查询以定位Internet中特定网络上的所有网关。

请注意，这两种服务都类似于可以通过反向查询执行的功能;不同之处在于，域名空间的这一部分是根据地址构建的，因此可以保证在不对域空间进行穷举搜索的情况下定位适当的数据。

该域从IN-ADDR.ARPA开始，并具有遵循Internet寻址结构的子结构。

除IN-ADDR.ARPA后缀外，IN-ADDR.ARPA域中的域名还定义为最多包含四个标签。每个标签代表一个互联网地址的八位字节，并表示为0-255范围内的十进制值的字符串（省略前导零，除了零八位字节由单个零表示）。

主机地址由指定了所有四个标签的域名表示。因此，Internet地址10.2.0.52的数据位于域名52.0.2.10.IN-ADDR.ARPA。反转虽然难以阅读，但允许委派区域，这些区域恰好是一个地址空间网络。例如，10.IN-ADDR.ARPA可以是包含ARPANET数据的区域，而26.IN-ADDR.ARPA可以是MILNET的单独区域。地址节点用于保存指向普通域空间中主要主机名的指针。

网络号对应于IN-ADDR.ARPA域中不同深度的一些非终端节点，因为因特网网络号是1,2或3个八位字节。网络节点用于保存指向连接到该网络的网关的主要主机名的指针。

由于网关根据定义在多个网络上，因此通常会有两个或多个指向它的网络节点。网关还将在其完全限定的地址上具有主机级指针。

网络节点处的网关指针和全地址节点处的正常主机指针都使用PTR RR指向相应主机的主域名。

例如，IN-ADDR.ARPA域将包含有关网络10和26之间的ISI网关的信息，从网络10到MIT的MIT网关

净18，并主持A.ISI.EDU和MULTICS.MIT.EDU。 假设ISI网关的地址为10.2.0.22和26.0.0.103，名称为MILNET-GW.ISI.EDU，而MIT网关的地址为10.0.0.77和18.10.0.4，名称为GW.LCS.MIT.EDU，域数据库将包含：

```
10.in-addr.arpa。          ptr milnet-gw.isi.edu。
10.in-addr.arpa。          ptr gw.lcs.mit.edu。
18.in-addr.arpa。          ptr gw.lcs.mit.edu。
26.in-addr.arpa。          ptr milnet-gw.isi.edu。
22.0.2.10.in-addr.arpa。   ptr milnet-gw.isi.edu。
103.0.0.26.in-addr.arpa。  ptr milnet-gw.isi.edu。
77.0.0.10.in-addr.arpa。   ptr gw.lcs.mit.edu。
4.0.10.18.in-addr.arpa。   ptr gw.lcs.mit.edu。
103.0.3.26.in-addr.arpa。  ptr a.isi.edu。
6.0.0.10.in-addr.arpa。    ptr multics.mit.edu。
```

因此，想要在网络10上定位网关的程序将产生QTYPE = PTR，QCLASS = IN，QNAME = 10.IN-ADDR.ARPA形式的查询。 它会收到两个RR作为回应：

```
10.in-addr.arpa。          ptr milnet-gw.isi.edu。
10.in-addr.arpa。          ptr gw.lcs.mit.edu。
```

然后，程序可以为MILNET-GW.ISI.EDU发起QTYPE = A，QCLASS = IN查询。和GW.LCS.MIT.EDU。发现这些网关的Internet地址。

想要找到与Internet主机地址10.0.0.6对应的主机名的解析器将执行QTYPE = PTR，QCLASS = IN，QNAME = 6.0.0.10.IN-ADDR.ARPA形式的查询，并将收到：

```
6.0.0.10.in-addr.arpa。    ptr multics.mit.edu。
```

有几项注意事项适用于这些服务的使用：

- 由于IN-ADDR.ARPA特殊域和特定主机或网关的普通域将位于不同的区域，因此存在数据可能不一致的可能性。

- 网关通常在不同的域中有两个名称，其中只有一个可以是主要的。

- 使用域数据库初始化其路由表的系统必须从足够的网关信息开始，以保证它们可以访问相应的名称服务器。

- 网关数据仅以与当前HOSTS.TXT文件等效的方式反映网关的存在。 它不会替换GGP或EGP中的动态可用性信息。

3.6. 定义新类型，类和特殊命名空间

以前定义的类型和类是截至本备忘录之日使用的类型和类。 应该预期新的定义。 本节向考虑添加现有设施的设计人员提出一些建议。 邮件列表 NAMEDROPPERS@SRI-NIC.ARPA 是一个讨论设计问题的论坛。

通常，当要将新信息添加到数据库中关于现有对象时，新类型是合适的，或者我们需要新数据格式用于某些全新对象。 设计人员应尝试定义通常适用于所有类的类型及其RDATA格式，并避免重复信息。 当DNS要用于需要新的特定于类的数据格式的新协议等时，或者当需要现有名称空间的副本时，新类是合适的，但是需要单独的管理域。

新类型和类需要主文件的助记符;主文件的格式要求类型和类的助记符是不相交的。

TYPE和CLASS值必须分别是QTYPE和QCLASS的正确子集。

本系统使用多个RR来表示类型的多个值，而不是在单个RR的RDATA部分中存储多个值。 对于大多数应用来说效率较低，但确保RR更短。 多个RR假设被纳入动态更新方法的一些实验工作中。

本系统试图最小化数据库中的数据重复以确保一致性。 因此，为了找到邮件交换的主机地址，您将邮件域名映射到主机名，然后将主机名映射到地址，而不是直接映射到主机地址。 这种方法是首选，因为它避免了不一致的机会。

在定义新类型的数据时，不应使用多个RR类型来创建条目之间的排序或表示等效绑定的不同格式，而是应该在RR的主体中携带此信息并使用单个类型。 此策略避免了缓存多个类型和定义QTYPE以匹配多个类型的问题。

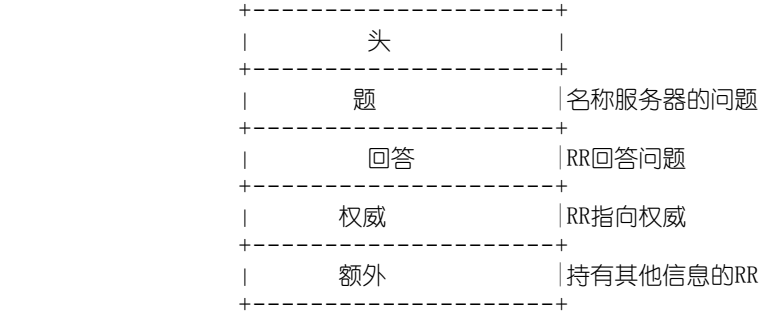
例如，邮件交换绑定的原始形式使用两种RR类型，一种代表“更接近”的交换（MD），另一种代表“不太接近”的交换（MF）。 困难在于高速缓存中存在一种RR类型不会传达关于另一种的任何信息，因为获取高速缓存信息的查询可能使用了MF，MD或MAILA的QTYPE（两者都匹配）。 重新设计

service在RDATA部分使用了一个带有“preference”值的单一类型（MX），它可以命令不同的RR。但是，如果在缓存中找到任何MX RR，则所有MX RR都应该在那里。

4. 消息

4.1. 格式

域协议内的所有通信都以称为消息的单一格式承载。消息的顶级格式分为5个部分（其中一些在某些情况下为空），如下所示：

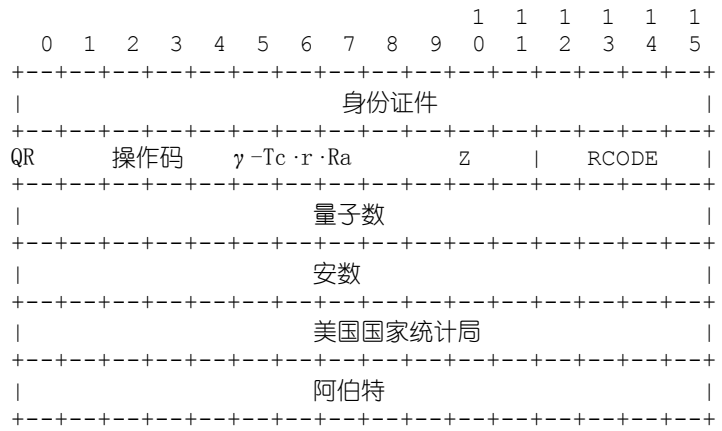


标题部分始终存在。标题包括指定其余部分存在的字段，还指定消息是查询还是响应，标准查询或其他操作码等。

标题之后的节的名称来源于它们在标准查询中的使用。问题部分包含向名称服务器描述问题的字段。这些字段是查询类型（QTYPE），查询类（QCLASS）和查询域名（QNAME）。最后三个部分具有相同的格式：可能为空的连锁资源记录列表（RR）。答案部分包含回答问题的RR;权限部分包含指向权威名称服务器的RR;附加记录部分包含与查询相关的RR，但不是问题的严格答案。

4.1.1. 标题部分格式

标头包含以下字段:



哪里:

身份证件 由程序分配的16位标识符，用于生成任何类型的查询。 此标识符将复制到相应的回复中，并且请求者可以使用该标识符来匹配对未完成查询的回复。

QR 一位字段，指定此消息是查询（0）还是响应（1）。

操作码 一个四位字段，用于指定此消息中的查询类型。 此值由查询的发起者设置并复制到响应中。 价值观是:

0 标准查询（QUERY）

1 反向查询（IQUERY）

2 服务器状态请求（STATUS）

3-15 留作将来使用

机 管 局 权威答案 - 此位在响应中有效，并指定响应名称服务器是相关域名的权限。

请注意，由于别名，答案部分的内容可能有多个所有者名称。 AA位

对应于与查询名称匹配的名称，或者与答案部分中的第一个所有者名称相对应的名称。

锡 TrunCation - 指定此消息由于长度大于传输通道上允许的长度而被截断。

研发 递归所需 - 此位可以在查询中设置并复制到响应中。如果设置了RD，它会指示名称服务器以递归方式查询。
递归查询支持是可选的。

镭 Recursion Available - 可以在响应中设置或清除，并表示名称服务器中是否提供递归查询支持。

Z 保留供将来使用。 所有查询和响应中必须为零。

RCODE 响应代码 - 此4位字段被设置为响应的一部分。 这些值具有以下解释：

0	没有错误的情况
1	格式错误 - 名称服务器无法解释查询。
2	服务器故障 - 由于名称服务器出现问题，名称服务器无法处理此查询。
3	名称错误 - 仅对来自权威名称服务器的响应有意义，此代码表示查询中引用的域名不存在。
4	未实现 - 名称服务器不支持请求的查询类型。
5	拒绝 - 名称服务器因策略原因拒绝执行指定的操作。例如，名称服务器可能不希望向特定请求者提供信息，或者名称服务器可能不希望执行特定操作（例如，区域）

转移）特定数据。

6-15 保留供将来使用。

量子数 无符号的16位整数，指定问题部分中的条目数。

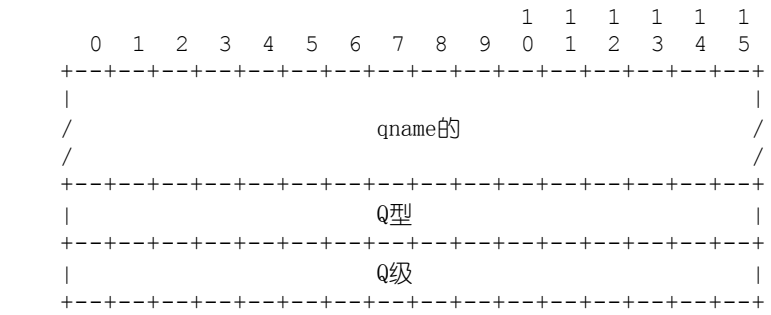
安数 无符号的16位整数，指定答案部分中的资源记录数。

美国国家统计局 无符号16位整数，指定权限记录部分中的名称服务器资源记录数。

阿伯特 无符号的16位整数，指定附加记录部分中的资源记录数。

4.1.2. 问题部分格式

问题部分用于在大多数查询中携带"问题"，即定义所询问内容的参数。 该部分包含QDCOUNT（通常为1）条目，每种条目都是以下格式：



哪里：

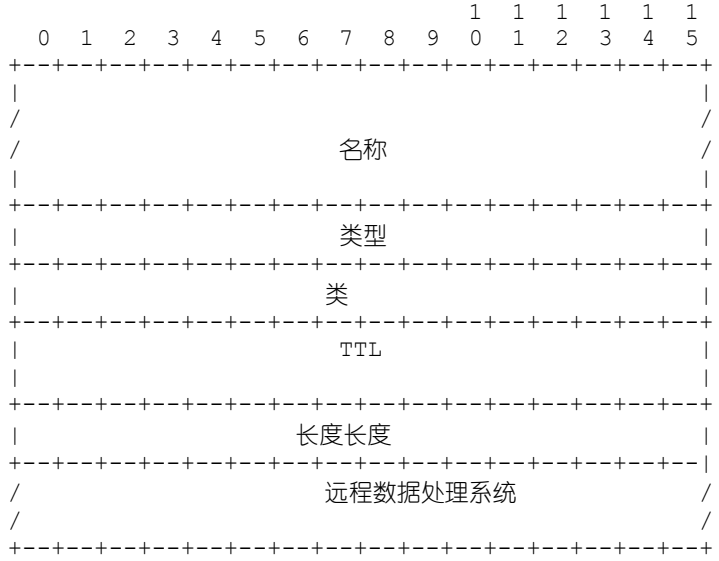
qname的 域名表示为标签序列，其中每个标签由长度八位字节后跟该八位字节数组成。域名以根的空标签的零长度八位字节终止。 注意，该字段可以是奇数个八位字节；没有使用填充。

Q型 两个八位字节代码，用于指定查询的类型。该字段的值包括对TYPE字段有效的所有代码，以及可以匹配多种类型RR的一些更通用的代码。

Q级 两个八位字节代码，用于指定查询的类。
例如，QCLASS字段为Internet的IN。

4.1.3. 资源记录格式

答案，权限和其他部分都共享相同的格式：可变数量的资源记录，其中记录的数量在标题的
相应计数字段中指定。每个资源记录都具有以下格式：



哪里：

名称 此资源记录所属的域名。

类型 两个八位字节包含一个RR类型代码。 该字
段指定RDATA字段中数据的含义。

类 两个八位字节，用于指定RDATA字段中的数据类。

TTL 一个32位无符号整数，指定资源记录在被丢弃之前可以缓存的时间间隔（以
秒为单位）。 零值被解释为RR只能用
于正在进行的事务，并且不应该被缓存。

长度长度 无符号16位整数，指定RDATA字段的八位字节长度。

远程数据处理系统 一个可变长度的八位字节串，用于描述资源。 此信息的格式根据资源记录的TYPE和CLASS而有所不同。例如，如果TYPE是A而CLASS是IN，则RDATA字段是4个八位位组的ARPA因特网地址。

4.1.4. 消息压缩

为了减小消息的大小，域系统使用压缩方案，该方案消除了消息中域名的重复。在该方案中，域名末尾的整个域名或标签列表被替换为指向同一名称的先前出现的指针。

指针采用两个八位字节序列的形式：

```
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
| 1  1|                抵消                     |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

前两位是1位。 这允许指针与标签区分开，因为标签必须以两个零位开始，因为标签限制在63个八位字节或更少。 （10和01组合保留供将来使用。） OFFSET字段指定从消息开始的偏移量（即，域头中的ID字段的第一个八位字节）。 零偏移指定ID字段的第一个字节等。

压缩方案允许将消息中的域名表示为：

- 以零八位字节结尾的标签序列
- 一个指针
- 以指针结尾的标签序列

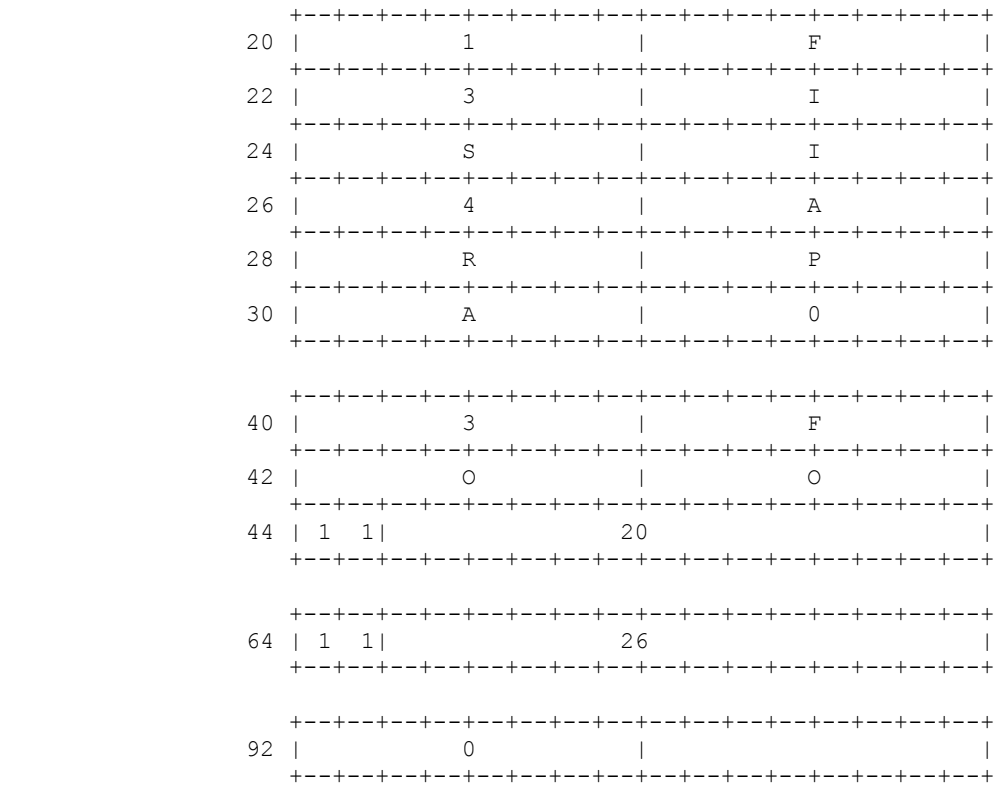
指针只能用于域名的出现，其格式不是特定于类的。如果不是这种情况，则需要名称服务器或解析器知道它处理的所有RR的格式。到目前为止，还没有这种情况，但它们可能会在未来的RDATA格式中出现。

如果域名包含在受长度字段限制的消息的一部分中（例如RR的RDATA部分），则压缩是

使用时，压缩名称的长度用于长度计算，而不是扩展名称的长度。

程序可以自由地避免在它们生成的消息中使用指针，尽管这会减少数据报容量，并可能导致截断。但是，所有程序都需要了解包含指针的到达消息。

例如，数据报可能需要使用域名F.ISI.ARPA，FOO.F.ISI.ARPA，ARPA和根。 忽略邮件的其他字段，这些域名可能表示为：



F.ISI.ARPA的域名显示在偏移量20处。 域名FOO.F.ISI.ARPA显示在偏移40处；此定义使用指针将FOO的标签连接到先前定义的F.ISI.ARPA。 域名ARPA在偏移64处定义，使用指向名称F.ISI.ARPA的ARPA组件的指针为20；请注意，此指针依赖于ARPA是20中字符串中的最后一个标签。 根域名是

由92的单个八位字节定义;根域名没有标签。

4.2. 运输

DNS假定消息将作为数据报或由虚拟电路承载的字节流传输。 虽然虚拟电路可用于任何DNS活动，但由于其较低的开销和更好的性能，数据报是查询的首选。 区域刷新活动必须使用虚拟电路，因为需要可靠的传输。

Internet支持使用服务器端口53（十进制）上的TCP [RFC-793]进行名称服务器访问，以及使用UDP端口53（十进制）上的UDP [RFC-768]进行数据报访问。

4.2.1. UDP使用情况

使用UDP用户服务器端口53（十进制）发送的消息。

UDP承载的消息限制为512字节（不包括IP或UDP标头）。截断较长的消息，并在标头中设置TC位。

UDP不适用于区域传输，但是是Internet中标准查询的推荐方法。 使用UDP发送的查询可能会丢失，因此需要重新发送策略。 查询或其响应可以由网络重新排序，也可以通过名称服务器中的处理重新排序，因此解析器不应依赖于按顺序返回它们。

最佳UDP重传策略将根据Internet的性能和客户端的需求而变化，但建议采用以下方法：

- 在重复查询服务器的特定地址之前，客户端应尝试其他服务器和服务器地址。

- 如果可能，重传间隔应基于先前的统计。 过于激进的重传可能会轻易减慢整个社区的响应速度。 根据客户端与预期服务器的连接情况，最小重传间隔应为2-5秒。

有关服务器选择和重新传输策略的更多建议可以在本备忘录的解析器部分找到。

4.2.2. TCP使用情况

通过TCP连接发送的消息使用服务器端口53（十进制）。 该消息以两个字节长度字段为前缀，该字段给出消息

长度，不包括两个字节长度字段。此长度字段允许低级处理在开始解析之前汇编完整的消息。

建议使用多种连接管理策略：

- 服务器不应阻止等待TCP数据的其他活动。
- 服务器应支持多个连接。
- 服务器应该假定客户端将启动连接关闭，并应延迟关闭其连接结束，直到满足所有未完成的客户端请求。
- 如果服务器需要关闭休眠连接以回收资源，它应该等到连接空闲一段时间大约两分钟。特别是，服务器应该允许在单个连接上进行SOA和AXFR请求序列（开始刷新操作）。由于服务器无论如何都无法回答查询，因此可以使用单边关闭或重置而不是优雅关闭。

5. 主文件

主文件是包含文本形式的RR的文本文件。由于区域的内容可以以RR列表的形式表示，因此主文件最常用于定义区域，尽管它可用于列出缓存的内容。因此，本节首先讨论主文件中RR的格式，然后讨论使用主文件在某个名称服务器中创建区域时的特殊注意事项。

5.1. 格式

这些文件的格式是一系列条目。条目主要是面向行的，但括号可用于跨行边界继续项目列表，文本文字可在文本中包含CRLF。制表符和空格的任意组合充当构成条目的单独项之间的分隔符。主文件中任何行的结尾都可以以注释结束。评论以";"开头（分号）。

定义了以下条目：

〈空白〉[〈评论〉]

$Orth.<域名> >注释>

$ INCLUDE <文件名> [<域名>] [<注释>]

<域名> RR>

<空白> RR> >注释>

在文件中的任何位置都允许有空白行（带或不带注释）。

定义了两个控制条目: $ ORIGIN和$ INCLUDE。 $ ORIGIN之后是域名，并将相关域名的当前来源重置为声明的名称。 $ INCLUDE将指定文件插入到当前文件中，并且可以选择指定用于设置所包含文件的相对域名来源的域名。 $ INCLUDE也可能有评论。 请注意，$ INCLUDE条目永远不会更改父文件的相对原点，无论在包含的文件中对相对原点所做的更改如何。

最后两种形式代表RR。 如果RR的条目以空白开头，则假定RR由最后声明的所有者拥有。 如果RR条目以<domain-name>开头，则重置所有者名称。

<rr>内容采用以下形式之一: [<TTL>] [<class>] <type>

<RDATA>

[<类> [< TTL>] < > < RDAT>

RR以可选的TTL和类字段开头，后跟适合于类型和类的类型和RDATA字段。 类和类型使用标准助记符，TTL是十进制整数。 省略的类和TTL值默认为最后显式指定的值。 由于类型和类助记符是不相交的，因此解析是唯一的。 （请注意，此顺序与示例中使用的顺序和实际RR中使用的顺序不同;给定的顺序允许更容易的解析和默认。）

<domain-name> s构成主文件中的大部分数据。
域名中的标签表示为字符串并用点分隔。 引用约定允许将任意字符存储在域名中。 以点结尾的域名称为绝对域名，并视为完整。 不以点结尾的域名称为相对域;实际域名是相对部分与$ ORIGIN，$ INCLUDE中指定的原点的串联，或者作为主文件加载例程的参数的串联。 没有原点可用时，相对名称是错误。

<character-string>以一种或两种方式表示：作为没有内部空格的连续字符集，或者以“以”结尾“开头的字符串。 在"分隔的字符串中，任何字符都可以出现，除了"本身，必须使用\（反斜杠）引用。

因为这些文件是文本文件，所以需要几种特殊编码来允许加载任意数据。 特别是：

的根。

@ 自由站立@用于表示当前来源。

\X 其中X是除数字（0-9）之外的任何字符，用于引用该字符，以使其特殊含义不适用。 例如，“\。”可用于在标签中放置点字符。

DDD 其中每个D是一个数字是与DDD描述的十进制数相对应的八位字节。 假定生成的八位字节是文本，不检查特殊含义。

( ) 括号用于对跨越行边界的数据进行分组。 实际上，在括号内无法识别行终止。

; 分号用于发表评论;该行的其余部分被忽略。

5.2. 使用主文件定义区域

使用主文件加载区域时，如果在主文件中遇到任何错误，则应该禁止该操作。 这样做的理由是单个错误可能会产生广泛的后果。 例如，假设定义委托的RR具有语法错误;然后，服务器将返回子区中所有名称的权威名称错误（除了服务器上也存在子区域的情况）。

除了确保文件在语法上正确之外，还应执行其他几项有效性检查：

1. 文件中的所有RR应具有相同的类。

2. 区域顶部应该只有一个SOA RR。

3. 如果代表团在场并且需要胶水信息，则应该存在。

4. 区域中权威节点之外的信息应该是粘合信息，而不是原点或类似错误的结果。

5.3. 主文件示例

以下是可用于定义ISI.EDU区域的示例文件。并且加载了ISI.EDU的原点:

```
@    在   SOA      金星          Actual.[域]
                                      20       ;连载7200
                                               ;刷新600
                                               ;重试
                                      3600000;到期
                                      60)      ;最低限度

          纳秒      a.isi.edu。
          纳秒      金星
          纳秒      瓦萨
          MX        10        金星
          MX        20        瓦萨

A         A         26.3.0.103

金星      A         10.1.0.52
          A         128.9.0.32

瓦萨      A         10.2.0.27
          A         128.9.0.33
```

$包括<SysS> IIS-MyBox x.txt

文件<SUBSYS> ISI-MAILBOXES.TXT的位置是: MOE MB

```
                a.isi.edu。
     拉里       MB        a.isi.edu。
     柯利       MB        a.isi.edu。
    傀儡                  教育部
                毫克       拉里
                 毫克       柯利
```

请注意在SOA RR中使用\字符来指定负责人人邮箱“Action.domains@E.ISI.EDU”。

6. 名称服务器实现

6.1. 建筑

名称服务器的最佳结构将取决于主机操作系统以及名称服务器是否与解析程序操作集成，可以通过支持递归服务，也可以通过与解析程序共享其数据库。 本节讨论与解析程序共享数据库的名称服务器的实现注意事项，但大多数这些问题都存在于任何名称服务器中。

6.1.1. 控制

名称服务器必须使用多个并发活动，无论它们是作为主机操作系统中的单独任务实现还是在单个名称服务器程序内进行多路复用。 名称服务器在等待TCP数据进行刷新或查询活动时阻止UDP请求的服务是完全不可接受的。 类似地，名称服务器不应尝试在不同时处理此类请求的情况下提供递归服务，尽管它可能选择序列化来自单个客户端的请求，或者将来自同一客户端的相同请求视为重复请求。 名称服务器在从主文件重新加载区域时或在将新刷新的区域合并到其数据库中时，不应该大大延迟请求。

6.1.2. 数据库

虽然名称服务器实现可以自由选择使用的任何内部数据结构，但建议的结构包含三个主要部分：

- 一个"目录"数据结构，列出了该服务器可用的区域，以及区域数据结构的"指针"。
  此结构的主要目的是找到最近的祖先区域（如果有），以便到达标准查询。

- 名称服务器所拥有的每个区域的单独数据结构。

- 缓存数据的数据结构。（或者可能是不同类的单独缓存）

所有这些数据结构都可以实现相同的树结构格式，不同的数据链接在不同部分的节点上：在目录中，数据是指向区域的指针，而在区域和缓存数据结构中，数据将是RR。 在设计树框架时，设计人员应该认识到查询处理需要使用不区分大小写的标签比较来遍历树。然后

在实际数据中，一些节点具有非常高的分支因子（100-1000或更高），但绝大多数节点具有非常低的分支因子（0-1）。

解决案例问题的一种方法是将每个节点的标签存储为两部分：标签的标准化案例表示，其中所有ASCII字符都在一个案例中，还有一个位掩码，表示哪些字符实际上是不同的情况。 可以使用节点的简单链接列表来处理分支因子分集，直到分支因子超过某个阈值，并且在超过阈值之后转换到散列结构。 在任何情况下，用于存储树节的散列结构必须确保散列函数和过程保留DNS的大小约定。

为数据库的不同部分使用单独的结构是出于以下几个因素：

- 目录结构可以是几乎静态的结构，只有在系统管理员更改服务器支持的区域时才需要更改。 此结构还可用于存储用于控制刷新活动的参数。

- 区域的各个数据结构允许仅通过更改目录中的指针来替换区域。 区域刷新操作可以构建一个新结构，并在完成后通过简单的指针替换将其拼接到数据库中。 在刷新区域时，查询不应同时使用旧数据和新数据，这一点非常重要。

- 通过适当的搜索过程，区域中的权威数据将始终"隐藏"，因此优先于缓存数据。

- 导致重叠区域等的区域定义中的错误可能导致对查询的错误响应，但是问题确定被简化，并且一个"坏"区域的内容不能破坏另一个。

- 由于缓存是最频繁更新的，因此在系统重新启动期间最容易受到损坏。 它也可能充满过期的RR数据。 在任何一种情况下，都可以轻松丢弃它而不会干扰区域数据。

数据库设计的一个主要方面是选择一个允许名称服务器处理名称服务器主机崩溃的结构。名称服务器应在系统崩溃时保存的状态信息

包括目录结构（包括每个区域的刷新状态）和区域数据本身。

6.1.3. 时间

RR的TTL数据和刷新活动的定时数据都取决于以秒为单位的32位定时器。 在数据库内部，刷新定时器和缓存数据的TTL在概念上“倒计时”，而区域中的数据保持不变的TTL。

建议的实施策略是以两种方式存储时间： 作为相对增量和绝对时间。 一种方法是对一种类型使用正32位数，对另一种使用负数。 区域中的RR使用相对时间;刷新定时器和高速缓存数据使用绝对时间。 对于某些已知原点采用绝对数字，并在放置在对查询的响应中时转换为相对值。 当转换为相对值后绝对TTL为负数时，则数据已过期且应被忽略。

6.2. 标准查询处理

标准查询处理的主要算法在[RFC-1034]中给出。

当使用QCLASS = *或其他一些匹配多个类的QCLASS处理查询时，除非服务器能够保证响应涵盖所有类，否则响应永远不应具有权威性。

在编写响应时，可以从附加部分省略要插入附加部分但在答案或权限部分中重复RR的RR。

当响应太长而需要截断时，截断应该从响应结束开始并在数据报中向前进行。 因此，如果权限部分有任何数据，则保证答案部分是唯一的。

应使用SOA中的MINIMUM值来设置从区域分发的数据的TTL。 当数据被复制到响应中时，应该执行此floor函数。 这将允许未来的动态更新协议更改SOA MINIMUM字段而不会产生模糊语义。

6.3. 区域刷新和重新加载处理

尽管服务器做了最大努力，但由于语法错误等原因，它可能无法从主文件加载区域数据，或者无法刷新其到期参数内的区域。 在这种情况下，名称服务器

应该回答问题，好像它不应该拥有该区域。

如果主设备通过AXFR发送区域，并且在传输过程中创建了新版本，则主设备应尽可能继续发送旧版本。 在任何情况下，它都不应该发送一个版本的一部分和另一个版本的一部分。 如果无法完成，主站应重置进行区域传输的连接。

6.4. 反向查询（可选）

反向查询是DNS的可选部分。 名称服务器不需要支持任何形式的反向查询。 如果名称服务器收到它不支持的反向查询，它将返回错误响应，并在标头中设置“未实现”错误。 虽然反向查询支持是可选的，但所有名称服务器必须至少能够返回错误响应。

6.4.1. 反向查询和响应的内容 反向查询反转标准查询操作执行的映射；当标准查询将域名映射到资源时，反向查询将资源映射到域名。 例如，标准查询可能会将域名绑定到主机地址;相应的反向查询将主机地址绑定到域名。

反向查询采用消息的答案部分中的单个RR的形式，具有空的问题部分。 查询RR的所有者名称及其TTL不重要。 该响应在问题部分中包含问题，该问题标识具有名称服务器知道的查询RR的所有名称。 由于没有名称服务器知道所有域名空间，因此永远不能假定响应是完整的。 因此，反向查询主要用于数据库管理和调试活动。 反向查询不是将主机地址映射到主机名的可接受方法;请改用IN-ADDR.ARPA域名。

在可能的情况下，名称服务器应该为反向查询提供不区分大小写的比较。 因此，要求MX RR为“Venera.isi.edu”的反向查询应该得到与“VENERA.ISI.EDU”的查询相同的响应。对于HINFO RR“IBM-PC UNIX”的反向查询应该产生与“IBM-pc unix”的反向查询相同的结果。 但是，这不能保证，因为名称服务器可能拥有包含字符串的RR，但名称服务器不知道数据是字符。

当名称服务器处理反向查询时，它返回：

1. 指定资源的零个，一个或多个域名作为问题部分中的QNAME

2. 一个错误代码，指示名称服务器不支持指定资源类型的逆映射。

当对逆查询的响应包含一个或多个QNAME时，定义逆查询的答案部分中的RR的所有者名称和TTL被修改为与在第一个QNAME处找到的RR完全匹配。

反向查询中返回的RR不能使用与用于标准查询的答复相同的机制进行高速缓存。 其中一个原因是名称可能具有多个相同类型的RR，并且只会出现一个。 例如，对多宿主主机的单个地址进行反向查询可能会产生仅存在一个地址的印象。

6.4.2. 反向查询和响应示例 用于检索与Internet地址10.1.0.52对应的域名的反向查询的总体结构如下所示:

```
                +----------------------+头                    |
                opc= iQuid, ID＝997                           |
                +---------------------------------------------+
题              |                  〈空〉                      |
                +---------------------------------------------+
 回答           |          〈101.0.52中的AyNyMe〉              |
                +---------------------------------------------+
权威            |                  〈空〉                      |
                +---------------------------------------------+
额外            |                  〈空〉                      |
                +---------------------------------------------+
```

此查询请求一个问题，其答案是Internet样式地址10.1.0.52。 由于所有者名称未知，因此任何域名都可以用作占位符（并被忽略）。 通常使用表示根的单个八位字节，因为它最小化了消息的长度。 RR的TTL并不重要。 对此查询的响应可能是:

```
                                +------------------------+头                    |
                                操作码=响应，id = 997                             |
                                + ---------------------------------------- +问题
                                qqType＝a，qCal=＝in，qNoNe= vEnEn.I.EDU
                                + ---------------------------------------- +答案 |
                                伊甸园                  a 10.1.0.52               |
                                +-----------------------------------------+
          权威                  |                    〈空〉                      |
                                +-----------------------------------------+
          额外                  |                    〈空〉                      |
                                +-----------------------------------------+
```

请注意，对反向查询的响应中的QTYPE与反向查询的答案部分中的TYPE字段相同。 当反向不唯一时，对反向查询的响应可能包含多个问题。 如果响应中的问题部分不为空，则修改答案部分中的RR以对应于第一个QNAME处的RR的精确副本。

6.4.3. 反向查询处理

支持反向查询的名称服务器可以通过对其数据库的详尽搜索来支持这些操作，但随着数据库大小的增加，这变得不切实际。 另一种方法是根据搜索关键字反转数据库。

对于支持多个区域和大量数据的名称服务器，建议的方法是针对每个区域进行单独的反转。 在刷新期间更改特定区域时，只需重做其反转。

在域系统的未来版本中可能包含对此类反转的传输的支持，但此版本不支持。

6.5.完成查询和响应

RFC-882和RFC-883中描述的可选完成服务已被删除。 重新设计的服务将来可能会出现。

7. 解析器实现

推荐的解析器算法的最高级别在[RFC-1034]中讨论。本节讨论实现细节，假设本备忘录的名称服务器实现部分中建议了数据库结构。

7.1. 将用户请求转换为查询

解析器采取的第一步是将客户端的请求（以适合本地操作系统的格式说明）转换为特定名称与特定QTYPE和QCLASS匹配的RR的搜索规范。在可能的情况下，QTYPE和QCLASS应该对应于单个类型和单个类，因为这使得缓存数据的使用更加简单。其原因是高速缓存中存在一种类型的数据并不能确认其他类型数据的存在与否，因此唯一可以确定的方法是咨询权威来源。 如果使用QCLASS = *，则无法获得权威答案。

由于解析器必须能够多路复用多个请求才能有效地执行其功能，因此每个待处理请求通常在某些状态信息块中表示。 此状态块通常包含：

- 一个时间戳，指示请求开始的时间。
  时间戳用于确定数据库中的RR是否可以使用或已过期。此时间戳使用先前在区域和高速缓存中针对RR存储所讨论的绝对时间格式。注意，当RRs TTL指示相对时间时，RR必须是及时的，因为它是区域的一部分。 当RR具有绝对时间时，它是高速缓存的一部分，并且将RR的TTL与请求开始的时间戳进行比较。

  请注意，使用时间戳优于使用当前时间，因为它允许TTL为零的RR以常规方式输入缓存，但仍然由当前请求使用，即使在由于系统负载而导致的多秒间隔之后也是如此，查询重传超时等

- 某种参数用于限制将对此请求执行的工作量。

  解析器为响应客户端请求而执行的工作量必须限制为防止数据库中的错误（例如循环CNAME引用）和操作问题（例如阻止

解析器访问它需要的名称服务器。 虽然解析器将特定查询重新传输到特定名称服务器地址的次数的局部限制是必不可少的，但解析器应具有全局每请求计数器以限制单个请求的工作。 计数器应设置为某个初始值，并在解析器执行任何操作（重新传输超时，重新传输等）时递减 如果计数器通过零，则请求以临时错误终止。

请注意，如果解析器结构允许一个请求并行启动其他请求，例如当需要访问一个请求的名称服务器导致名称服务器地址的并行解析时，应该使用较低的计数器启动生成的请求。 这可以防止数据库中的循环引用启动解析器活动的连锁反应。

- SLIST数据结构在[RFC-1034]中讨论。

  如果必须等待来自外部名称服务器的答案，此结构将跟踪请求的状态。

7.2. 发送查询

如[RFC-1034]中所述，解析器的基本任务是制定一个查询，该查询将回答客户端的请求并将该查询定向到可以提供信息的名称服务器。 解析器通常只有非常强烈的提示，要求以NS RR的形式询问哪些服务器，并且可能必须修改查询，以响应CNAME，或者修改解析器要求的名称服务器集，以响应委托响应指出解析器将服务器命名为更接近所需信息。 除了客户请求的信息之外，解析器可能必须调用其自己的服务来确定它希望联系的名称服务器的地址。

在任何情况下，本备忘录中使用的模型都假设解析器在多个请求之间多路复用，一些来自客户端，一些是内部生成的。 每个请求由一些状态信息表示，并且期望的行为是解析器以最大化请求被回答的概率的方式向名称服务器发送查询，最小化请求所花费的时间，并且避免过度传输。 密钥算法使用请求的状态信息来选择要查询的下一个名称服务器地址，并且还计算超时，如果响应未到达，将导致下一个操作。 下一个操作通常是传输给其他服务器，但可能是一个临时错误

客户。

解析器始终以要查询的服务器名称列表（SLIST）开头。该列表将是所有NS RR，其对应于解析器知道的最近的祖先区域。 为了避免启动问题，解析器应该有一组默认服务器，如果它没有适当的当前NS RR，它将会询问它们。 解析器然后将名称服务器的所有已知地址添加到SLIST，并且当解析器具有名称服务器的名称但没有地址时，可以启动并行请求以获取服务器的地址。

要完成SLIST的初始化，解析器会将其拥有的任何历史信息附加到SLIST中的每个地址。 这通常包括地址响应时间的某种加权平均值，以及地址的击球平均值（即地址对请求的响应频率）。 请注意，此信息应基于每个地址而不是基于每个名称服务器，因为特定服务器的响应时间和击球平均值可能因地址而异。 另请注意，此信息实际上特定于解析程序地址/服务器地址对，因此具有多个地址的解析程序可能希望为其每个地址保留单独的历史记录。 这一步的一部分必须处理没有这种历史的地址;在这种情况下，预计5-10秒的往返时间应该是最坏的情况，对同一本地网络的估计值较低等。

请注意，只要遵循委派，解析程序算法就会重新初始化SLIST。

该信息建立了可用名称服务器地址的部分排名。 每次选择一个地址并且应该更改状态以防止再次选择该状态，直到尝试了所有其他地址。 每次传输的超时应比平均预测值大50-100%，以允许响应的变化。

一些细节:

- 解析器可能会遇到SLIST中指定的任何名称服务器都没有可用地址的情况，并且列表中的服务器正是通常用于查找其自己的地址的服务器。 当胶合地址RR的TTL小于NS RR标记委托时，或者当解析器缓存NS搜索的结果时，通常会发生这种情况。 解析器应该检测到这种情况并在下一个祖先区域重新开始搜索，或者在根目录下重新开始搜索。

- 如果解析器从名称服务器收到服务器错误或其他奇怪的响应，它应该从SLIST中删除它，并且可能希望安排立即传输到下一个候选服务器地址。

7.3. 处理回复

处理到达响应数据报的第一步是解析响应。该程序应包括：

- 检查标题是否合理。丢弃作为预期响应时查询的数据报。

- 解析消息的各个部分，并确保所有RR的格式正确。

- 作为可选步骤，检查到达数据的TTL，寻找具有过长TTL的RR。如果RR具有过长的TTL，例如大于1周，则要么丢弃整个响应，要么将响应中的所有TTL限制为1周。

下一步是将响应与当前解析程序请求进行匹配。建议的策略是使用域标头中的ID字段进行初步匹配，然后验证问题部分是否与当前所需的信息相对应。这要求传输算法将域ID字段的若干位专用于某种请求标识符。这一步有几个要点：

- 某些名称服务器从不同于用于接收查询的地址发送其响应。也就是说，解析器不能依赖于响应将来自它发送相应查询的相同地址。UNIX系统中通常会遇到此名称服务器错误。

- 如果解析器将特定请求重新发送到名称服务器，则它应该能够使用来自任何传输的响应。但是，如果它使用响应来对访问名称服务器的往返时间进行采样，则它必须能够确定哪个传输与响应匹配（并保留每个传出消息的传输时间），或者仅基于计算往返时间初始传输。

- 根据某些NS RR，名称服务器有时根本没有区域的当前副本。解析器应该只从当前的SLIST中删除名称服务器，然后继续。

7.4. 使用缓存

通常，我们希望解析器缓存它在响应中收到的所有数据，因为它可能有助于回答未来的客户端请求。
但是，有几种类型的数据不应该被缓存：

- 当具有相同类型的多个RR可用于特定所有者名称时，解析器应该全部缓存它们或者根本不缓存它们。 当截断响应并且解析器不知道它是否具有完整集时，它不应该缓存可能部分的RR集。

- 永远不应该使用缓存数据优先于权威数据，因此如果缓存会导致这种情况发生，则不应缓存数据。

- 不应缓存反向查询的结果。

- 标准查询的结果，如果数据可能用于构造通配符，则QNAME包含“*”标签。 原因是高速缓存不一定包含限制通配符RR的应用所必需的现有RR或区域边界信息。

- RR数据在可疑可靠性的响应中。 当解析器收到未经请求的响应或RR请求之外的RR数据时，它应该丢弃它而不缓存它。 基本含义是对数据包的所有健全性检查应在任何缓存之前执行。

类似地，当解析器在响应中具有针对某个名称的一组RR并且想要缓存RR时，它应该检查其缓存中是否存在已存在的RR。 根据具体情况，响应中的数据或缓存是首选，但两者不应组合。 如果响应中的数据来自答案部分中的权威数据，则始终是首选。

8. 邮件支持

域系统定义了将邮箱映射到域名的标准，以及使用邮箱信息派生邮件路由信息的两种方法。 第一种方法称为邮件交换绑定，另一种方法是邮箱绑定。 邮箱编码标准和邮件交换绑定是DNS官方协议的一部分，是Internet中邮件路由的推荐方法。 邮箱绑定是一项实验性功能，仍处于开发阶段，可能会发生变化。

邮箱编码标准假定表单的邮箱名称
“<本地部分> @ <邮件域>”。 虽然这些部分中允许的语法在各种邮件互联网之间存在很大差异，但ARPA互联网的首选语法在[RFC-822]中给出。

DNS将<local-part>编码为单个标签，并对其进行编码
<mail-domain>作为域名。 来自<local-part>的单个标签以<mail-domain>中的域名开头，以形成与邮箱对应的域名。 因此，邮箱HOSTMASTER@SRI-NIC.ARPA被映射到域名HOSTMASTER.SRI-NIC.ARPA。 如果
<local-part>包含点或其他特殊字符，它在主文件中的表示将需要使用反斜杠引用以确保正确编码域名。 对于例如，邮箱Action.domains@ISI.EDU 将表示为Action \ .domains.ISI.EDU。

8.1. 邮件交换绑定

邮件交换绑定使用邮箱规范的<mail-domain>部分来确定邮件的发送位置。 甚至没有咨询<local-part>。 [RFC-974]详细说明了这种方法，在尝试使用邮件交换支持之前应该参考。

这种方法的一个优点是它将邮件目的地命名与用于支持邮件服务的主机分离，代价是查找功能中的另一层间接。 但是，添加层应该消除<local-part>中复杂的“%”，“！”等编码的需要。

该方法的本质是将<mail-domain>用作域名来定位类型MX RR，列出愿意接受邮件的主机<mail-domain>，以及根据管理员为<mail-domain>指定的顺序对主机进行排名的首选项值。

在本备忘录中，<mail-domain> ISI.EDU在示例中与主机VENERA.ISI.EDU和VAXA.ISI.EDU一起用作ISI.EDU的邮件交换。 如果邮件收到Mockapetris@ISI.EDU的消息， 它会通过查找ISI.EDU的MX RR来路由它。 ISI.EDU上的MX RR名称为VENERA.ISI.EDU和VAXA.ISI.EDU，类型A查询可以找到主机地址。

8.2. 邮箱绑定（实验）

在邮箱绑定中，邮件程序使用整个邮件目标规范来构造域名。邮箱的编码域名用作QTYPE = MAILB查询中的QNAME字段。

此查询可能有以下几种结果：

1．查询可以返回名称错误，指示邮箱不作为域名存在。

从长远来看，这表明指定的邮箱不存在。 但是，在使用邮箱绑定是通用的之前，应将此错误条件解释为表示由全局部分标识的组织不支持邮箱绑定。 适当的程序是在此时恢复交换绑定。

2．该查询可以返回邮件重命名（MR）RR。

MR RR在其RDATA字段中携带新邮箱规范。 邮件程序应使用新邮箱替换旧邮箱，然后重试该操作。

3．查询可以返回MB RR。

MB RR在其RDATA字段中携带主机的域名。 邮件程序应通过适用的任何协议（例如，b，SMTP）将消息传递给该主机。

4．该查询可以返回一个或多个邮件组（MG）RR。

这种情况意味着邮箱实际上是邮件列表或邮件组，而不是单个邮箱。 每个MG RR都有一个RDATA字段，用于标识作为该组成员的邮箱。 邮件程序应将邮件的副本发送给每个成员。

5．查询可以返回MB RR以及一个或多个MG RR。

这种情况意味着邮箱实际上是一个邮件列表。 邮件程序可以将消息传递给MB RR指定的主机，然后MB RR将传递给所有成员，或者邮件程序可以使用MG RR来进行扩展。

在任何这些情况下，响应可以包括邮件信息（MINFO）RR。此RR通常与邮件组相关联，但与MB合法。 MINFO RR识别两个邮箱。 其中一个标识原始邮箱名称的负责人。 此邮箱应该用于添加到邮件组等的请求.MINFO RR中的第二个邮箱名称标识应该收到邮件失败错误消息的邮箱。 当成员名称中的错误应报告给向列表发送消息的人以外的人员时，这尤其适用于邮件列表。

将来可能会向此RR添加新字段。

9.参考文献和参考书目

[代尔87] S. Dyer, F。Hsu, “Hesiod”, Project Athena 技术计划 - 名称服务, 1987年4月, 版本1.9。描述Hesiod名称服务的基础知识。

[IEN-116] J. Postel, “Internet Name Server”, IEN-116, USC / Information Sciences Institute, 1979年8月。

名称服务已被域名系统废弃，但仍在使用中。

[Quarterman 86] J. Quarterman和J. Hoskins, “着名计算机网络”, ACM通讯, 1986年10月, 第29卷, 第10期。

[RFC-72] K. Harrenstien, “NAME / FINGER”, RFC-742, 网络信息中心, SRI国际, 1977年12月。

[RCF-768] J. Postel, “用户数据报协议”, RFC-768, USC /信息科学研究所, 1980年8月。

[RCF-793] J. Postel, “传输控制协议”, RFC-793, USC /信息科学研究所, 1981年9月。

[RCF-799] D. Mills, “Internet Name Domains”, RFC-799, COMSAT, 1981年9月。

建议引入层次结构来代替Internet的平面名称空间。

[RCF-805] J. Postel, “计算机邮件会议记录”, RFC-805, USC /信息科学研究所, 1982年2月。

[RCF-810] E. Feinler, K。Harrenstien, Z。Su和V. White, “DOD Internet Host Table Specification”, RFC-810, Network Information Center, SRI International, 1982年3月。

已过时。 参见RFC-952。

[RCF-811] K. Harrenstien, V。White和E. Feinler, “Hostnames Server”, RFC-811, 网络信息中心, SRI国际, 1982年3月。