

第9章 网络信息系统

在运行一个局域网时，你的主要任务通常是为你的用户提供一个友好、透明的网络环境。最重要的一步是实时保存各台主机的用户账号信息之类的大量数据。在接触主机名解析之前，我们还介绍过一个强大而复杂的服务，那就是 DNS。对其他任务来说，还没有这类特殊服务。而且，如果管理的只是一个没有接入因特网的小型局域网，对许多管理员来说，都觉得没必要费太多脑筋去设置 DNS。

这就是 Sun 微系统开发 NIS（网络信息系统）的原因。NIS 提供了常见的数据库访问设备，可用于将信息分发到网络上的各台主机，这些信息原本包含在 `passwd` 和 `groups` 文件中。这样一来，整个网络就像一个独立的系统一样，所有主机的账号都是一样的。以类似的方式，还可使用 NIS 将主机名信息从 `/etc/hosts` 分发给网络上的各台主机。

NIS 是建立在 RPC 基础上的，由一个服务器、一个客户端库和若干个管理工具构成。起初，NIS 叫做“Yellow Page”（黄页）或 YP，至今，这个称呼仍然非常普遍。另一方面，“Yellow Page”是英国电信的注册商标，他们要求 Sun 放弃这个名字。但人们仍然记得最初的名字，YP 仍然作为多数 NIS 命令名的前缀广为流传，比如 `ypserv`、`ypbind` 等等。

如今，有一些免费的 NIS 实施方案。其中之一来自 BSD Net-2，衍生于 Sun 免费发放的公用域参考实施方案。长期以来，它发布的库客户机程序一直包含在 `GNUlibc` 内，而管理程序近来才水落石出，是 Swen Thümmeler 移植过来的（邮件地址 `swen@uni-paderborn.de`）。这些 NIS 客户机程序可用作 `yp-linux.tar.gz`，后者源于 `system/Network` 内的 `sunsite.unc.edu`。NIS 服务器没有包括在参考实施方案内。Tobias Rebert 编写了另一个 NIS 包，其中包含所有的工具和一个服务器；该 NIS 包名为 `yps`（当前版本是 `yps-0.21`，可从 `/pub/NYS` 目录下的 `ftp.lysator.liu.se` 获得）。

目前，完全重写 NIS 代码的 NYS 代码正由 Peter Eriksson（邮件地址 `pen@lysator.liu.se`）负责编写，它将同时提供对普通 NIS 和 Sun 修订过若干次的 NIS+ 的支持。NYS 不止提供一个 NIS 工具集和一个服务器，还将增加一个全新的库函数集，后者最终将 NIS 变成一个标准的 `libc`。NYS 中包括了一个用于主机名解析新配置方案，将利用 `host.conf` 替换当前采用的方案。其中各个函数的特性将在随后的小节中讨论。

本章的重点在于 NYS，而不是另外传统意义上的 NIS 程序包。如果想运行这些包，本章内容并不充分。要想获得更多的详情，可参考一本关于 NIS 的书，比如 Hal Stern 所著的《NFS 和 NIS》，或者查看 `howto` 文件，该文件位于 `www.suse.de/~kukuk/linux/HOWTO/NIS-HOWTO.html`。

现在，NYS 仍处于开发阶段，所以诸如网络程序或登录程序之类的标准实用程序还不能识别 NYS 配置方案。直到有一天，NYS 揉合到主流 `libc` 中，你希望自己所有的二进制程序使用 NYS 配置方案时，才有必要对这些二进制程序进行重新编译。这类程序的 `Makefiles` 会将 `libc` 前的最后一个选项 `-lnsl` 指定为连接程序。这样便链接到 `libnsl` 的相关函数和 NYS 库，而不是标准的 C 语言库。

9.1 NIS概述

NIS将数据库信息保存在自己的所谓映射表内，该映射表内包含的是成对出现的关键字 - 值。映射信息保存在运行NIS服务器的中央主机内。客户机可以通过各种RPC调用，获取中央主机内的信息。一般说来，映射是保存在DBM文件内的（DBM是一个简单的数据库管理库，利用散列技术加快搜索操作。GNU工程组曾发布一个免费的DBM实施，名为gdbm，许多Linux厂商都将其包含在自己的产品内）。

映射本身一般源于主管文本文件，比如/etc/hosts和/etc/passwd。对于有的文件，可创建好几个映射，一个映射代表一类搜索关键字。例如，可以在搜索IP地址的同时，在hosts文件内搜索主机名。相应地，就会从这个hosts文件内创建两个NIS映射，分别是hosts.byname和hosts.byaddr。

在某些NIS包或其他地方，还可找到相应的其他文件和映射。其中包含的应用程序信息本书没有讨论，比如某些BOOTP服务器所用的bootparams映射，或当前还没有排上用场的映射，如ethers.byname和ethers.byaddr映射。

对于有些映射，人们常采用nickname（绰号）来表示它们，这是因为绰号更短，更方便键入。要想得到一份你的NIS工具能识别的绰号列表（以Red Hat 6为例），运行下面的命令即可：

```
$ cat /var/yp/nickname
```

过去，NIS服务器被称为ypserv。对一个中等大小的网络来说，一台服务器就能够应付了；大型网络可能会在不同的机器上，不同的网络分段上运行若干个服务器，减轻服务器上路由器的负荷。这类服务器的同步更新是通过令其中一台服务器为主管（master）服务器，其他为从属（slave）服务器来完成的。只有主管服务器主机才创建映射。建好之后，再分发到所有的从属服务器。

大家将注意到，我们一直在谈的“网络”一词非常含糊；诚然，NIS中，这一词指的是一个通过NIS，共享系统配置数据的所有主机的集合：即NIS域。但令人遗憾的是，NIS域和我们在DNS域内见过的东西没什么两样。为尽可能使本章明晰可辨，我将着重指出我所指的域究竟是什么。

NIS域只有一个纯管理功能。除了该域内的所有主机共享密码之外，多数时候，用户都不能看到该功能的体现。因此，为NIS域指定的域名只和管理员有关。一般情况下，任何名字都是允许的，只要不和本地网络内的其他NIS域同名就行。例如，Virtual Brewery网络管理员创建了两个NIS域，一个供Brewery自己用，一个供Winery用，她将它们分别命名为brewery和winery。另一个相当常见的方案是简单地利用DNS域名来代表NIS域名。要想设定和显示主机的NIS域名，可利用domainname命令。在不带参数的情况下，它的调用结果是当前的NIS域名；要想设置域名，必须先成为超级用户，然后再键入：

```
# domainname <YOURNISDOMAIN>
```

NIS域判断应用程序将查询哪个NIS服务器。比如，以Winery上的一台主机为例，它的登录程序只能查询Winery的NIS服务器（如果该网络有若干台服务器的话，则是其中之一），要求得到用户的密码信息；Brewery上的主机同样如此，只能查询自己的服务器。

还有一点疑问有待解决，那就是客户机如何查找准备与之连接的服务器。最简单的方法

是能够有一个配置文件，其中将客户机指定与某台服务器连接。但这种方法非常不灵活，因为它不允许客户机使用别的服务器（当然指的是源于同一个域的那些）。因此，传统的NIS实施依赖于一个名为 ypbind 的特殊 daemon（程序），通过它，在NIS域内侦测合适的NIS服务器。在能够执行任何 NIS 查询之前，所有应用程序都应该先从 ypbind 中找出准备使用的服务器。ypbind 通过向本地 IP 网络广播查询，查出自己需要的服务器，第一个应答的服务器将被定为速度最快的，并将用于后续的所有 NIS 查询中。特定时间间隔之后，或服务器不能使用之时，ypbind 将再次查找活动的服务器。

现在，关于动态绑定，其有待考证的是其用途少，而且还可能引发安全问题：ypbind 盲目地相信任何人的回答，无论对方是低级的 NIS 服务器，还是心怀不轨的入侵者。需要提醒大家注意的是，如果你通过 NIS 管理自己的密码数据库，这种方法会为你带来许多麻烦。因此，默认情况下，NIS 是不会采用 ypbind 的，而是从配置文件内选出服务器的主机名。

9.2 NIS与NIS+之比较

除了名字和常见用法相同外，很难再找出 NIS 和 NIS+ 的相同点了。NIS+ 采用了另一种截然不同的结构。原先带有分散 NIS 域的单调的域名空间，被一个类似于 DNS 的分层式域名空间代替。原先的映射，被所谓的表格代替，这类表由列和行组成，一行代表 NIS+ 数据库内的一个对象，而列代表 NIS+ 已知的并关心的对象之属性。针对具体 NIS+ 域的表由其父域的表组成。另外，表中的条目还可能包含一条指向另一个表的链接。这些特性使得我们可以采用不同的方式构建信息。

传统 NIS 的 RPC 版本号是 2，NIS+ 则是 3。

目前，NIS+ 的使用似乎不太广泛，我本人对它也知之甚少（坦白说是一无所知）。鉴于此，我们打算对它进行讨论，如果你对它有兴趣，不妨去看看它的 howto 文档，可在 www.suse.de/~kukuk/linux/HOWTO/NIS/NIS-HOWTO.html 找到它。

9.3 NIS的客户端

如果你熟谙网络应用程序的编写和移植，肯定会注意到上面列出的许多 NIS 映射都对应于 C 语言库内的库函数。例如，要想获得 passwd 信息，一般会采用 getpwnam(3) 和 getpwuid(3) 这两个函数，它们将分别返回与具体用户名或数字化的用户 ID 关联的账号信息。正常情况下，这些函数将对标准文件，比如 /etc/passwd 执行所要求的查找。

但 NIS 实施将修改这些函数的行为，并任命 RPC 调用让 NIS 服务器来查找用户名或 ID。这个过程对应用程序来说，是完全透明的。这个函数有两种可能：要么绑定在 NIS 映射上，要么用原始文件来“替换”这个映射。当然，这里的修改，并不是指真正修改了文件，只是针对应用程序而言，就像这个文件真的已被替换或绑定。

就传统 NIS 实施来说，哪些映射应该替换，哪些又应该绑定到原始信息，过去曾有相关的约定。有的映射，比如 passwd，要求对 passwd 文件稍作修改，以免该文件一旦出错，就会导致安全漏洞。为避免这类情况，NIS 采用了一个常规配置方案，用于判断特定的客户机函数集是否使用的是原始文件、NIS 还是 NIS+，使用顺序又是什么样的。有关详情，参见本章最后一节。

9.4 NIS服务器的运行

吹了那么多理论上的技术泡泡，现在该动手实际配置了。本小节将全面讨论 NIS服务器的配置。如果你所在的网络上已经有一台 NIS服务器，不必再自行设置；最好跳过本小节。

注意 如果只是想体验一下配置服务器，也不能对网络中正在运行的NIS域名动手动脚。

因为这样可能瓦解整个网络服务，令许多人感到不快，甚至愤怒。

目前，有两个NIS服务器可用于Linux，一个包含在Tobias Reber的ypserv包内，另一个包含在Peter Eriksson的ypserv包内。不管你使用的是 NYS还是当前libc内的标准NIS客户机代码，其运行结果都大同小异。在本书完稿时，ypserv中的用于处理NIS从属服务器的程序也更为完整。所以，在不得不处理从属服务器时，ypserv便是你的首选。

下一小节将解释如何配置NIS客户机代码。如果你的设置无效，应该试着查找请求是否已抵达你的服务器。如果将-D命令行标记指定为NYS服务器，它就会在控制台打印出关于所有进入NIS查询的测试信息并返回结果。你也可从中了解问题出在哪里。

9.5 用NYS设置一个NIS客户机

从现在开始，我们将全面介绍NIS客户机的配置。

第一步，应该在/etc/yp.conf配置文件内，设置服务器，从而告诉NYS，你准备将这台服务器用于NIS服务。以Winery网络上的一台主机为例，它的示范配置文件如下所示：

```
# /etc/yp.conf - ypbind configuration file
# Valid entries are
#
#domain NISDOMAIN server HOSTNAME
#           Use server HOSTNAME for the domain NISDOMAIN.
#
#domain NISDOMAIN broadcast
#           Use broadcast on the local net for domain NISDOMAIN
#
#ypserver HOSTNAME
#           Use server HOSTNAME for the local domain.
#           The IP-address of server must be listed in /etc/hosts.
```

第一个语句告示所有的NIS客户机，它们属于winery NIS域。如果省略这一行，NYS就会采用你通过domainname命令为自己的系统所分配的域名。server语句命名准备使用的NIS服务器。当然，还必须在hosts文件内，设置与vbardolino对应的hosts的IP地址；或者采用server语句自带的IP地址。

上面的示例中，server命令要求NYS采用named服务器，不管当前的NIS域是什么。但是，如果你的机器经常性地不同的NIS域间游移，肯定想将各个域的相关信息统统保存在yp.conf文件内。将你需要的NIS域名增加到server语句内，yp.conf文件中便能拥有若干个NIS域的相关信息。

创建好基本的配置文件，并保证全世界的人都能读懂它之后，就应该开始第一轮测试，看你是否能连接到自己的服务器。务必保证选择你的服务器分发的映射，比如hosts.byname映射，并试着通过ypcat实用程序获取这些映射。与其他管理性的NIS工具一样，ypcat应该保存

在/usr/sbin内。

此时，你得到的输出应该和上面的示例类似。如果出现类似的错误消息：“不能绑定充当域的服务器”(Can't bind to server which serves domain)，意味着不是你设置的NIS域名之匹配服务器尚未在yp.conf文件内定义，就是由于某种原因，该服务器不能抵达。后一种情况中，要保证发给主机的ping会产生一个肯定结果，而该主机确正在运行一个NIS服务器。利用rpcinfo，可对后一种情况进行验证，它将产生表9-1那样的输出。

表9-1 NIS配置的验证输出

程 序	vers	协 议	端 口	
100000	4	tcp	111	rpcbind
100000	3	tcp	111	rpcbind
100000	2	tcp	111	rpcbind
100000	4	udp	111	rpcbind
100000	3	udp	111	rpcbind
100000	2	udp	111	rpcbind
100024	1	udp	32772	status
100024	1	tcp	32771	status
100133	1	udp	32772	
100133	1	tcp	32771	
100021	1	udp	4045	nlockmgr
100021	2	udp	4045	nlockmgr
100021	3	udp	4045	nlockmgr
100021	4	udp	4045	nlockmgr
100012	1	udp	32773	sprayd
100001	2	udp	32774	rstatd
100001	3	udp	32774	rstatd
100001	4	udp	32774	rstatd
100221	1	tcp	32772	
100235	1	tcp	32773	
100068	2	udp	32775	
100068	3	udp	32775	
100068	4	udp	32775	
100068	5	udp	32775	
100083	1	tcp	32774	
100021	1	tcp	4045	nlockmgr
100021	2	tcp	4045	nlockmgr
100021	3	tcp	4045	nlockmgr
100021	4	tcp	4045	nlockmgr
805502976	2	tcp	911	
805502976	1	tcp	912	
300598	1	udp	32796	
300598	1	tcp	32782	
805306368	1	udp	32796	
805306368	1	tcp	32782	
100249	1	udp	32799	
100249	1	tcp	32783	

9.6 挑选合适的映射

确定自己能够抵达特定的 NIS 服务器后，必须决定用 NIS 映射来替换或增加配置文件。通常，大家会想到主机和密码查找函数所用的 NIS 映射。如果不运行 BIND 的话，前者显得身手不凡。后者允许所有的用户从这个 NIS 域的任何系统，登录到他们的账号；这通常要求所有的主机通过 NFS，共享一个 central/home 目录。有关详情将留在稍后讨论。其他映射，比如 services.byname，就没有这么大的本事了，但可以为你省些编辑工夫，前提是你安装的网络应用采用的服务名是标准 services 文件内没有的。

一般说来，在查找（lookup）函数采用 local 文件，并查询 NIS 服务器时，人们都想能有丰富的选项！NYS 允许大家对函数访问服务的顺序进行配置。这是通过 /etc/nsswitch.conf 文件来控制的，该文件代表的是“域名服务开关”，当然，它并不局限于域名服务。针对 NYS 支持的所有数据查找函数，它含有一行，对准备采用服务进行命名。

服务访问顺序和数据的类型有关。services.byname 映射不可能包含能够把本地 services 文件内的条目区分开来的条目，它只能包含尽可能多的条目。所以，最好先查询本地文件，如果未找到服务名，再查看 NIS。另一方面，主机名信息可能会频繁变动，所以 DNS 或 NIS 服务器应该始终保持最新、最准确的账号，而本地 hosts 文件只能充当 DNS 或 NIS 不能使用时的替补对员。这种情况下，只好查看本地文件了。

目前，NYS 支持的 nsswitch.conf 条目有：hosts、networks、passwd、group、shadow、gshadow services、protocols、rpc 和 ethers。还可能添加更多的条目。

清单 9-1 向大家展示了一个比较复杂的示例，引入了 nsswitch.conf 的另一个特性：hosts 条目中的 [NOTFOUND=return] 关键字要求 NYS 在没有在 NIS 或 DNS 数据库内找到所需要的项目时返回。也就是说，只有对 NYS 和 DNS 服务器的调用由于某种原因而失败时，NYS 才会继续在本地文件内查找所需项目。NIS 服务器关闭后再启动时，将本地文件当作备份使用。

清单 9-1 域名服务开关配置文件示例

```
#
# /etc/nsswitch.conf
#
# An example Name Service Switch config file. This file should be
# sorted with the most-used services at the beginning.
#
#The entry '[NOTFOUND=return]' means that the search for an
#entry should stop if the search in the previous entry turned
#up nothing. Note that if the search failed due to some other reason
#(like no NIS server responding) then the search continues with the
#next entry.
#
#Legal entries are:
#
#    nisplus or nis+ Use NIS+ (NIS version 3)
#    nis or yp Use NIS (NIS version 2), also called YP
#    dns Use DNS (Domain Name Service)
#    files Use the local files
#    [NOTFOUND=return] Stop searching if not found so far
#
```

```
passwd:    files nisplus nis
shadow:    files nisplus nis
group:     files nisplus nis

hosts:     files nisplus nis dns

services:  nisplus [NOTFOUND=return] files
networks:  nisplus [NOTFOUND=return] files
protocols: nisplus [NOTFOUND=return] files
rpc:       nisplus [NOTFOUND=return] files
ethers:    nisplus [NOTFOUND=return] files
netmasks: nisplus [NOTFOUND=return] files
bootparams: nisplus [NOTFOUND=return] files

netgroup:  nisplus

publickey: nisplus

automount: files nisplus
aliases:   files nisplus
```

9.7 使用passwd和group映射

NIS的主要应用之一是同步更新 NIS域内的所有主机的相关用户和账号信息。鉴于此，一般都要保留一个小型的本地 `/etc/passwd` 文件，NIS映射的站点级信息将绑定在一个文件内。但是，只将NIS查找用于 `nsswitch.conf` 内的这项服务是远远不够的。

在依赖NIS分发的密码信息时，首先必须保证：你自己的本地 `passwd` 文件内任何一个用户的数字化ID与NIS服务器中的用户ID匹配。当然，你肯定还想过把它用于别的用途，比如从网络内的其他主机装入NIS卷。

如果 `/etc/passwd` 或 `/etc/group` 内的数字化ID不包含在映射中，就必须对属于特定用户的所有文件之所属关系进行调整。首先，将 `passwd` 和 `group` 内的所有 `uid` 和 `gid` 改成新值，然后，再找出属于新用户的所有文件，最后，再更改这些文件的所属关系。假设 `news` 过去的用户ID是9，而 `okir` 的用户ID是103，它们的用户ID将发生变动；可执行下面的命令：

```
# find /-uid 9 -exec chown news {} \;
followed by a
# find /-uid 103 -exec chown okir {} \;
```

先说说第一条命令：从根目录中，找出用户ID是9的用户拥有的所有目录及文件，在找到的目录或文件中，将拥有者改为“`news`”用户（其UID刚才在密码文件中进行了更新）。需要查找(`find`)的原因是用户名和组名只代表对应的用户，所有文件和目录都归这个数字化的用户ID(`uid`)和组ID(`gid`)所有。`passwd`或`group`文件内，特定用户的`uid`和`gid`值发生变动时，磁盘上的文件仍然归以前的`uid/gid`对的`uid/gid`拥有。要纠正这些文件的所属关系，必须通过文件系统进行递推，并将文件“`chown`”向新`uid/gid`。为什么说`chown`呢？其原因是`chown`将在密码文件内查找和`news`对应的`uid`。最后一个`\;`终止了`find`命令；外壳程序中，斜杠后面的冒号是省略了的，将被解释为命令终止符。

利用才安装的新 `passwd` 文件来执行这些命令，并且在改变文件所属关系之前收集所有文

件名是非常重要的。要更新文件的组所属关系，可采用和前面类似的命令。

至此，系统上的 uid 和 gid 值都将和 NIS 域内其他主机上的那些值一致。下一步是在启用 NIS 查找用户和组信息的 `nsswitch.conf` 文件内，增加相应的配置行（参见上面的清单 9-1。特别是 “`passwd`”、“`shadow`” 和 “`group`” 这几行）。

如此一来，用户在试图登录时，`login` 命令和它的所有对等体都会率先查询 NIS 映射，如果查找失败，就回到本地文件中查找。通常，你会从自己的本地文件中删除所有的用户信息，只保留根和常用账号所用的条目，比如 `mail`。这是因为有些重要的系统任务可能要求将 uid 映射为用户名，或把用户名映射为 uid。例如，管理 `cron` 的任务是执行 `su` 命令，临时性地充当 `news` 系统，或 UUCP 子系统可能将邮寄一份状态报告。如果本地 `passwd` 文件内没有针对 `news` 和 `uucp` 的条目，这些任务就会以失败告终。

这里为大家提出两大警告：一方面，迄今为止介绍的设置只适用于没有采用“影子”密码的登录套件，比如包括在 `util-linux` 包内的那些。关于随 NIS 一起使用的影子密码之复杂性将在后面讨论。另一方面，登录命令不是唯一的可访问 `passwd` 文件的命令——看看 `ls` 命令，它是多数人一直都在使用的。只要需要大型的列举，`ls` 就会显示出一个文件之用户和组拥有者的象征性用户名；也就是说，对自己碰到的每一个 uid 和 gid，它都必须一一查询 NIS 服务器。这样显然效率不高，更糟糕的是，NIS 服务器不在同一个物理网络上时，数据报只好通过路由器进行传递。

还有一点需要说明：看看用户打算更改自己的密码时，会发生什么样的事情。通常，她会调用 `passwd`，该命令读取新密码并更新本地 `passwd` 文件。这对 NIS 来说，是不可能的。因为本地 `passwd` 文件已不能在本地使用，而是在用户打算更改自己的密码时，只有登录到 NIS 服务器才能更改自己的密码，别无选择。鉴于此，NIS 用名为 `yppasswd` 的一个 daemon 来替换了 `passwd`，后者能在 NIS 中实现本地更改密码。所以，为更改服务器主机的密码，它通过 RPC，与该服务器主机上的 `yppasswd` daemon 取得联系，为它提供更新过的密码信息。通常，执行下面的代码，便可在普通程序上安装 `yppasswd`：

```
# cd /usr/bin
# mv passwd passwd.local
# ln yppasswd passwd
```

如此这般，便使用 `yppasswd` 命令替换了 `passwd`。

与此同时，你还必须在服务器上安装 `rpc.yppasswdd`，并从 `rc.inet2` 开始启用。这样，便可有效地偷梁换柱，NIS 一如既往地运行。

9.8 NIS与影子支持

John F. Haugh（`shadow` 套件的作者）近来在 `comp.source.misc` 发布了影子库函数的一个版本，该版本符合 GNU Library GPL 标准。它已经能够支持 NIS，但不很全面，所以尚未包括在标准 C 语言库内。另一方面，通过 NIS，从 `/etc/shadow` 出版信息也可能对影子套件的性能产生影响。

尽管 NYS 密码查找函数没有采用 `shadow.byname` 映射或其他类似的东西，但 NYS 支持透明使用本地 `/etc/shadow` 文件。调用 `getpwnam` 的 NYS 实施方案查找与指定登录用户名相关的信息时，就会对 `nsswitch.conf` 文件内 `passwd` 指定的设备进行查找。NIS 服务只查找 NIS 服务器上

passwd.byname映射内的用户名。但是，files服务将检查是否有/etc/shadow，如果有，就会试着打开它。如果没有，或用户没有root特权，它就会恢复其原有的行为，只在/etc/passwd内查找用户信息。但是如果影子文件存在，并且可以将其打开，NYS将会从影子文件内抽取用户密码。getpwuid函数的实施也如前所说。这种方式中，用NYS编译的二进制文件将透明地处理影子套件的本地安装。

9.9 使用传统的NIS代码

如果你采用的客户机代码属于当前标准libc的一部分，NIS客户机的配置则稍有不同。一方面，在要求得到自己需要的信息时，它用一个ypbind daemon向所有活动服务器发出广播，而不是从配置文件内收集。因此，一定要在启动时，开始启用ypbind。而且必须在NIS域已经被设定和RPC portmapper（端口映射器）已经启动之后才调用它。然后，像前面所说的那样，调用ypcat测试服务器。

近来，有大量的错误报告出现，说NIS失败时将出现这样一条错误消息：“clntudp_create:RPC portmapper failure -RPC:unable to receive”，意思是RPC端口映射器无效，RPC不能接收信息。之所以出现这种情况，是因为ypbind将绑定信息传达给库函数时，采取的方式与以前的不兼容。最新NIS实用程序源代码的获得和编译可以解决这个问题（yp-linux的源代码可从ftp.uni-paderborn.de获得，后者位于/pub/Linux/LOCAL目录中）。

与此同时，对传统NIS来说，判断是否和如何合并NIS信息和本地文件内的信息的方式和NYS所用的方式有所不同。例如，为了使用NIS密码，必须将下面这行包含在etc/passwd映射内的某个地方：

```
+.....
```

这一行标志密码查找函数从这里“插入”NIS映射。在/etc/group内插入类似的行（减去最后两个冒号），对group.*映射来说，同样如此。如果要用NIS分发的hosts.*映射，改变host.conf文件内的“顺序”行即可。比如，如果想用NIS、DNS和/etc/hosts文件（按下面的顺序），需要将顺序行改为：

```
order nis , bind , hosts
```

目前传统NIS实施方案还不支持其他的映射。