Handle系统名称空间和服务定义

英文原文：[https://www.rfc-editor.org/rfc/pdfrfc/rfc3651.txt.pdf](https://www.rfc-editor.org/rfc/pdfrfc/rfc3652.txt.pdf)

本备忘录的状态

本备忘录为互联网社区提供信息。它没有规定任何类型的互联网标准。本备忘录的分发是无限制的。

版权声明

版权所有(C)互联网协会(2003)。 保留所有权利。

IESG注意

IETF和IRTF中的几个小组讨论了Handle系统和它与现有标识符系统的关系。IESG希望指出，这些讨论并没有导致IETF在所描述的Handle系统上达成一致，也没有导致它如何适合用于标识符的IETF体系结构。虽然已经讨论过将Handle作为URI的一种形式，特别是作为URN，但是这些文档描述了名称空间和标识符在互联网上如何工作的另一种观点，并包含了可能与IETF共识视图不匹配的现有系统的特征。

摘要

Handle系统是一种通用的全局名称服务，它允许在公共互联网上进行安全的名称解析和管理。本文档提供了Handle系统名称空间及其数据、服务和操作模型的详细描述。名称空间定义指定Handle语法及其语义结构。数据模型定义了Handle系统协议使用的数据结构和用于执行Handle服务的任何预定义数据类型。服务模型提供各种Handle系统组件的定义，并解释它们如何在网络上协同工作。最后，基于Handle系统的操作模型从客户机和服务器之间传输的消息描述服务操作，基于Handle系统认证协议的客户机认证过程。

目录

1. 介绍。。。。。。。。。。。。。。。。。。。。。。。。。2
2. Handle系统名称空间中。。。。。。。。。。。。。。。。。3.
3. Handle系统数据模型。。。。。。。。。。。。。。。。4   
   3.1。Handle值集。。。。。。。。。。。。。。。。。。。。4

3.2。预定义的Handle数据类型……9   
3.2.1之上。handle管理员:HS\_ADMIN…10

3.2.2。服务站点信息:HS\_SITE…14

3.2.3。命名权威委托服务: HS\_NA\_DELEGATE………19

3.2.4。 服务Handle:HS\_SERV………20.

3.2.5。 别名Handle:HS\_ALIAS………21

3.2.6。主要站点:HS\_PRIMARY……21

3.2.7。handle值列表:HS\_VLIST………22

* + - 1. Handle系统服务模型…………22   
         4.1。Handle系统服务组件……23以下  
         4.4.1。全局Handle注册(GHR)……23

4.1.2。本地Handle服务(LHS)……26日  
4.2。Handle系统中间件组件……  
4.2.1。准备27日Handle系统缓存服务……27

4.2.2。Handle系统代理服务器……28日

4.3。Handle系统客户机组件……28

* 1. Handle系统运作模式…………29日

5.1。Handle系统服务请求和响应…30.

5.2。Handle系统认证协议……32

* 1. 安全方面的考虑。。。。。。。。。。。。。。。。。37
  2. 确认。。。。。。。。。。。。。。。。。。。。38
  3. 参考文献和参考书目…………38
  4. 作者的地址。。。。。。。。。。。。。。。。。。。40
  5. 完整的版权声明。。。。。。。。。。。。。。。。41

# 介绍

Handle系统将Handle管理为互联网资源的全局惟一名称。它最初是由Robert Kahn和Robert Wilensky在1995年的一篇论文中构想和描述的。Handle系统提供了一个通用的全局名称服务，允许在公共互联网上安全地解析和管理Handle。Handle系统将其服务分为两类:Handle解析服务和Handle管理服务。客户机使用Handle解析服务将Handle解析为它们的值。Handle管理服务处理客户机请求来管理这些Handle，包括添加和删除Handle，以及更新Handle值。

“Handle系统概述”文档[1]提供了Handle系统的体系结构概述，以及它与其他互联网服务(如DNS[2,3]和LDAP[4])的关系。本文档提供了Handle系统名称空间、它的数据和服务模型以及它的操作模型的详细描述。它假设读者熟悉概述文档中描述的Handle系统的基本概念。

名称空间定义指定Handle语法及其语义结构。数据模型定义了Handle系统协议使用的数据结构和用于执行Handle服务的任何预定义数据类型。服务模型提供各种Handle系统组件的定义，并解释它们如何在网络上协同工作。最后，基于Handle系统的操作模型从客户机和服务器之间传输的消息描述服务操作，基于Handle系统认证协议的客户机认证过程。

# Handle系统命名空间

Handle是由很多字符组成的字符串。Handle系统中的每个Handle由两部分组成:它的命名权威，以及在命名权威下唯一的本地名称。命名权威和本地名称由ASCII字符“/”(字节0x2F)分隔。下表提供了ABNF[5]表示法的Handle语法定义:

<Handle>= <NamingAuthority> "/" <LocalName>

<NamingAuthority> = \*(<NamingAuthority>".") <NAsegment>

<NAsegment>= 1\*(%x00-2D / %x30-3F / %x41-FF)

* 任何映射到UTF-8编码的Unicode 2.0字符
* 除了 ' 0x2E '和' 0x2F '(其中
* 对应于ASCII字符“.”
* 和“/”)。

<LocalName> = \* (% x00-FF)

* 任何映射到UTF-8编码的Unicode 2.0字符

表2.1:Handle语法

如表2.1所示，<NamingAuthority>和<LocalName>都是UTF-8[6]编码的字符串。Handle系统协议要求对通过网络传输的Handle进行UTF-8编码。<LocalName>可以由Unicode 2.0标准[7]中的任何字符组成。<NamingAuthority>可以使用Unicode 2.0标准中的任何字符，除了ASCII字符' / ' (0x2F)，它是用来将<NamingAuthority>与<LocalName>分开的。一个<NamingAuthority>可以由多个非空的<NAsegment>段组成，每个段由ASCII字符“.” （0x2e）分隔。

命名权威以类似于树结构的分层方式定义。树的每个节点和叶子都有一个对应于命名权威段的标签(<NAsegment>)。父节点表示父命名权威。从左到右构造命名权威，将标签从树的根连接到表示命名权威的节点。每个标签(或其<NAsegment>)由字符“.”（0x2e）分隔。例如，数字对象标识符(DOI)项目的命名权威是“10”。它是根级别的命名权威，因为它自己没有父级的命名权威。然而，它可以有许多子命名权威。例如，“10.1045”是D-Lib杂志的“10”的子命名权威。

默认情况下，Handle是区分大小写的。但是，Handle服务(全局的或本地的)可以实现它的名称空间，因此名称空间下的ASCII字符被视为大小写不敏感的。例如，全局Handle服务(正式名称为全局Handle注册表(GHR))的实现方式是将ASCII字符视为大小写不敏感的。由于GHR管理命名权威的所有Handle，因此命名权威中的ASCII字符被视为不区分大小写。

# Handle系统数据模型

Handle系统通过公共互联网提供名称-值绑定服务。每个Handle可以有一组分配给它的值。Handle系统维护每个Handle的值集，并返回它来响应任何Handle解析请求。Handle系统数据模型定义了这些值的概念数据结构。协议使用的数据模型可能与任何特定实现中用于存储的物理数据模型不完全相同。相反，它是“Handle系统协议规范”[8]中指定的Handle系统协议所遵循的数据模型。

## 3.1Handle值集

每个Handle可以有一组分配给它的值。这些Handle值对其数据使用一个公共数据结构。例如，每个Handle值都有一个惟一的索引号，将其与值集合中的其他值区分开来。除此之外，每个Handle值还包含一组管理信息，如TTL和权限。图3.1显示了具有三个Handle值的Handle“10.1045/may99-payette”。其中一个值(索引号设置为1)被详细显示。(注意，图3.1中没有显示每个字段的长度编码。此外，空的<reference>字段由一个值为0的4字节整数组成。

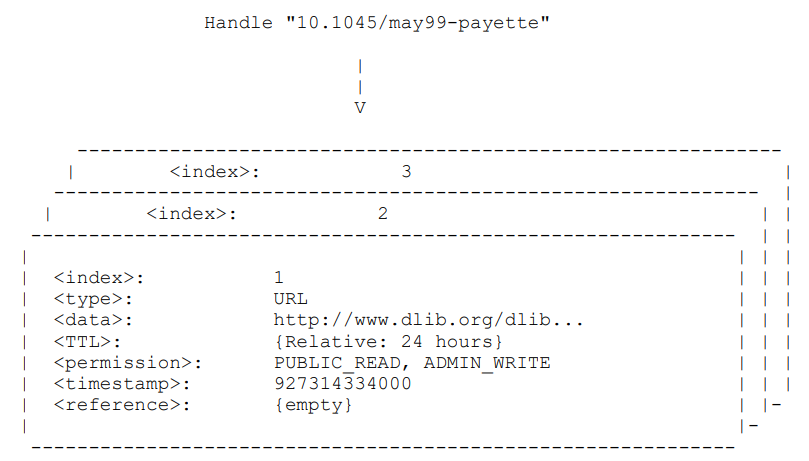


图3.1:Handle“10.1045/may99-payette”及其值集合

在图3.1中，它显示了一个Handle值，其索引设置为1。Handle值的数据类型是URL。在<data>字段中声明的URL数据是“http://www.dlib.org/dlib..”。TTL (time to live)条目建议在再次查询信息源之前的24小时内缓存值记录。<permission>字段授予任何人读取的权限，但是只有管理员可以更新值。<reference>字段为空。它可能包含对其他Handle值的引用列表，作为此Handle值的凭据。

因此，Handle值可以看作是由一组数据字段组成的记录。每个数据字段的定义如下:

<index>

一个无符号32比特整数，惟一地从其他Handle值标识Handle值。

<type>

标识值记录的数据类型的UTF8字符串。注意，在整个文档中，UTF8字符串被定义为一个数据结构，它由一个4字节无符号整数和一个UTF-8编码的字符串组成。该整数指定字符串中的字节数。

<type>字段标识下一个<data>字段中定义数据语法和语义的数据类型。数据类型可以在Handle系统中注册，以避免潜在的冲突。Handle系统有一个保留的命名权威“0”。用于注册数据类型。例如“URL”(如图3.1所示)

是已注册的数据类型。它被注册为Handle“0.TYPE/URL”。Handle可以有一个值来解释数据类型的语法和语义。

Handle系统下的数据类型可以是分层的。层次结构的每一层都可以用不含“.”(0x2E)的UTF8字符串命名。“.”字符用于标记层次级别之间的边界。例如，Handle系统数据类型“a.b”可以被认为是“a”类型下的子类型“b”。类似地，Handle <type> "a.b.x”、“a.b.y”和“a.b.z”的值可以视为公共类型层次结构“a.b”下的Handle值。

对于任何Handle值，<type>字段中的UTF8字符串不能以字符“.”结尾。换句话说，任何Handle系统数据类型都不应该以“.”字符结尾。然而,“.”字符可能出现在Handle查询中<type>参数的末尾。这用于查询公共类型层次结构下的所有Handle值。例如，可以通过将<type>参数设置为"a.b."，来查询类型层次结构“a.b”下的所有Handle值(例如，Handle<type> "a.b.x”、“a.b.y”和“a.b.z”的值)。注意，上述<type>参数以“.”字符结束。Handle查询操作的详细信息可以在Handle系统协议规范[8]中找到。

<data>

一个描述由Handle标识的资源的字节序列(最前面4字节无符号整数表示序列长度)。这些字节的语法和语义由<type>字段标识。

<permission>

一种8位的位掩码，用于Handle值的访问控制。访问控制是根据读、写和执行权限定义的，适用于一般公共或handle管理员。每个Handle值的权限字段可以指定为以下任意位的组合:

|  |  |  |
| --- | --- | --- |
| PUBLIC\_WRITE | (0x01) | 允许任何人修改或删除handle值的权限。 |
| PUBLIC\_READ | (0x02) | 允许任何人读handle值的权限 |
| ADMIN\_WRITE | (0x04) | 允许任何Handle管理员修改或删除handle值的权限 |
| ADMIN\_READ | (0x08)\_ | 允许该Handle值被任何handle管理员以AUTHORITIVE\_READ特权读 |
| PUBLIC\_EXECUTE | (0x10) | 允许任何人执行程序的权限，在handle主机上标识handle值为匿名用户。由于这可能带来的安全风险，实现可能选择不支持这种权限，或者提供选项以便在部署时禁用。 |
| ADMIN\_EXECUTE | (0x20) | 允许handle管理员在handle服务器上运行由handle值标识的程序的权限。handle服务器必须在执行程序之前对handle管理员进行身份验证。handle管理员必须在handle服务器上有一个已建立的帐户。handle值的执行应该与赋予handle管理员的帐户的特权相同。由于这可能带来的安全风险，实现可能选择不支持这种权限，或者提供选项以便在部署时禁用它。 |

注意，没有PUBLIC\_READ或ADMIN\_READ权限的Handle值不能离开Handle服务器。例如，它可以用于存储用于身份验证的密钥。既没有PUBLIC\_WRITE权限也没有ADMIN\_WRITE权限的Handle值使Handle值不可变，任何Handle管理员(通过Handle系统协议)都不能删除它。

给定Handle的管理员必须为每个Handle值指定权限。实现可以选择PUBLIC\_READ和ADMIN\_WRITE作为每个Handle值的默认权限。handle服务器必须在完成任何客户机请求之前检查权限。

<TTL>

一个字节后跟一个4字节整数，指定值记录的存活时间。它用于描述在再次查询信息源之前，可以缓存值记录多长时间。TTL的零值表示该值记录应该只用于正在进行的事务，而不应该被缓存。任何非零的TTL都是根据TTL类型(在第一个字节中指定)和TTL值(TTL类型后面的32位无符号整数)来定义的。TTL类型指示TTL值是绝对的还是相对的。TTL的绝对值以秒为单位定义自1970年1月1日00:00:00 UTC起的时间。相对TTL指定从客户机从任何Handle服务器获得该值后经过的秒数作为生存时间。

<timestamp>

一个8字节(长)的整数，记录最后一次在服务器上更新值的时间。字段包含自00:00:00 UTC(1970年1月)起经过的时间(以毫秒为单位)。毫秒的选择是为了在更新值时避免潜在的冲突。

<reference>

一个4字节的整数，后跟对其他Handle值的引用列表。该整数指定列表中的引用数量。列表中的每个引用指向另一个Handle值，表示为一个UTF8字符串和一个4字节的整数(其中UTF8字符串是Handle名称，整数是值索引)。引用通常用于将凭据添加到当前Handle值。例如，一个Handle值通过引用一个由通常受信任的实体发出的数字签名可以使自己更值得信任。

默认情况下，Handle系统在响应任何解析请求时返回具有公共读权限的所有Handle值。客户机可以使用特定的数据类型(例如，分配给Handle的所有url)来请求这些值的子集。客户机还可以根据特定的值索引请求特定的Handle值。

通过Handle及其值索引的组合，可以惟一地引用每个Handle值。在更改值索引时必须小心，因为它可能会破坏对Handle值的现有引用。例如，假设HandleX/Y有一个索引为1的值。这个值可以称为X/Y:1。如果Handle管理员将值索引从1更改为2，那么对X/Y:1的引用将会过时。对Handle值的任何引用都必须更改为X/Y:2。

分配给任何Handle的值记录可能有也可能没有连续的索引号。也不能假定索引从0或1开始。Handle管理员可以使用任何索引分配Handle值，只要每个索引在值集中是唯一的。

通过将其<permission>字段设置为“authorized-read”，可以将Handle值“私有化”或“禁用”。这限制为仅对Handle管理员的读访问权限。然后可以使用“私有化”值来保存任何历史数据(代表Handle管理员)，而不必将其公开。这种方法还可以用于防止意外重用任何过时的Handle或命名权威。

## 3.2预定义的Handle数据类型

每个Handle值必须在其<type>字段中指定一个数据类型。Handle系统提供了一种类型注册服务，允许组织为其应用程序注册新的数据类型。数据类型可以注册为“0.TYPE”下的Handle。例如，URL数据类型在Handle系统下注册为Handle“0.TYPE/URL”。Handle可以有一个引用RFC1738[9]的Handle值，RFC1738[9]是一个IETF标准文档，定义了URL的语法和语义。

Handle系统预先定义了一组数据类型来执行Handle服务。例如，HS\_ADMIN是一个预定义的数据类型，用于描述Handle管理员或管理员组。

HS\_SITE是一个预定义的数据类型，用于描述任何Handle系统服务组件的服务接口。下面的部分提供了Handle系统下这些预定义数据类型的详细描述。

### 3.2.1 Handle管理员:HS\_ADMIN

每个Handle有一个或多个管理员。任何管理操作(例如，添加、删除或修改Handle值)只能由具有足够权限的Handle管理员执行。Handle管理员是根据HS\_ADMIN值定义的。每个Handle必须至少有一个定义其管理员的HS\_ ADMIN值。每个HS\_ADMIN值可用于定义一组共享相同管理权限的Handle管理员。具有不同权限的多个管理员的Handle可能具有多个HS\_ADMIN值。Handle系统使用HS\_ADMIN值在完成任何Handle管理请求之前对Handle管理员进行身份验证。

如上所述，命名权威本身注册为保留的命名权威“0.NA”下的Handle。这些Handle称为命名权威Handle。任何命名权威的管理员都被定义为相应的命名权威Handle的管理员。例如,“0.NA/10”是命名权威“10”的命名权威Handle。因此，命名权威Handle为“0”的任何管理员。NA/10”也是命名权威“10”的管理员。命名权威管理员是唯一能够在命名权威下创建Handle或子命名权威的人。子命名权威可以定义自己的一组管理员，以创建Handle或进一步级别的子命名权威。例如，命名权威“10.1045”可能拥有一组与其父命名权威“10”完全不同的管理员。

HS\_ADMIN值是一个Handle值，其<type>字段是HS\_ADMIN，其<data>字段由以下条目组成:

<AdminRef>

对Handle值的引用。该引用由Handle名称(UTF8字符串)和用于Handle值索引的4字节无符号整数组成。Handle值标识Handle的管理员集。

<AdminPermission>

一个16位的位掩码，定义由HS\_ADMIN值标识的一组Handle管理员的管理特权。

<AdminRef>条目引用一个Handle值，该Handle值可用于对Handle管理员进行身份验证。这样的Handle值称为Handle管理员引用。Handle管理员引用可能包含由Handle管理员提供的密钥、公钥或X.509证书[10]。例如，<AdminRef>条目可能包含一个Handle管理员引用，其<type>字段是DSS\_WITH\_DES\_CBC\_SHA，其<data>字段包含DES密钥[11]，用于CBC操作模式[12,13]。Handle服务器可以使用密钥对Handle管理员进行身份验证。为了实现更强的密码算法，Handle管理员引用可以包含一组三重DES密钥[23]，并将其<type>设置为DES-EDE3-WITH-CBC。

可以使用HS\_ADMIN值和Handle管理员引用分配单个Handle。换句话说，<AdminRef>条目可能引用分配给具有HS\_ADMIN值的同一Handle的Handle值。在这种情况下，Handle管理员的身份验证不依赖于任何其他Handle。或者，Handle管理员引用可以是另一个Handle下的Handle值。因此，来自不同Handle的HS\_ADMIN值可以共享一个公共Handle管理员引用。这个特性允许在不同的Handle之间共享Handle管理员。Handle管理员引用包含这些Handle的管理员提供的密钥、公钥或X.509证书。

Handle管理员引用可以是HS\_VLIST类型，其<data>字段包含对其他Handle值的引用列表。每个Handle值定义一个Handle管理员引用。HS\_VLIST值定义一个管理员组。HS\_VLIST中的每个Handle管理员引用都是管理员组的成员。每个Handle值引用是根据<Handle>:<索引>对定义的。管理员组也可以包含其他管理员组作为其成员。这允许以分层方式定义管理员组。但是，必须注意避免管理员或管理员组的循环定义。由于缺乏效率，应该避免多级管理员组，但不会将其标记为错误。客户机软件应该准备检测任何可能的管理员循环定义或<AdminRef>条目，这些条目指向不存在的Handle值，并将其视为错误。

Handle可以有多个HS\_ADMIN值，每个HS\_ADMIN值定义一个不同的Handle管理员。不同的管理员可以扮演不同的角色或被授予不同的权限。例如，命名权威Handle为“0.NA/10”可能有两个管理员，其中一个可能只有权限在命名权威下创建新Handle，而另一个可能有权限创建新的子命名权威(例如，“10.1045”)。Handle管理员可能的权限集定义如下:

Add\_Handle (0x0001)

此权限允许命名权威管理员在给定的命名权威下创建新Handle。

Delete\_Handle (0x0002)

此权限允许命名权威管理员删除给定命名权威下的Handle。

Add\_NA (0x0004)

此权限允许命名权威管理员创建新的子命名权威。

Delete\_NA (0x0008)

此权限允许命名权威管理员删除现有的子命名权威。

Modify\_Value (0x0010)

该权限允许Handle管理员修改除HS\_ADMIN值之外的任何Handle值。HS\_ADMIN值用于定义Handle管理员，由一组不同的权限管理。

Delete\_Value (0x0020)

该权限允许Handle管理员删除除HS\_ADMIN值之外的任何Handle值。

Add\_Value (0x0040)

该权限允许Handle管理员添加除HS\_ADMIN值之外的Handle值。

Modify\_Admin (0x0080)

此权限允许Handle管理员修改HS\_ADMIN值。

Remove\_Admin (0x0100)

该权限允许Handle管理员删除HS\_ADMIN值。

Add\_Admin (0x0200)

该权限允许Handle管理员添加新的HS\_ADMIN值。

Authorized\_Read (0x0400)

此权限授予Handle管理员使用ADMIN\_READ权限handle值的读访问权限。没有此权限的管理员将无法访问读访问身份验证的需要的handle值。

LIST\_Handle (0x0800)

此权限允许命名权威管理员列出给定命名权威下的Handle。

LIST\_NA (0x1000)

此权限允许命名权威管理员在给定的命名权威下列出直接的子命名权威。

管理员权限编码在任何HS\_ADMIN值的<data>字段中的<AdminPermission>条目中。每个权限都被编码为一个位标志。如果标志设置为1，则授予权限，否则设置为0。

图3.2.1显示了HS\_ADMIN值的一个示例，该值为命名权威Handle“0.NA/10”定义了一个管理员。在图3.2.1中，命名权威管理员由分配给命名权威Handle“0.NA/10”的HS\_ADMIN值标识。可以根据Handle值“0”对管理员进行身份验证。NA/10”:3，是分配给命名权威Handle“0”的Handle值。NA/10”，并将其索引设置为3。Handle值“0”。NA/10”:3可能包含管理员使用的密钥或公钥。管理员被授予在“10”下添加、删除或修改子命名权威的权限，并直接在命名权威下添加或删除Handle。管理员还可以添加、删除或修改分配给命名权威Handle的任何Handle值(HS\_ADMIN值除外)。换句话说，管理员不允许为命名权威添加、删除或修改任何管理员。

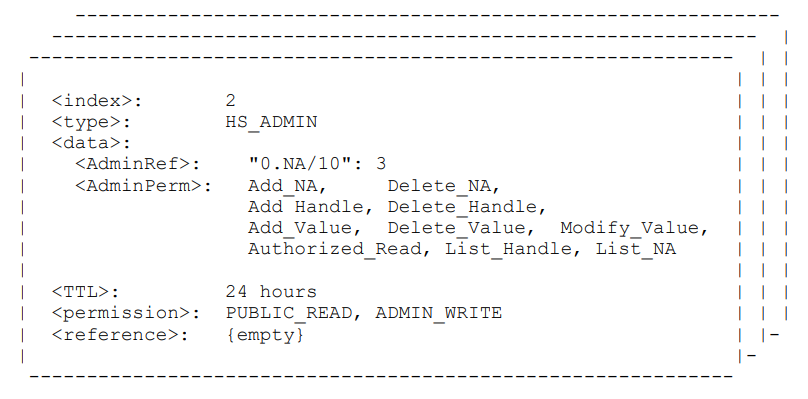


图3.2.1:命名权威Handle“0.NA/10”的管理员

Handle服务器使用HS\_ADMIN值在完成任何管理请求之前对Handle管理员进行身份验证。服务器通过检查客户机是否拥有与Handle管理员引用中的密钥匹配的密钥(或私钥)来对客户机进行身份验证。身份验证是通过Handle系统身份验证协议进行的，本文档后面将对此进行描述。

HS\_ADMIN值可能需要对读访问进行身份验证，以防止数据公开。另外，包含管理员密钥的Handle管理员引用应该既没有PUBLIC\_READ权限，也没有ADMIN\_READ权限来阻止密钥离开服务器。

### 3.2.2 服务站点信息:HS\_SITE

Handle系统由单个分布式全局Handle服务(也称为全局Handle注册表(GHR))和无限数量的本地Handle服务(LHSs)组成。每个Handle服务，无论是全局的还是本地的，都可以复制到多个服务站点。每个服务站点可能由多个服务器计算机组成。针对任何Handle服务的服务请求可以分布到不同的服务站点中，也可以分布到任何服务站点中的不同服务器计算机中。这样的体系结构确保每个Handle服务能够管理任意数量的Handle和handle请求。它还为每个Handle服务提供了避免任何单点故障的方法。

每个Handle服务，无论是全局的还是本地的，都可以提供相同的一组功能来解析和管理它的Handle集合。Handle服务的主要区别在于每个服务负责一组不同的Handle。它们的组件(如用于提供handle解析和管理的服务器)的选择、数量和配置也可能不同。不同的组织可以创建和管理不同的Handle服务。他们每个人都有自己的目标和策略。

服务站点通常由驻留在本地互联网域中的服务器计算机集群组成。这些计算机协同工作，在站点中分配数据存储和处理负载。虽然不推荐，但可以从分布广泛的不同位置的服务器组成站点。此外，甚至可以使用同一组服务器组合两个不同的站点。

每个服务站点由HS\_SITE值定义。HS\_SITE是一个预定义的Handle系统数据类型。HS\_SITE值通过标识包含服务配置（如端口号）的服务器计算机（如IP地址）来定义服务站点。HS\_SITE值通常分配给命名权威Handle。分配给命名权威Handle的一组HS\_SITE值称为该命名权威的服务信息。

服务信息由命名权威管理员管理。它必须反映命名权威的Handle服务的配置。请注意，可以使用一个称为服务Handle的附加间接层来允许多个命名权威引用一组HS\_SITE值，如本文档后面所述(参见第3.2.3节)。Handle系统的客户机在发送其服务请求之前，依赖于服务信息来定位负责的Handle服务器。客户机还可以使用服务信息来验证来自Handle服务器的任何服务响应。

HS\_SITE值是一个Handle值，其<type>字段是HS\_SITE，其<data>字段由以下条目组成:

<Version>

一个2字节的值，用于标识HS\_SITE的版本号。版本号标识HS\_SITE值使用的数据格式。它被定义为允许随时间的推移而向后兼容。本文档定义版本号为0的HS\_SITE。

<ProtocolVersion>

标识Handle协议版本的2字节整数值。值的高字节表示主版本，低字节表示次版本。Handle系统协议的详细信息在[8]中指定。

<SerialNumber>

一个2字节的整数值，每改变一次HS\_SITE值就增加1(并可能环绕到0)。Handle系统协议中使用它来同步客户机和服务器之间的HS\_SITE值。

<PrimaryMask>

标识Handle服务的主站点的8位掩码。字节的第一个位是<MultiPrimary>位。它指示Handle服务是否有多个主站点。字节的第二个位是<PrimarySite>位。它指示HS\_SITE值是否为主站点。主站点是支持其Handle的管理操作的站点。具有0值的<MultiPrimary>条目表示Handle服务只有一个主站点，所有Handle管理必须在该站点上完成。非零<MultiPrimary>条目表示Handle服务有多个主站点。每个主站点可用于管理Handle服务下管理的Handle。此类服务管理的Handle可以使用HS\_PRIMARY值标识其主要站点，如3.2.5节所述。

<HashOption>

一个8位字节，用于标识服务站点用于在其服务器之间分发Handle的散列选项。有效的选项包括HASH\_BY\_NA (0x00)、HASH\_BY\_LOCAL (0x01)或HASH\_BY\_HANDLE (0x02)。这些选项分别指示哈希操作应该仅应用于Handle的命名权威部分，还是仅应用于Handle的本地名称部分，还是应用于整个Handle。每个服务站点使用标准的MD5哈希算法[14]在其服务器之间分发Handle。

<HashFilter>

保留供将来使用的UTF8字符串条目。

<AttributeList>

一个4字节的整数，后跟一个UTF8字符串对列表。该整数表示随后的UTF8字符串对的数目。每个UTF8字符串对是一个<attribute>:<value>对。它们用于添加服务站点的文字说明。例如，如果<attribute>是“组织”，<value>应该包含承载服务站点的组织的描述。可以定义其他<attribute>来帮助区分服务站点。

<NumOfServer>

定义服务站点中服务器数量的4字节整数。条目后面是一个<ServerRecord>列表。每个<ServerRecord>定义一个Handle服务器，它是服务站点的一部分。每个<ServerRecord>由以下数据字段组成:

<ServerRecord>:: = <ServerID>

< Address> <PublicKeyRecord> <ServiceInterface>

其中每个字段定义如下:

<ServerID>

一个4字节无符号整数，唯一标识服务站点下的服务器进程。<ServerID>非必须以1开头，也非必须是连续的数字。它们用于区分服务站点下的服务器。注意，可以在任何给定的计算机上驻留多个服务器，每个服务器具有不同的<ServerID>。

<Address>

handle服务器的16字节IPv6[15,16]地址。任何IPv4地址都应该表示为::::FFFF:xxxx:xxxx(其中xxxx:xxxx可以是任何4字节的IPv4地址)。

<PublicKeyRecord>

一个4字节的整数，后跟一个包含服务器公钥的字节数组。该整数指定字节数组的大小。字节数组(用于公钥)由三部分组成:描述密钥类型的UTF8字符串、保留供将来使用的双字节选项字段和包含公钥本身的字节数组。例如，UTF8字符串“DSA\_PUB\_KEY”表示<PublicKeyRecord>包含一个DSA公钥。然后可以从Handle“0.type/DSA\_PUB\_KEY”找到字节数组中的DSA密钥的存储格式。<PublicKeyRecord>中的公钥可用于验证来自Handle服务器的任何服务响应。

<PublicKeyRecord>也可能包含一个X.509证书。如果键类型字段包含UTF8字符串“CERT.X509”，就会发生这种情况。在这种情况下，“CERT.X509”将映射到Handle“0.TYPE/CERT.X509”，这个Handle可以包含描述公钥或其证书的语法和语义的信息。还可以注册其他密钥类型(作为“0.TYPE”下的Handle)，以进一步区分不同类型的X.509证书。例如,“CERT.X509.DSA"可用于表示包含DSA公钥的X.509证书。如果<PublicKeyRecord>的密钥类型字段声明“CERT.X509”。<PublicKeyRecord>必须包含一个包含DSA公钥的X.509证书。

<ServiceInterface>::= <InterfaceCounter>

* [<ServiceType> <TransmissionProtocol> <PortNumber>]

一个4字节的整数，后跟一个由<ServiceType, TransmissionProtocol, PortNumber>组成的三元组数组。4字节整数指定三个一组的数目。每个三元组列出一个由Handle服务器提供的服务接口。对于每个三元组，<ServiceType>是一个字节(作为位掩码)，它指定接口是用于Handle解析(0x01)、Handle管理(0x02)，还是两者都是。<TransmissionProtocol>也是一个指定传输协议的字节(作为位掩码)。可能的传输协议包括TCP(0x01)、UDP(0x02)和HTTP(0x04)。<PortNumber>是一个4字节无符号整数，指定接口使用的端口号。默认端口号是2641。

图3.2.2显示了一个使用HS\_SITE值的Handle服务站点示例。HS\_SITE值被分配给命名权威Handle“0.NA/10”。<PrimaryMask>表明它是Handle服务的惟一主站点。该站点由三个Handle服务器组成，如<NumOfServer>所示。这些服务器为命名权威“10”下的每个Handle提供Handle解析和管理服务。第一个服务器记录(ServerID 0)显示两个服务接口，一个用于Handle解析，另一个用于Handle管理。每个接口都有自己的端口。

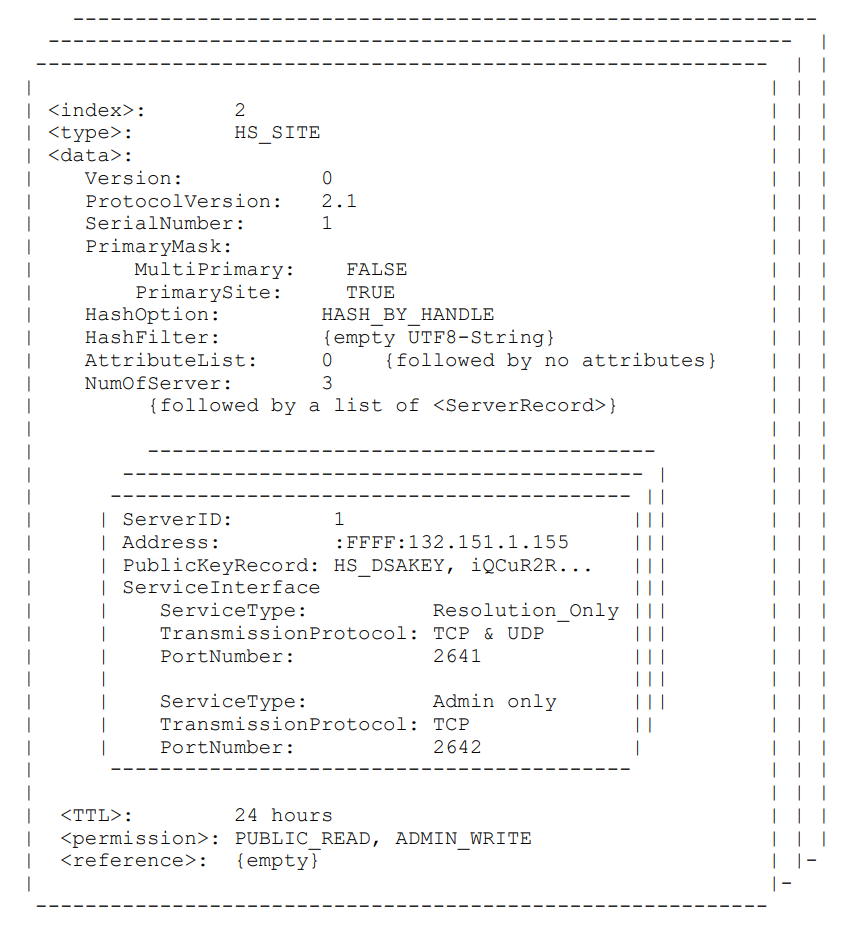


图3.2.2:命名权威“10”的主要服务站点

服务站点中的每个服务器负责Handle服务管理的一组Handle子集。客户机可以通过执行公共散列操作来查找负责任的服务器。哈希操作首先将Handle中的所有ASCII字符转换成大写。然后，它对转换后的Handle字符串的部分应用MD5散列(根据<HashOption>条目)。结果是一个16字节的整数。该整数的绝对值将除以服务器的数量(在<NumOfServer>条目中指定)。余数是HS\_SITE值中列出的<ServerRecord>的序列号(从零开始)。从<ServerRecord>，客户机可以为他们的Handle请求找到Handle服务器的IP地址。

### 3.2.3命名权威委托服务:HS\_NA\_DELEGATE

HS\_NA\_DELEGATE是一个预定义的Handle系统数据类型。它具有与HS\_SITE值完全相同的格式。与HS\_SITE值一样，HS\_NA\_DELEGATE值用于描述LHS的服务站点。

可以将HS\_NA\_DELEGATE值分配给命名权威Handle，以便将命名权威管理指定给LHS。带有一组HS\_NA\_DELEGATE值的命名权威Handle表示，该命名权威的所有子命名权威都由HS\_NA\_DELEGATE值所描述的LHS来管理。

例如，假设命名权威“foo.bar”决定将其子命名权限委托给LHS。为此，可以使用一组描述LHS的HS\_NA\_DELEGATE值分配给命名权威handle“0.NA/foo.bar”。HS\_NA\_DELEGATE值集表示“foo.bar”的任何子命名权威的服务信息，比如“foo.bar.baz”，可以从LHS通过查询命名权威handle“0.NA/foo.bar.baz”来找到。

### 3.2.4服务Handle:HS\_SERV

任何handle服务，不管是全局的还是本地的，都可以用一组HS\_SITE值来定义。这些HS\_SITE值可以直接分配给相关的命名权威handle，或者通过使用服务handle引入额外的间接级别。服务handle可以看作是handle服务的名称。它可用于维护handle服务的HS\_SITE值，并通过HS\_SERV值从命名权威handle引用。HS\_SERV值是一个handle值，其<type>字段是HS\_SERV，其<data>字段包含对服务handle的引用。HS\_SERV值通常分配给命名权威handle，以便将客户机引用到负责的handle服务。

使用服务Handle允许多个命名权威共享服务信息。它还允许在一个地方而不是在每个涉及的命名权威Handle中更改服务配置(例如，添加一个新站点)。该机制还可以用于支持出于任何原因将服务从一个Handle服务转到另一个Handle服务。

一个命名权威Handle最多只能分配一个HS\_SERV值，否则就是一个错误。如果命名权威Handle包含HS\_SITE值列表和HS\_SERV值，则HS\_SITE值应该用作命名权威的服务信息。

服务Handle可以在保留的命名权威“0.SERV”下注册。 “0.SERV”下的handle由GHR管理。例如，可以创建服务Handle“0.SERV/123”来维护在命名权威“123”（及其任何子命名权威）下管理Handle的Handle服务的服务信息。

类似地，可以创建服务Handle“0.SERV/a.b.c”来托管在“a.b.c”命名权威下管理Handle的Handle服务的服务信息。

使用服务Handle会引起一些特殊的考虑。由于缺乏效率，应该避免多级服务Handle重定向，但不应将其标记为错误。应该捕获指向不存在的服务Handle的服务Handle或HS\_SERV值的循环引用，并将错误条件传递回用户。

### 3.2.5别名Handle:HS\_ALIAS

在实践中，一个数字对象很可能有多个名称来标识该对象。Handle系统通过预定义的数据类型HS\_ALIAS支持这种特性。HS\_ALIAS值是一个Handle值，其<type>字段是HS\_ALIAS，其<data>字段包含对另一个Handle的引用。一个具有HS\_ALIAS值的Handle是HS\_ALIAS值中引用的Handle的别名Handle。别名Handle除了HS\_ALIAS或HS\_ADMIN(用于管理)值之外，不应具有任何其他Handle值。这对于防止Handle及其别名之间的任何不一致是必要的。

在Handle解析期间，客户机可能会返回一个HS\_ALIAS值，这表示所涉及的Handle是别名Handle。然后，客户机可以根据HS\_ALIAS值中指定的Handle重试查询，直到获得最终结果。

别名Handle的使用引入了一些特殊的注意事项。例如，为了提高效率，应该避免使用多级别名，但不应将其标记为错误。应该捕获指向不存在Handle的别名循环和别名，并将错误条件传递回用户。

别名Handle的一个潜在用途是支持任何命名资源的所有权转移。当由Handle标识的资源从一个组织转移到另一个组织时，可能会创建该资源的新Handle。为了避免不一致和任何不完整的引用，可以将所有权转移之前使用的Handle更改为别名Handle，并将其HS\_ALIAS值指向新创建的Handle。

### 3.2.6主要站点:HS\_PRIMARY

HS\_PRIMARY是一个预定义的数据类型，用于为任何给定Handle指定主要服务站点。具有多个主要服务站点的Handle服务称为多主要服务。否则，它被称为单一主服务。由多主Handle服务管理的每个Handle可以使用HS\_PRIMARY值指定其主服务站点。HS\_PRIMARY值是一个Handle值，其<type>字段是HS\_PRIMARY，其<data>字段包含对HS\_SITE值的引用列表。每个HS\_SITE都为Handle定义了一个主要的服务站点。

每个Handle最多可以分配一个HS\_PRIMARY值。否则就是一个错误。一个handle没有HS\_PRIMARY值但由多主Handle服务管理，这不是错误。在这种情况下，Handle服务的每个主要服务站点也将是Handle的主要站点。由单一主Handle服务管理的Handle不需要任何HS\_PRIMARY值，应该忽略任何此类值。

### 3.2.7 Handle值列表:HS\_VLIST

HS\_VLIST是一个预定义的数据类型，它允许将Handle值用作对其他Handle值列表的引用。HS\_VLIST值是一个Handle值，它的<type>是HS\_VLIST，它的<data>由一个4字节无符号整数组成，后面是对其他Handle值的引用列表。该整数指定列表中的引用数量。引用可以引用同一Handle下的Handle值，也可以引用来自任何其他Handle的Handle值。每个引用都被编码为一个UTF8字符串，后跟一个4字节无符号整数，用于标识引用Handle及其值索引。

HS\_VLIST值可用于为Handle定义管理员组。在本例中，HS\_VLIST中的每个引用定义管理员组的一个成员，HS\_VLIST值标识整个组。但是，客户机软件必须小心，避免循环定义值引用。

# Handle系统服务模型

Handle系统是一个分布式全局名称服务。它由一个分布式全局Handle注册表(GHR)和无限数量的本地Handle服务(LHS)组成。这些服务组件代表Handle系统客户机组件提供名称服务(包括解析和管理)。为了提高效率，Handle系统客户机组件也可以选择使用Handle系统中间件组件(例如，Handle系统缓存服务)。本节描述这些组件及其相互之间的关系。

## 4.1 Handle系统服务组件

Handle系统定义了一个层次服务模型。在顶层是单个分布式全局Handle服务，也称为全局Handle注册表(GHR)。在GHR下面，可以有任意数量的本地Handle服务(LHS)。每个LHS必须向GHR注册，以便在一组不同的命名权威下管理Handle。命名权威由GHR通过命名权威Handle(即，在命名权威“0.NA”下的handle)。还可以使用命名权威Handle来定位服务信息(以HS\_SITE值的形式)，该信息描述负责命名权威下Handle的Handle服务。从服务信息中，客户机可以选择一个服务站点，并为其handle请求找到负责任的服务器。

Handle系统服务组件是可扩展的，可以适应任何大量的服务负载。Handle服务(全局的或本地的)可以由多个服务站点组成，彼此复制。每个服务站点还可以由一组计算机组成，它们一起工作以服务各自的名称空间。拥有多个服务站点可以避免任何单点故障，并允许这些服务站点之间的负载平衡。在任何服务站点上使用多个服务器将服务负载分布到多个服务器进程中，并允许将低配的计算机用于名称服务。

### 4.4.1全局Handle注册表(GHR)

全局Handle注册表(GHR)主要用于管理命名权威Handle，并为Handle系统下的每个命名权威提供服务信息。GHR还可以用于管理和向非命名权威Handle提供解析和管理服务。与任何LHS不同的是，GHR主要用于注册命名权威，并为每个LHS提供服务信息。换句话说，GHR是注册每个LHS并通过使用命名权威Handle提供其服务信息的单一根服务。Handle系统下的每个命名权威都必须作为命名权威Handle在GHR下注册。命名权威Handle提供管理命名权威下所有Handle的Handle服务的服务信息。服务信息可以按照一组HS\_SITE值提供，或者按照前面描述的引用服务Handle的HS\_SERV值提供。

GHR可能由多个服务站点组成，每个服务站点在一个HS\_SITE值中进行描述。这些HS\_SITE值被分配给指定的命名权威Handle“0.NA/0.NA”，也叫**根Handle**。根Handle是维护GHR服务信息的命名权威Handle。**顶级命名权威只能由根Handle的管理员创建**。

客户机软件为了与GHR进行通信，需要事先获得GHR服务信息。服务信息可以最初与客户机软件一起发布，也可以从其他安全来源(例如，邮政局邮件、安全网站等)获得。客户机软件可以保留服务信息以与GHR通信，直到服务信息过期(根据其TTL)。GHR必须在每次更改配置时更新其服务信息(分配给根Handle)。带有过时服务信息的客户机软件每次与GHR通信时都将收到更新通知。GHR必须以这样一种方式维护，即任何具有过时GHR服务信息的客户机软件仍然可以查询根Handle以获得最新更新。

图4.1.1以一组HS\_SITE值表示GHR服务信息。GHR可能由许多服务站点组成，每个服务站点在HS\_SITE值中进行描述。图中显示了位于美国东海岸的GHR服务站点，如<AttributeList>所示。

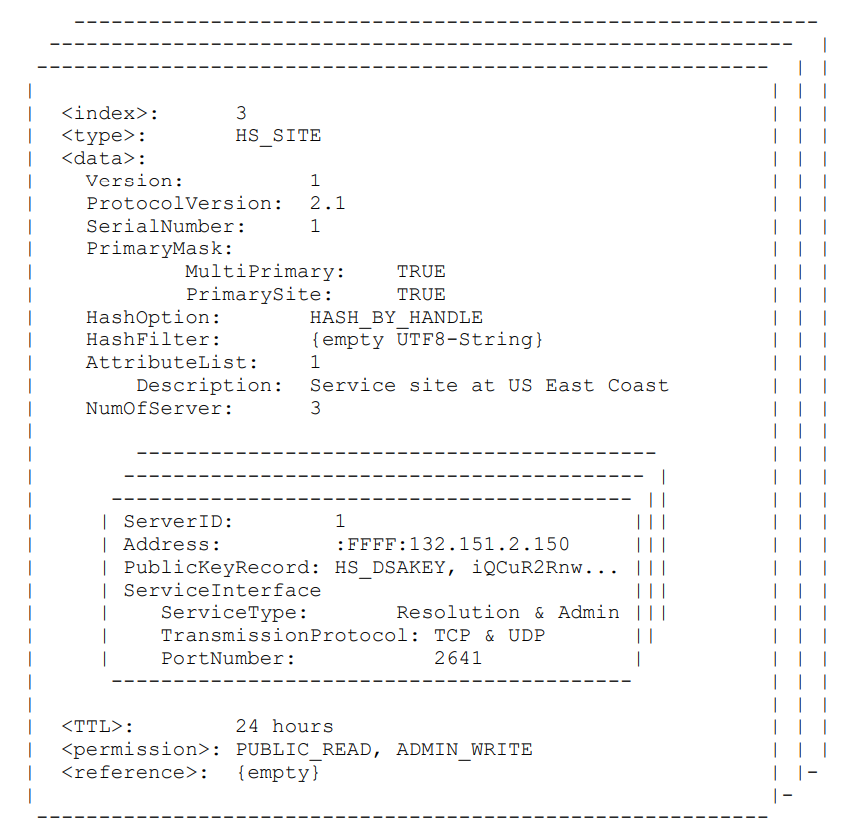


图4.1.1:GHR服务信息

GHR及其服务信息为任何客户机软件与Handle系统通信提供了一个入口点。对于任何给定Handle，客户机软件都可以查询GHR的命名权威Handle。这将返回在命名权威下管理每个Handle的LHS的服务信息。服务信息将客户机软件指向管理Handle的LHS中的Handle服务器。

### 4.1.2本地Handle服务(LHS)

本地Handle服务(LHS)根据给定的命名权威集管理Handle。每个命名权威定义一个“本地”名称空间，该名称空间由命名权威下的所有Handle组成。注意，就任何网络拓扑而言，LHS都不是“本地”服务。它被称为“本地”Handle服务，因为它通常管理一个受限制的(本地)名称空间。

如果一个命名权威下的所有Handle都由某LHS管理，则这个命名权威“安家”在该LHS。LHS可能是多个命名权威的家。另一方面，命名权威只能在一个LHS处“安家”。请注意，命名权威也可以位于GHR。

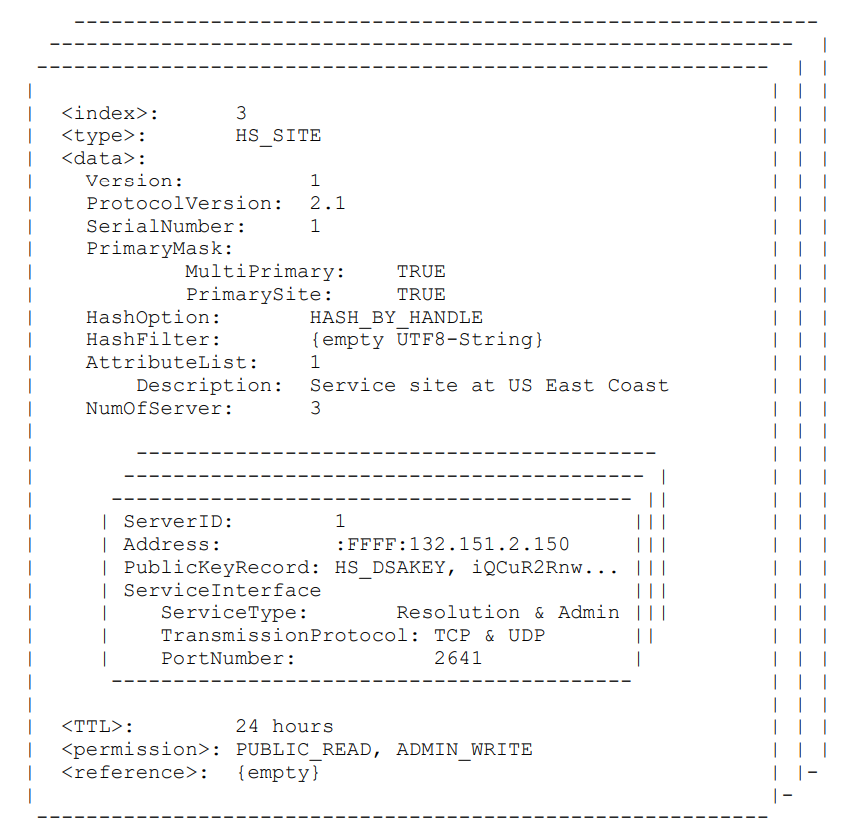


图4.1.2:LHS服务信息

与GHR一样，LHS也可能由许多服务站点组成，每个站点由一个HS\_SITE值描述。任何LHS的HS\_SITE值集都可以分配给服务Handle或相关的命名权威Handle。图4.1.2显示了LHS的HS\_SITE值示例。这些HS\_SITE值被分配给命名权威Handle“0.NA/10”。这表明，命名权威“10”“安家”在这些HS\_SITE值中指定的LHS上。客户可以通过查询GHR来获取服务信息，以便与LHS进行通信。命名权威Handle的管理员负责维护服务信息并使其保持最新。

请注意，LHS可以将客户机服务请求的响应转到另一个LHS。这允许LHS以分层的方式进一步分发其服务。

## 4.2 Handle系统中间件

Handle系统中间件目前包括Handle系统缓存服务器和Handle系统代理服务器。这些Handle系统中间件组件是Handle系统服务组件的客户机，是Handle系统客户机软件的服务器。Handle系统中间件用于为基本Handle服务提供附加接口。例如，Handle系统缓存服务器可用于在本地社区中共享解析结果。此外，可以使用Handle系统代理服务器通过HTTP隧道绕过任何组织防火墙。

### 4.2.1准备 Handle系统缓存服务

Handle系统缓存服务可用于减少Handle系统客户机和服务器之间的网络流量。缓存Handle数据，包括任何LHS的服务信息，允许重用从早期查询中获得的信息。

每个Handle值都包含一个<TTL> (Time to Live)字段，该字段告诉缓存服务缓存的值可以有效多长时间。零值TTL表示该值只能用于正在进行的事务，不应该被缓存。缓存服务可以直接从Handle服务获取数据，也可以从最终从Handle服务获取数据的另一个缓存服务获取数据。

缓存服务可以根据HS\_SITE值定义，并且可以由多个缓存服务器组成。对于任何给定的Handle，客户机可以在缓存服务中找到负责的缓存服务器，使用在Handle服务中定位Handle服务器的相同散列算法。

缓存服务不是任何Handle系统管理或身份验证层次结构的一部分。Handle系统协议不验证来自缓存服务的任何响应。客户机负责与他们选择的缓存服务建立信任关系。它们还将依赖缓存服务正确地验证来自任何Handle服务器的任何响应。

### 4.2.2 Handle系统代理服务器

Handle系统代理服务器可用于通过其他互联网协议启用Handle解析。例如，CNRI已经构建并提供了一个Handle系统HTTP代理服务器，它将处理HTTP协议中的任何Handle解析。这个代理服务器的当前DNS地址是“hdl.handle.net”。代理服务器允许通过HTTP URL解析任何Handle。URL可以构造为“http://hdl.handle.net/<handle>”，其中<handle>可以是Handle系统中的任何Handle。例如，Handle“ncstrl.vatech\_cs/tr-93-35"可以通过HTTP URL "http://hdl.handle.net/ncstrl.vatech\_cs/tr-93-35"从任何web浏览器解析。在本例中，URL以HTTP请求的形式发送到代理服务器。代理服务器将查询Handle数据的Handle系统，并根据HTTP响应返回结果。

使用HTTP URL允许从标准web浏览器解析Handle，而不需要任何附加的客户机软件。但是，对Handle的引用也将自己绑定到代理服务器。如果代理服务器更改了其DNS名称或以其他方式变得无效，则引用(即，到Handle的HTTP URL)将中断。因此，应该仔细评估代理服务器的选择或使用。

代理服务器不属于任何Handle系统管理或身份验证层次结构的一部分。Handle系统协议不验证来自代理服务器的任何响应。客户机负责与他们选择的代理服务器建立信任关系。它们还将依赖代理服务器正确地验证来自任何Handle服务器的任何响应。

## 4.3 Handle系统客户机组件

Handle系统客户机组件是与Handle系统服务组件通信的客户机软件。客户机软件可以使用Handle系统协议，并将其请求直接发送给服务组件。来自服务组件的响应可能是请求的最终答案，也可能是对另一个服务组件的引用。为了完成事务，客户机软件将不得不遵循该引用。

客户机软件也可以配置为通过一个中间件组件来隧道其请求。因此，中间件组件将负责获取最终结果并将其返回给客户机。与服务组件不同，中间件组件只返回客户机请求的最终结果。不会从中间件组件返回任何服务引用。

可以为各种应用程序开发各种Handle系统客户机组件。CNRIHandle系统分解器[17]就是这样一个组件。解析器扩展了web浏览器(例如Netscape或Microsoft IE)，其处理方式可以直接用“hdl:”统一资源标识符(URI)解析。Grail web浏览器[18]是用Python[19]开发的一个可免费下载的软件，它也支持“hdl:”URI模式，并将相应地解析Handle。例如，Handle“10.1045/july95-arms”可以通过将其HandleURI作为“hdl:10.1045/july95-arms”输入到这些支持解析器的浏览器中来解析。HandleURI语法的详细信息将在单独的文档中指定。

# Handle系统操作模式

Handle系统操作可以分为解析和管理。客户机使用Handle解析服务来查询任何Handle值。Handle管理允许客户机管理Handle，包括添加和删除Handle，以及更新它们的值。它还通过命名权威Handle处理命名权威管理。本节解释各种Handle系统组件如何协同工作以完成这些服务操作。

解析和管理都可能需要客户机的身份验证。可以通过本节后面介绍的Handle系统身份验证协议进行身份验证。是否需要身份验证取决于所涉及的操作类型、分配给相关Handle值的权限以及相关服务组件部署的策略。

Handle系统协议指定Handle系统客户机及其服务器组件之间交换的每个消息的语法和语义。本节提供用于完成任何服务操作的协议的高级概述。每个消息的精确编程细节(即它们的字节设计或语法)在单独的文档[8]中详细说明。

## 5.1 Handle系统服务请求和响应

Handle系统为响应客户机请求提供服务。客户机可以向任何Handle服务器发送请求来触发响应。响应要么提供对请求的回答，要么提供包含相关信息的状态代码，这些信息或将请求引用到另一个服务组件，或请求客户机身份验证，或发出一些错误状态信号。

Handle系统下的每个Handle都由其本地服务管理。命名权威Handle提供在命名权威下管理所有Handle的Handle服务的服务信息(以HS\_SERV或HS\_SITE值表示)。任何Handle请求都必须指向相关Handle的本地服务。客户机可以根据GHR查询相应的命名权威Handle来找到home服务。或者，可以在本地缓存中找到这些信息，甚至可以将其作为本地客户机配置的一部分。给定服务信息，客户机可以选择一个服务站点，并在该站点中找到负责的Handle服务器。

例如解析Handle“ncstrl.vatech\_cs/te-93-35”，客户机软件需要知道命名权威“ncstrl.vatech\_cs”的home服务。home服务可以通过从GHR查询命名权威Handle“0.NA/ncstrl.vatech\_cs”来获得。GHR将根据分配给命名权威Handle的HS\_SITE值返回服务信息。从服务信息中，客户机可以选择一个服务站点，在站点中找到负责的Handle服务器，并将解析请求发送到Handle服务器。

为了验证来自服务器的任何响应，客户机可能需要来自Handle服务器的数字签名。可以使用服务器的私钥生成签名。客户机可以使用来自服务信息的公钥来验证签名(参考3.2.2中讨论的<PublicKeyRecord>条目)。

还可以在任何客户机和Handle服务器之间建立通信会话。每个会话由服务器管理的惟一会话ID标识。会话可用于管理需要多个交互的请求。它还可以用于在多个服务事务之间共享任何TCP连接或身份验证信息。每个会话可以建立一个会话密钥，并使用它对会话内交换的任何消息进行身份验证。它还可以用于加密客户机和服务器之间的任何消息，以实现数据机密性。

下面的关系图显示了基于客户机软件和Handle系统服务组件之间交换的消息的Handle解析过程。在本例中，客户机试图解析Handle“ncstrl.vatech\_cs/tr-93-35”。它假定客户机尚未获得由命名权威“ncstrl.vatech.cs”提供的LHS“到家”服务信息。客户机必须从GHR管理的命名权威Handle获取服务信息。服务信息允许客户机找到负责的LHS并查询Handle值。

(HS客户机)- - - - - - - - - - - - - - - - - - - ->(全局Handle注册表)

1. 向命名权威Handle“0.NA/ncstrl.vatech\_cs”请求服务信息

(HS客户机)<- - - - - - - - - - - - - - - - - - -(全局Handle注册表)

1. 命名权威“ncstrl.vatech\_cs”的服务信息

(HS客户机)- - - - - - - - - - - - - - - - - - - - ->(本地Handle服务)

1. 对负责的handle服务器查询Handle“ncstrl.vatech\_cs/tr-93-35"

\……

(可选的客户机身份验证，取决于服务请求)

\……

(HS客户机)<- - - - - - - - - - - - - - - - - - - -(本地Handle服务)

1. 来自Handle服务器+(可选)服务器签名的查询结果

图5.1: Handle解析示例

在图5.1中，对于任意handle服务，客户机被配置为与GHR通信。在本例中，客户机首先查询GHR来查找Handle的命名权威的home服务。GHR返回管理命名权威下每个Handle的LHS的服务信息。从服务信息中，客户机可以找到负责的Handle服务器，并向服务器查询Handle。如果Handle值中的任何一个需要身份验证，服务器可以设置一个会话来对客户机进行身份验证。否则，服务器将简单地将Handle值返回给客户机。如果客户机需要，服务器可以发送数字签名作为其响应的一部分。

上述过程假设客户机软件已经拥有GHR服务信息。这些信息可能是从客户机软件发行版获得的。GHR将通知客户机软件，如果它知道客户机软件使用的服务信息已经过期。客户机软件可以从根Handle“0.NA/0.NA”中检索最新的服务信息。根Handle还维护着可用于验证服务信息的公钥。

请注意，客户机可以缓存任何命名权威的服务信息，以便在同一命名权威下对Handle的后续查询可以重用服务信息，并绕过图5.1中所示的前两个步骤。客户机软件也可以配置为直接查询缓存或代理服务器的任何Handle。在这种情况下，缓存或代理服务器将充当图5.1中的[HS客户机]，然后将查询结果返回给客户机。

某些组织下的客户机软件也可以选择绕过GHR，直接与组织管理的LHS通信。这样做可以更快地响应在LHS下管理的Handle。对于非LHS管理的handle，客户机软件将被转交给GHR。

## 5.2 Handle系统认证协议

Handle系统支持公共互联网上的Handle管理。可以在每个Handle值上定义访问控制。Handle系统身份验证协议是任何Handle服务器在任何管理请求时用来对Handle管理员进行身份验证的协议。当客户机查询Handle管理员只读的Handle值时，身份验证也是必要的。Handle管理包括添加、删除或修改Handle值，以及添加或删除Handle。命名权威管理转化为（对应到命名权威的）Handle管理。

Handle系统身份验证协议不执行任何服务器身份验证（译者：指没有类似TLS的服务器认证机制）。但是，客户机可以通过要求服务器使用数字签名来对任何服务器响应进行身份验证。

默认情况下，Handle系统通过挑战-响应协议对客户机进行身份验证。也就是说，在接收到客户机请求之后，如果需要身份验证，服务器将向客户机发出挑战。要被认证为管理员，客户机必须返回一个挑战-响应，这是一个证明管理员的秘密处理过程的消息。秘密可以是管理员的私钥，也可以是管理员的私钥。此挑战-响应允许服务器将客户机身份验证为Handle管理员。成功的身份验证之后，如果授予管理员足够的权限，服务器将满足客户机的请求。

例如，假设一个客户机向Handle服务器发送一个请求来添加一个新的Handle值。服务器将向客户机发出一个挑战，以便对客户机进行身份验证，使其作为Handle管理员。如果客户机拥有管理员的私钥，则可以使用它对服务器的挑战进行签名，并返回签名作为挑战响应的一部分。服务器将验证签名以便对客户机进行身份验证。如果验证失败，将通知客户机。否则，服务器将进一步检查管理员是否具有添加Handle值的权限。如果是，服务器将添加Handle值并向客户机报告成功。否则，将返回一个拒绝访问的消息。

下面的关系图显示了一个典型的身份验证过程，它涉及客户机和Handle服务器之间交换的消息。

[客户机] --------------------------------> [Handle服务器]

1. 客户机请求

+ (可选) 客户机凭据

[客户机] <-------------------------------- [Handle服务器]

2. 服务器对客户机的挑战

+ (即， nonce + 客户机请求的MD5)

[客户机] -------------------------------> [Handle服务器]

3. 引用handle管理员

+ 从客户机的挑战-响应

[客户机] <------------------------------- [Handle服务器]

4. 服务器确认

图5.2:Handle系统认证过程

在图5.2中，客户机向Handle服务器发送管理请求(以及后面讨论的可选凭据)。服务器决定需要客户机身份验证，并向客户机发出挑战。客户机将自己标识为Handle管理员，并将挑战-响应返回给服务器。服务器根据挑战-响应将客户机身份验证为管理员。它还检查管理员是否被授权进行管理请求。如果是，服务器将完成请求并确认客户机。

在满足任何需要管理员特权的请求之前，Handle服务器必须对客户机进行身份验证。确切的身份验证过程取决于管理员使用的是公钥还是密钥。它还取决于用于存储管理员密钥的Handle是否由相同的Handle服务器管理。

当使用公钥时，来自客户机的挑战响应包含其对服务器挑战的数字签名。服务器可以通过验证基于管理员公钥的数字签名来对客户机进行身份验证。如果使用了秘密密钥，来自客户机的挑战-响应将携带使用秘密密钥生成的消息身份验证代码(MAC)。服务器可以使用管理员的密钥生成相同的MAC，并将其与挑战-响应进行比较，从而对客户机进行身份验证。

图5.2中对handle管理员的引用也称为密钥引用。它引用了一个包含管理员使用密钥的Handle值。如果密钥引用由同一Handle服务器管理(例如，将Handle值分配给同一Handle)，则服务器可以直接使用该密钥进行身份验证。如果密钥引用是由另一个Handle服务器管理的(不管是否在同一个Handle服务中)，服务器将必须向另一个Handle服务器发送验证请求，将其称为密钥服务器，以便对客户机进行身份验证。对键服务器的验证请求包含服务器的挑战和客户机的挑战响应。密钥服务器将返回一个验证响应，使用密钥服务器的私钥进行签名。验证响应的内容将取决于密钥引用引用的Handle值。如果密钥引用引用管理员使用的公钥，则验证响应将包含管理员的公钥。否则，密钥服务器将代表请求服务器验证挑战-响应，并在验证-响应中返回结果。下图显示了身份验证过程的控制流，其中密钥引用引用包含管理员的公共(或秘密)密钥的Handle值，而密钥服务器是其他一些Handle服务器。

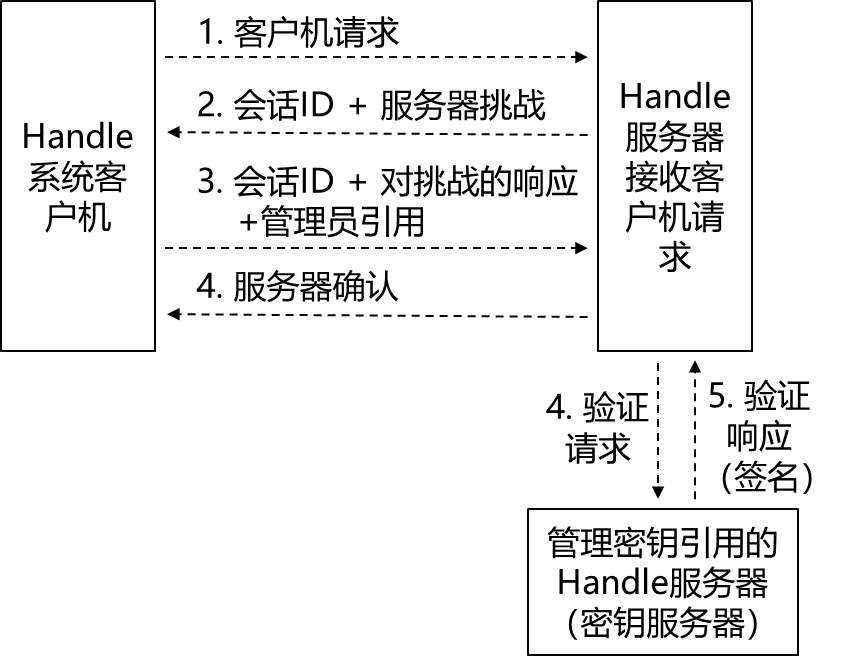


图5.3:验证过程需要来自第二个Handle服务器的验证

基于密钥的身份验证通过第二个Handle服务器，即密钥服务器提供了一种方便的方法，可以在由不同的Handle服务器管理的Handle之间共享公共密钥(例如，传递短语)。但是，它不应该用于管理高度敏感的handle或handle数据。身份验证过程本身是昂贵的，并且依赖于第三方（密钥服务器），以进行正确的操作。此外，密钥本身也会受到字典攻击，因为密钥服务器无法确定验证请求是否来自合法的Handle服务器。Handle服务可以设置它的本地策略，因此只有在Handle服务器(接收客户机请求)也是密钥服务器的情况下才能执行基于密钥的身份验证。

本地Handle服务可以为身份验证和/或授权定义附加的本地策略。Handle系统服务组件还可以选择使用其他互联网身份验证机制，如Kerberos[20]或某些传输层安全协议[21]。这些细节将在另一份文档中讨论。

# 安全注意事项

Handle系统安全性注意事项在“Handle系统概述”[1]中进行了讨论，该讨论同样适用于本文档。

Handle系统将Handle管理委托给每个Handle管理员，这些管理员可能是服务器管理员，也可能不是。Handle管理员可以选择自己用于身份验证的公钥/密钥。Handle系统认证的安全性取决于Handle管理员对密钥的正确选择和维护。为了保护Handle数据，Handle管理员必须仔细选择和保护自己的认证密钥。Handle服务器实现可能会部署一些策略来规范用于身份验证的公钥/密钥的选择。例如，Handle服务器可能要求任何身份验证密钥必须不少于一定数量的位数。它还可能禁止使用秘密密钥（secret key），因为存在潜在的字典攻击。

Handle系统数据模型支持每个Handle值的执行权限(PUBLIC\_EXECUTE、ADMIN\_EXECUTE)。虽然这样可以更好地共享网络资源，但也带来了许多安全问题。应该将执行权限限制在服务器上某个用户帐户(对应于Handle管理员)的权限内，以防止系统崩溃。在服务器的计算平台之间切换也应该小心，以避免任何意外的行为。实现可以选择不支持执行权限，或者提供选项以禁用它。

为了防止不负责任地使用系统资源，handle服务器可以实现配额控制。配额控制可用于限制命名权威下的Handle数量、任何给定Handle允许的Handle值数量、任何Handle值的最大大小以及命名权威下的子命名权威数量。如果Handle管理的结果违反了这些限制，则Handle服务器必须报告错误。

# 感谢

这项工作源自Handle系统实现的早期版本。1995年，Robert Kahn和Robert Wilensky[22]在一篇论文中描述了整个数字对象体系结构，包括Handle系统。发展在CNRI继续，作为计算机科学技术报告(CSTR)项目的一部分，由美国国防部高级项目局(DARPA)资助，资助号为MDA-972-92-J-1029和MDA-972-99-1-0018。设计思想基于Handle系统开发团队中的讨论，包括David Ely、Charles Orth、Allison Yu、Sean Reilly、Jane Euler、Catherine Rey、Stephanie Nguyen、Jason Petrone和Helen She。我们感谢他们对这项工作的贡献。

作者还感谢Russ Housley (housley@vigilsec.com)、Ted Hardie (hardie@qualcomm.com)和Mark Baugher (mbaugher@cisco.com)的广泛评论，以及来自IETF/IRTF社区其他成员的建议。

# 引用和参考书目

[1] Sun, S. and L. Lannom, "Handle System Overview", RFC 3650,

November 2003.

[2] Mockapetris, P., "Domain Names - Concepts and Facilities," STD

13, RFC 1034, November 1987.

[3] Mockapetris, P., "Domain Names - Implementation and

Specification", STD 13, RFC 1035, November 1987.

[4] Wahl, M., Howes, T. and S. Kille, "Lightweight Directory Access

Protocol (v3)", RFC 2251, December 1997.

[5] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax

Specifications: ABNF", RFC 2234, November 1997.

[6] Yergeau, F., "UTF-8, A Transform Format for Unicode and

ISO10646", RFC 2279, January 1998.

[7] The Unicode Consortium, "The Unicode Standard, Version 2.0",

Addison-Wesley Developers Press, 1996. ISBN 0-201-48345-9

[8] Sun, S., Reilly, S. and L. Lannom, "Handle System Protocol (ver

2.1) Specification", RFC 3652, November 2003.

[9] Berners-Lee, T., Masinter, L. and M. McCahill, "Uniform Resource

Locators (URL)", RFC 1738, December 1994.

Sun, et al. Informational [Page 38]

RFC 3651 Handle System Service Definition November 2003

[10] Housley, R., Polk, W. Ford, W. and D. Solo, "Internet X.509

Public Key Infrastructure - Certificate and Certificate

Revocation List (CRL) Profile", RFC 3280, April 2002.

[11] Federal Information Processing Standards Publication (FIPS PUB)

46-1, Data Encryption Standard, Reaffirmed 1988 January 22

(supersedes FIPS PUB 46, 1977 January 15).

[12] Federal Information Processing Standards Publication (FIPS PUB)

81, DES Modes of Operation, 1980 December 2.

[13] Balenson, D., "Privacy Enhancement for Internet Electronic Mail:

Part III: Algorithms, Modes, and Identifiers", RFC 1423,

February 1993.

[14] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April

1992.

[15] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6)

Specification", RFC 1883, December 1995.

[16] Hinden, R. and S. Deering, "IP Version 6 Addressing

Architecture", RFC 2373, July 1998.

[17] CNRI Handle System Resolver, http://www.handle.net/resolver

[18] Grail browser home page, http://grail.sourceforge.net/

[19] Python language website, http://www.python.org/

[20] Kohl, J. and C. Neuman, "The Kerberos Network Authentication

Service (V5)", RFC 1510, September 1993.

[21] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC

2246, January 1999.

[22] R. Kahn, R. Wilensky, "A Framework for Distributed Digital

Object Services, May 1995, http://www.cnri.reston.va.us/k-w.html

[23] American National Standards Institute. ANSI X9.52-1998, Triple

Data Encryption Algorithm Modes of Operation. 1998.

# 作者的地址

Sam X. Sun

Corporation for National Research Initiatives (CNRI)

1895 Preston White Dr., Suite 100

Reston, VA 20191

Phone: 703-262-5316

EMail: ssun@cnri.reston.va.us

Sean Reilly

Corporation for National Research Initiatives (CNRI)

1895 Preston White Dr., Suite 100

Reston, VA 20191

Phone: 703-620-8990

EMail: sreilly@cnri.reston.va.us

Larry Lannom

Corporation for National Research Initiatives (CNRI)

1895 Preston White Dr., Suite 100

Reston, VA 20191

Phone: 703-620-8990

EMail: llannom@cnri.reston.va.us

# 完整的版权声明

版权所有(C)互联网协会(2003)。 保留所有权利。

这个文档和翻译的可以被复制和提供给他人,和衍生著作、评论或者解释或者协助其实现可能准备,复制,发布和分发,在全部或部分,没有任何形式的限制,前提是上面的版权声明和本段包括在所有此类副本和衍生著作。然而,这个文档本身可能不会以任何方式修改,如通过删除版权通知或引用互联网协会或其他网络组织,除所需的目的为版权开发互联网标准在这种情况下,程序中定义的过程必须遵循网络标准,或根据需要翻译成除英语之外的其他语言。

上述所授予的有限权限是永久的，互联网协会及其继任者或受让人不会撤销上述权限。

这个文档,此处包含的信息是“AS IS”的基础上提供的互联网协会和互联网工程任务组并不保证,明示或暗示,包括但不限于任何保证,使用文中的信息不会侵犯任何权利或任何隐含保证适销性或健身为特定目的。

感谢

RFC编辑器功能的资金目前由互联网协会提供。

译者：王伟兵