

SelfLinux-0.12.2



NIS - Yellow Pages



Autor: Florian Frank (florian.frank@pingos.org)
Autor: Frédéric Raynal (pappy@users.sourceforge.net)
Formatierung: Florian Frank (florian.frank@pingos.org)
Lizenz: GFDL

Der Network Information Service (NIS) stellt auf einem Server eine Datenbank zur Verfügung. Jeder Computer im Netzwerk, auf dem ein NIS Client läuft, kann eine Abfrage an diese Datenbank absetzen, um Benutzerinformationen zu erhalten (z. B. Login-Name, Passwort, User Groups,...). Durch diese Datenbank wird die zentralisierte Administration einer grossen Anzahl von Computern ermöglicht - insbesondere, wenn hier gleichzeitig ein Dateisystem wie NFS eingesetzt wird, da durch den Server und die Datenbank Änderungen der Benutzerinformationen sofort allen Clients zur Verfügung stehen.

Inhaltsverzeichnis

1 Einleitung

2 Wie funktioniert YP (NIS)?

- 2.1 Die Struktur
- 2.2 Die Maps
- 2.3 Remote Procedure Calls (RPC)

3 Auf der Seite des Clients

- 3.1 Client Einführung
- 3.2 Konfiguration des NIS-Clients
- 3.3 ypbind
- 3.4 Einige Details
- 3.5 Das NIS-Protokoll
- 3.6 Die yp-Tools
- 3.7 Einige Worte zu NIS+

4 Die Server - Seite

- 4.1 Server Einführung
- 4.2 Vorwort
- 4.3 Der NIS Server
- 4.4 Installation
- 4.5 Installation eines NIS-Servers
- 4.6 Aktualisierung der NIS-Datenbank

5 Benutzungsratschläge zum Schluss

1 Einleitung

Der Network Information Service (NIS) ist ursprünglich eine Entwicklung von  [Sun](#) und als **Sun Yellow Pages** bekannt (noch bekannter einfach als **Yellow Pages** oder **YP**). Doch dies ist eigentlich eine Handelsmarke der British Telecom und dürfte konsequenterweise nicht ohne die entsprechenden Rechte benutzt werden. Die Yellow Pages der British Telecom sind das Branchentelefonbuch (wie im deutschsprachigen Raum die **Gelben Seiten**).

Die NIS Server speichern Kopien von gemeinsamen Konfigurationsdateien (z.B. `/etc/shadow`, `/etc/passwd`, `/etc/group` ...) verschiedener vernetzter Computer in einer Datenbank. Die NIS Clients wiederum richten ihre Anfragen an diese Server, anstatt eigene Konfigurationsdateien zu benutzen, bzw. als Erweiterung der eigenen Konfigurationsdateien.

Nehmen wir einmal an, wir wären User im Netzwerk und wollten das Passwort ändern. Und nehmen wir weiter an, YP sei nicht installiert. Wenn wir uns die Möglichkeit offenhalten wollten, uns von jedem Computer im Netzwerk einloggen zu können, müssten wir auch die Passwortdateien auf jedem einzelnen Computer aktualisieren. Wäre aber YP installiert, dann wäre es uns möglich, die Änderung auf einer einzigen Maschine vorzunehmen, auf der ein NIS-Client läuft. Das neue Passwort würde dann dem NIS-Server übermittelt und in der Datenbank geändert. Und wenn sich nun ein User an einem vernetzten Computer einklinken wollte, würde das Passwort mit dem in der Datenbank auf dem Server verglichen (natürlich müsste auch dann ein NIS-Client auf dem Computer des Users laufen).

glibc 2.x (libc6) unterstützt den Einsatz von NSS (**N**ame **S**witch **S**ervice). Dieser Dienst bestimmt durch die Datei `/etc/nsswitch.conf`, in welcher Reihenfolge Informationen gesucht werden müssen. Er unterstützt **Aliases**, das **Ethernet**-Protokoll, **Groups**, **Hosts**, **Netgroups**, **Netzwerke**, **Protokolle**, **öffentliche Schlüssel**, **Passwd**, **RPC**, **Dienste** und **Shadow Maps**.

2 Wie funktioniert YP (NIS)?

2.1 Die Struktur

Im Netzwerk wird ein Computer als NIS-Server für eine Domäne dienen. Diese Domäne stimmt mehr oder weniger mit dem Namen der Datenbank überein, die vom Server verwaltet wird. Der Domänenname ist der Schlüssel, der von den NIS-Clients gebraucht wird, um die benötigte Information auf dem Server zu lokalisieren. Dieser Domänenname hat absolut nichts mit dem DNS Domain Name zu tun. Es kann mehr als einen NIS-Server in derselben DNS-Domain geben. Sie können auf dem NIS-Level unterschiedliche Domänen verwalten, oder diesselbe NIS-Domäne (in diesem Fall gibt es einen Master-Server und einen Slave-Server).

Die Slave-Server speichern lediglich eine Kopie der Datenbank des Master-Servers. Sie unterstützen den Master, wenn er zu viel Zeit benötigt, um die Anfragen der Clients zu beantworten, oder er gar in die Knie geht.

Die Slaves werden über jede Änderung im Datenbestand durch das Programm **yppush** informiert, und sie werden daraufhin ihre eigenen Datenbanken auf den neuesten Stand bringen, um die Master-Datenbank exakt wiederzuspiegeln.

Die Clients benötigen ihrerseits keine Pflege, da sie ständig mit dem NIS-Server verbunden sind und auf die Informationen in dessen Datenbank zugreifen können.

2.2 Die Maps

Die YP-Datenbanken liegen im **GDBM**-Format vor, das aus dem **ASCII**-Format erzeugt wird. Diese Konvertierung geschieht bei der Installation des Servers durch das Programm **makedbm**.

Diese Maps bestehen aus **Schlüssel/Wert-Beziehungen**. Alle YP-Maps basieren auf diesem Modell. Für den Server ist der Inhalt dieser Paare ohne Bedeutung (mit Ausnahme der Daten, die den Master-Server betreffen). Das bedeutet, dass für den Server eine Map mit Passwörtern, Gruppen, oder was-auch-immer, nichts anderes ist als eine Ansammlung von Schlüssel/Wert-Paaren. Nur der Client weiss, wie diese richtig zu deuten sind, und wie er die Information findet, die er braucht.

Diese Repräsentation von Daten kann problematisch werden. Da der Server den zu einem Schlüssel gehörenden Wert nicht interpretierend lesen kann, kann er auch einen zweiten, verborgenen Schlüssel nicht finden. An einem Beispiel wird deutlich, was gemeint ist: Sucht der Client nach **Passwörtern**, könnte er vom Login-Namen ausgehen oder von der **UID (User ID)**, eine eindeutige Kennung für jeden User im Netzwerk). Um diese Suche zu ermöglichen, muss die Passwort-Information verdoppelt werden. Dies führt uns allerdings zu redundanter Information, wie man an den Dateien **passwd.byname** und **passwd.byuid** sehen kann. Für jede Form der Suche muss eine Map erzeugt werden, und bei einer Änderung müssen die Daten mehrfach übertragen werden.

Drei Parameter werden von dem Client benötigt, um eine gesuchte Information in der Datenbank aufzuspüren:

- * der Name der Domain: das ist der Name der Datenbank auf dem YP-Server
- * der Name der Map
- * der Name des Schlüssels

Benötigt also ein Client das Passwort des Users **toto** in der Domain **titi**, wird er in der Datei **/var/yp/titi/passwd.byname** nach dem User **toto** suchen.

Das führt zu einem sehr flexiblen System, da es nun, um eine neue Domain einzurichten, lediglich nötig ist, das Verzeichnis **/var/yp/new_domain** zu erzeugen, das **Makefile** zu kopieren, und mit den korrekten Optionen auszuführen.

2.3 Remote Procedure Calls (RPC)

Die Funktionalität der Yellow Pages basiert im wesentlichen auf den Remote Procedure Calls (RPCs), dem Austausch von Anfragen zwischen Server und den Clients.

Der RPC Portmapper **portmap** ist ein Programm, das die RPC-Programm-Nummern in Portnummern übersetzt. Wenn ein RPC gestartet wird, wird es portmap mitteilen, welchen Port es benutzen will und welche RPC-Programm-Nummer es ansprechen will.

Wenn ein Client eine RPC-Abfrage an eine bestimmte Programm-Nummer richten will, wird er zuerst den portmap-Server kontaktieren, um die Nummer des Ports zu erfahren, auf dem dieses Programm läuft. Dann kann der Client die RPC-Pakete an den entsprechenden Port schicken. Das YP Client/Server-Modell ist also nur ein Sonderfall des RPC Client/Server-Modells.

Die Datei **yp_prot.h** enthält die Strukturen und die Prototypen für 11 Funktionen, die das RPC-Protokoll definieren.

- * **YPPROC_DOMAIN** und **YPPROC_DOMAIN_NOACK** ermöglichen den Clients, zu einer gegebenen Domain den richtigen Server zu finden.
- * Die Funktionen **YPPROC_MATCH**, **YPPROC_FIRST**, **YPPROC_NEXT** und **YPPROC_ALL** ermöglichen es, auf die Daten der Maps zuzugreifen.
- * **YPPROC_XFR** wird von yppush aufgerufen, um den Slaves anzuzeigen, dass sich die Map auf dem Master geändert hat und die Kopien auf den neuesten Stand gebracht werden müssen.

- * **YPPROC_CLEAR** löscht den Inhalt des Caches und der File-Handles. Diese Funktion wird aufgerufen, nachdem eine Map upgedated wurde, z.B. nach dem **makedbm -c** Kommando.
- * **YPPROC_MASTER**, **YPPROC_ORDER** und **YPPROC_MAPLIST** ermöglichen es, spezielle Informationen über die Maps zu erhalten. Wenn zum Beispiel auf einem Client ein Passwort geändert wird, ruft das Programm **yppasswd** die Funktion **YPPROC_MASTER** auf, um den Server zu bestimmen, bevor dort die Datenbank geändert wird.

3 Auf der Seite des Clients

3.1 Client Einführung

Die Client-Dienste, die zu yellow pages gehören, basieren auf dem **ypbind** Daemon. Er sendet die Anforderungen an den YP-Server. Zunächst betrachten wir seine Arbeitsweise und die Konfiguration. Danach sehen wir, wie das NIS-Protokoll arbeitet. Im Anschluss befassen wir uns mit den verschiedenen Tools, die auf der Client-Seite zur Verfügung stehen.

3.2 Konfiguration des NIS-Clients

Die einzige Voraussetzung, um einen NIS-Client zur Verfügung zu haben, ist der Start des **ypbind**-Dämons.

3.3 ypbind

ypbind etabliert die Verbindung zwischen Client und dem NIS-Server. Diese Verbindung wird sichtbar durch eine Datei im Verzeichnis **/var/yp/binding1**, die normalerweise in der Form **domainname.version** benannt ist. Die einzige z. Z. unterstützte Version ist Version 2. Wenn der Name der NIS-Domäne etwa **messiah** lautet, heißt die Datei **messiah.2**.

Das Programm **ypbind** gehört dem Super-User (d. h. **root**), demzufolge findet es sich entweder in **/sbin** oder **/usr/sbin**.

Nach dem Start sucht und findet (hoffentlich) **ypbind** seine Anweisungen in der Datei **/etc/yp.conf**. Diese Datei kann folgende Einträge enthalten:

- * **domain nisdomain server hostname**: Der Client sucht nach hostname für die Domäne nisdomain;
- * **domain nisdomain broadcast**: Der Client fragt per Rundspruch im lokalen Netzwerk nach der Domäne nisdomain;
- * **ypserver hostname**: Der Client spricht direkt hostname für die lokale Domäne an. In dieser Konfiguration muß die **IP-Adresse** des Servers in der lokalen Datei **/etc/hosts** enthalten sein.

Wenn die Konfigurationsdatei nicht existiert oder falsche Einträge enthält, sucht **ypbind** per Rundspruch (broadcast) im lokalen Netzwerk nach dem NIS-Server für die lokale Domäne.

Mit wenigen Schritten lässt sich sicherstellen, dass **ypbind** korrekt konfiguriert ist.

1. Erstellen Sie die Datei **/etc/yp.conf**.

2. Überprüfen Sie, dass portmap aktiviert ist (**ps aux | grep portmap**). Wenn nicht, müssen wir ihn starten. Dieses Programm verbindet die **TCP/IP-** (oder **UDP/IP**)-Ports des Computers mit den Programmen. Während der Initialisierung eines RPC-Servers teilt dieser **portmap** mit, auf welchen Ports er lauscht und die Programmnummern, die er starten möchte. Wenn ein Client eine RPC-Anforderung für eine bestimmte Port-Nummer absetzt, nimmt er zunächst Kontakt mit **portmap** auf, um zu erfahren, an welchen Port die RPC-Pakete zu senden sind. Deshalb ist es unbedingt erforderlich, **portmap** vor **ypbind** zu starten.

3. Erstellen Sie das Verzeichnis **/var/yp**.

4. Starten Sie **ypbind**.

5. Benutzen Sie das Kommando `rpcinfo`, um sicherzustellen, dass `ypbind` korrekt arbeitet:

Die Ausführung von `rpc -p localhost` sollte folgende Informationen anzeigen:

```
program      vers  proto  port
100000      2    tcp    111    portmapper
100000      2    udp    111    portmapper
100007      2    tcp    637    ypbind
100007      2    udp    639    ypbind
```

oder

```
program      vers  proto  port
100000      2    tcp    111    portmapper
100000      2    udp    111    portmapper
100007      2    udp    758    ypbind
100007      1    udp    758    ypbind
100007      2    tcp    761    ypbind
100007      1    tcp    761    ypbind
```

oder versuchen Sie:

`rpcinfo -u localhost ypbind`, was als Ergebnis folgendes anzeigen sollte:

```
program 100007 version 2 ready and waiting
```

oder

```
program 100007 version 1 ready and waiting
program 100007 version 2 ready and waiting
```

abhängig von der Version von `ypbind`. Der wichtige Teil ist die Zeile mit der Version 2.

Nachdem `ypbind` nun korrekt arbeitet, wird Ihr Rechner zu einem NIS-Client. Nun können Sie Ihren NIS-Server ansprechen. Z. B. zeigt Ihnen `ypcat passwd.byname` alle Passwörter nach Benutzern geordnet an, die im entsprechenden Verzeichnis vorhanden sind.

3.4 Einige Details

Einige Dateien benötigen noch kleinere Veränderungen, damit YP effizienter arbeiten kann:

- * `/etc/host.conf`: `nis` für Namensauflösung hinzufügen.
- * `/etc/passwd`: folgende Zeile einfügen:

/etc/passwd
+:::~:

Dies erlaubt allen Personen, die im Verzeichnis des Servers enthalten sind, Kontakt zum Client aufzunehmen. Diese Berechtigungen können durch die Symbole + und - verfeinert werden, um den Zugang zum Client zu gewähren oder zu sperren. Um z. B. den **guest**-Benutzer auszusperrern, würden Sie die Zeile

/etc/passwd
-guest:::~:

hinzufügen. Die Felder, die Sie nicht verändern möchten, sollten leer bleiben. Es ist aber möglich, mehr hinzuzufügen:

/etc/passwd
+me:::~:/bin/ksh

Der Benutzer **me** bekommt **ksh** anstelle ihrer sonstigen Login-Shell (die in `/etc/passwd` des NIS-Servers definiert ist). Ich möchte noch hinzufügen, das NIS auch die Benutzung von Netzgruppen perfekt unterstützt

/etc/passwd
+@sysadmins:::~:

erlaubt den Mitgliedern der Netzgruppe **sysadmins** Verbindungen zu diesem Client.

`/etc/group` (und/oder `/etc/shadow` für bestimmte libc-Versionen): wie in `/etc/passwd`, müssen Sie hier

/etc/shadow
+:~:

hinzufügen. Sie können auch mit Gruppenberechtigungen experimentieren, indem sie + und - hinzufügen.

`/etc/nsswitch.conf`: der **Network Services Switch** erlaubt es (wie `/etc/hosts.conf`), die Reihenfolge zu definieren, in der die gewünschte Information gesucht wird. Zur Auswahl stehen:

nisplus	Suche per NIS+ (d. h. NIS Version 3, eine sichere Version von NIS)
nis	Suche per NIS (NIS Version 2, entspricht YP)
dns	Suche per DNS (Domain Name Server)
files	Suche in den lokalen Dateien

db Suche in der Datenbank /var/db

Hinter jeder Such-Option kann ein Befehl in folgender Form eingesetzt werden:

```
`[' ( `!'? STATUS `=' ACTION )+ `']'
```

wobei:

- * STATUS => "success" oder "notfound" oder "unavail" oder "tryagain"
- * ACTION => "return" oder "continue"

Abhängig von der verwendeten libc-Version verlaufen nicht alle Abfragen gleich. Zum Beispiel verwaltet die libc5 keine Passwörter in `/etc/shadow`. Die auf einer Maschine zur Verfügung stehenden Dienste benutzen die Bibliothek `/lib/libnss_SERVICE.so.X`. Weitere Informationen zu diesem Service finden sich in den [Handbuch-Seiten](#) zu `nsswitch.conf` ("man nsswitch.conf").

Shadow-Passwörter in NIS werden nur mit der glibc2.x-Bibliothek unterstützt. Deshalb sollte man sorgfältig überlegen, ob man sie in `nsswitch.conf` aktiviert.

3.5 Das NIS-Protokoll

Nachdem unser NIS-Client nun voll funktionsfähig ist, wollen wir sehen, wie er die benötigten Informationen gewinnt.

Wenn ein Client eine Information benötigt, die sich in einem YP-Verzeichnis befindet, sucht er zunächst nach einem YP-Server. Dazu öffnet er eine TCP-Verbindung zum lokalen ypbind-Dämonen. Der Client informiert ihn über die (NIS-)Domäne, zu der er gehört und `ypbind` setzt mittels der Funktion **RPC YPPROC_DOMAIN_NOACK** einen Rundspruch im lokalen Netz ab. Die NIS-Server, die diese Domäne bedienen, antworten mit einem **ACK**. Die anderen verhalten sich ruhig.

`ypbind` schickt dem Client das Ergebnis der Anfrage (Erfolg oder Misserfolg), und bei Erfolg die Adresse des YP-Servers, der zuerst geantwortet hat. Der Client kann nun den Server mit seiner Anfrage adressieren, indem er die Domäne, das Verzeichnis und den Schlüssel angibt.

Dieses Protokoll ist ziemlich langsam, weil es TCP-Verbindungen benutzt. Dies wird noch dadurch verschlimmert, weil eine Menge Sockets benutzt werden. Um dies zu vermeiden, wartet `ypbind` nicht auf Anfragen von Clients. Tatsächlich verwaltet es eine Liste von Servern für jede Domäne in der Datei `/var/yp/binding/<domainname>.<version>` und überprüft regelmässig, ob diese Server noch aktiv sind.

3.6 Die yp-Tools

Dieser Abschnitt beschreibt kurz einige der Programme aus dem yp-Tools-Paket. Wenn Sie mehr darüber erfahren wollen, können Sie für jedes dieser Tools eine sehr ausführliche [Handbuchseite](#) aufrufen.

- * `domainname`: zeigt oder ändert (abhängig von Optionen) den NIS-Domänen-Namen
- * `ypcat`: zeigt die Werte aller Schlüssel im NIS-Verzeichnis;
- * `ypmatch`: zeigt die Werte eines oder mehrerer Schlüssel im NIS-Verzeichnis
- * `ypset`: damit kann spezifiziert werden, zu welchen NIS-Server `ypbind` Verbindung aufnehmen muss
- * `ypwhich`: gibt den Namen des NIS-Servers zurück. Mit `-m Verzeichnisname` als Option gibt es den

Namen des Hauptverzeichnisses zurück.

* `yppoll`: hat als Argument einen Verzeichnisnamen und gibt den Namen des Hauptservers zurück..

3.7 Einige Worte zu NIS+

Bis jetzt haben wir noch nicht über eine NIS-Variante gesprochen. NIS in einem Netzwerk zu benutzen, ist ein **grosses Sicherheitsrisiko**. Wenn z. B. der NIS-Server schlecht geschützt ist und eine Person mit bösen Absichten

* den NIS-Domänen-Namen

* die **IP-Adresse** eines NIS-Clients

herausfindet, ist es sehr einfach vorzutäuschen, dass man von dieser Maschine mit dieser IP-Adresse arbeitet (Spoofing). Dann kann man durch ein `ypcat passwd` mit allergrößter Leichtigkeit die Passwort-Liste erhalten.

NIS+ bietet eine zusätzliche Sicherheitsschicht, indem es ein auf dem Austausch von Schlüsseln basierendes Authentifizierungsprotokoll integriert und Datenverschlüsselung unterstützt..

Die Daten werden in Tabellen gehalten, die sich in verschiedenen Verzeichnissen befinden. Jede Spalte einer Tabelle enthält einen Kopfeintrag, der z. B. angibt, ob bei den Daten Groß-/Kleinschreibung unterschieden wird oder ob es sich um binäre Daten handelt.

Die erwähnte Struktur erlaubt es, Zugriffsrechte für die Verzeichnisse und Tabellen und zusätzlich für die Spalten in den Tabellen zu definieren. Es ist daher möglich, den Zugriff auf die Passwort-Tabelle für jeden Benutzer zu sperren, der nicht auf dem NIS+-Server zugelassen ist. Aber es erlaubt allen berechtigten Benutzerinnen Zugriff auf die gesamte Passwort-Tabelle mit Ausnahme des **Passwort**-Feldes. Nur der Eigentümer des **Passwort**-Feldes kann dieses abfragen..

Es gibt 4 Sicherheitsstufen:

1. Nobody: Der Benutzer ist nicht berechtigt.
2. Owner: Der Benutzer ist als Eigentümer berechtigt.
3. Group: Der Benutzer ist berechtigt und gehört zu einer Gruppe, die Zugriff auf dieses Objekt hat.
4. World: Der Benutzer ist berechtigt, ist aber nicht Eigentümer und gehört keiner Gruppe mit Zugriff auf dieses Objekt an.

In dieser Konfiguration ist `root` fast nur ein gewöhnlicher Benutzer. Wenn er nicht die entsprechenden Rechte hat, hat er keinen Zugriff auf die Passwörter der anderen Benutzer. So ist es nicht mehr möglich, sich als ein anderer Benutzer zu authentifizieren, aber ein Wechsel mittels `su` ist noch möglich.

Die Daten werden mit Ausnahme der Passwörter im Netz unverschlüsselt übertragen. Kein Passwort wird im Klartext übertragen.

NIS+ ist ein mächtiges Werkzeug, aber es ist schwer zu konfigurieren.

4 Die Server - Seite

4.1 Server Einführung

Hier nun werden wir sehen, wie man den Server konfiguriert und wir werden einige Ratschläge zur Nutzung von NIS erteilen.

4.2 Vorwort

Bevor ich anfangen, eine Präzisierung. Bis jetzt haben wir nur von **NIS** gesprochen. Jedoch gibt es davon zwei Varianten: das, was man das **traditionelle NIS** nennen könnte und **NYS**. Praktisch gibt es keine Unterschiede zwischen den Beiden (speziell bei der Konfiguration sowohl des Clients als auch des Servers). Das **traditionelle NIS** gibt es schon länger, aber es unterstützt nicht alle Neuerungen, wie Shadowpasswörter, die von **NYS** transparent verwaltet werden.

Wir behandeln hier eine aktuelle Version von **ypserv** (d.h. eine Version die neuer ist als 1.3.2, um unter anderem die Verwaltung von Shadowpasswörtern zur Verfügung zu haben). Deshalb handelt es sich um einen **NYS** Server und kein **traditionelles NIS**, wobei wir weiterhin **fälschlich** von NIS sprechen werden.

4.3 Der NIS Server

Es gibt zwei NIS Server: **ypserv** und **yps**. Die Konfiguration unterscheidet sich nicht wesentlich. Aber **yps** wird von seinem Autor nicht mehr gepflegt und hat schwere Sicherheitslücken. Wir werden uns deshalb nur mit **ypserv** beschäftigen.

Zuerst zeigen wir die Schritte, die nötig sind, um einen Server zu installieren. Wir zeigen Schritt für Schritt, wie die einzelnen Konfigurationsdateien auf den Installationsprozess des Servers einwirken.

Im Artikel arbeiten wir auf der Maschine **charly**. Die NIS Domäne trägt den Name **bosley**. Die Slaveserver sind **sabrina**, **jill** und **kelly**.

4.4 Installation

Zu allererst muss man sicherstellen, dass der **portmap** Dämon läuft. Wenn dem nicht so ist, muss er gestartet werden.

Danach legt man den Namen der NIS-Domäne fest. Es handelt sich dabei nicht um einen Domänennamen im Sinne von **DNS**, sondern im Sinne von YPs. Dieser Name muss aus Sicherheitsgründen anders als der der Maschine sein.

Mit dem Kommando **domainname** kann man den Domänennamen festlegen. In unserem Fall benutzen wir es wie folgt:

```
root@linux / # /bin/domainname bosley
```

Dieses Kommando fixiert den NIS-Domänennamen im RAM-Speicher. Jedoch, wenn die Serverkonfiguration beendet ist, dann wünscht man sich doch, dass dies automatisch beim Starten der Maschine erledigt wird. Dafür muss man eine Zeile in der Netzwerkkonfiguration **/etc/sysconfig/network** ändern:

/etc/sysconfig/network
NISDOMAIN=bosley

oder bei Debian in [/etc/defaultdomain](#)

/etc/defaultdomain
bosley

Sobald das Netzwerk beim nächsten [Reboot](#) initialisiert wird, wird auch automatisch der NIS-Domänenname festgelegt.

Der folgende Abschnitt beschäftigt sich mit dem Starten des ypserv Dämons. Zuvor muss man ihn mittels der Datei [/etc/ypserv.conf](#) konfigurieren. Dies ist eine ASCII-Datei:

1. Kommentare: Die Zeilen, die mit dem #-Zeichen beginnen.
2. Optionen für den Dämon: Diese Zeile schreibt sich so:

/etc/ypserv.conf
option: [yes no]

Mögliche Optionen sind [dns](#) (der Server fragt DNS, um die Clients zu finden, die nicht in den hosts-Maps auftauchen), [sunos_kludge](#) (obsolet) und [xfr_check_port](#) (um den Server auf einen Port unter 1024 zu lenken - yes als Default)

3. Zugangsregeln zum NIS-Server. Das Format ist

host:map:security:mangle[:field]

Sie erlauben es festzulegen, wer was sehen darf.

Die Manpage von [ypserv.conf](#) führt sehr klar alle Optionen und Möglichkeiten für Regeln aus.

Jetzt kann man den Server starten:

root@linux / # /etc/init.d/ypserv start

Um zu verifizieren, dass alles korrekt läuft:

root@linux / # rpcinfo -u localhost ypserv
--

```
program 100004 version 1 ready and waiting
program 100004 version 2 ready and waiting
```

Wir haben gesehen, dass es zwei Typen von Servern gibt: Master und Slaves. Der Master besitzt die NIS-Referenzdatenbank, wovon die Slaves nur eine Kopie haben. Sie dienen dazu, den Master von zu vielen Requests zu entlasten. Die Datenbank wird nur auf dem Server gepflegt. Erst danach wird sie auf die Slaveserver weiterkopiert.

Alles ist jetzt bereit, bis auf die Datenbank. Man muss sie nur noch erstellen. Und wer erstellen meint, sagt [Makefile](#). Es ist bereits fertig geschrieben, es müssen lediglich ein paar Variablen angepasst werden. Es befindet sich im Verzeichnis `/var/yp`. Es ist ausführlich und klar kommentiert. Die wichtigste Zeile ist, wo die Maps, die von NIS benutzt werden sollen, definiert sind. Auf **charly** sind dies:

```
all: passwd group hosts rpc services netid protocols mail shadow \
# netgrp publickey
# networks ethers bootparams printcap \
# amd.home auto.master auto.home passwd.adjunct
```

Zu dem, was per Default vorgegeben ist, sollte man auch die Verwaltung der Shadowpasswörter hinzufügen. Man muss dann aber auch den Wert der Variable **MERGE_PASSWD** von **true** auf **false** setzen. Sie legt nämlich fest, dass für die Konstruktion der NIS-Datenbank die Dateien `/etc/passwd` und `/etc/shadow` zu mischen sind.

Noch ein letztes Detail bevor wir die NIS-Datenbank erstellen, die Verwaltung der Zugriffsrechte. Es gibt zwei Methoden den Zugang zum Server zu verwalten: entweder macht er alles selbst, oder über **tcp_wrapper**. Wir behandeln hier die Sicherheitseinstellungen über **ypserv** selbst.

Wenn Sie nur die Binaries von **ypserv** haben, dann sagt Ihnen die Option **-v** mit welcher Konfiguration Ihr Binary kompiliert wurde:

```
root@linux / # ypserv -v
ypserv - NYS YP Server version 1.3.12 (with securenets)
```

Die Datei `/etc/ypserv.securenets` enthält paarweise Kombinationen von netmask/network, mit denen Sie den Serverzugang kontrollieren können. Sie müssen diese Datei unter allen Umständen modifizieren: als Default enthält sie:

```
0.0.0.0      0.0.0.0
```

was aller Welt den Zugang auf Ihren NIS-Server erlaubt. Es ist anzumerken, dass der Datei lediglich die IP-Adressen bekannt sind (nicht der Namen der Maschinen).

Jetzt können wir die NIS-Datenbank erstellen. Wir benutzen dazu das Kommando **ypinit**. Es erstellt die Datenbank in `/var/yp` und benutzt die Dateien aus `/etc` (dies ist der Standard, man kann auch ein anderes Verzeichnis im [Makefile](#) festlegen). Hier sind die Dateien, die die Daten für die Datenbank liefern (`/etc/passwd`,

/etc/group, /etc/hosts, /etc/networks, /etc/services, /etc/protocols, /etc/netgroup, /etc/rpc).

Die Option **-m** gestattet es, den Server mit den Rohdaten zu initialisieren (**-m** für master), die Option **-s** kopiert die Daten von der Masterdatenbank auf einen Slave (**-s** für Slave - Sklave auf Englisch).

Auf Charly initialisieren wir unsere Datenbank wie folgt:

```
root@linux / # ypinit -m

At this point, we have to construct a list of the hosts which will run
NIS
servers. localhost is in the list of NIS server hosts. Please continue
to add
the names for the other hosts, one per line. When you are done with the
list, type a <control D>.
    next host to add: localhost
    next host to add: sabrina
    next host to add: jill
    next host to add: kelly
    next host to add:
The current list of NIS servers looks like this:
localhost
sabrina
jill
kelly

Is this correct? [y/n: y] y
We need some minutes to build the databases...
Building /var/yp/bosley/ypservers...
Running /var/yp/Makefile...
gmake[1]: Entering directory `/var/yp/bosley'
Updating passwd.byname...
Updating passwd.byuid...
Updating group.byname...
Updating group.bygid...
Updating hosts.byname...
Updating hosts.byaddr...
Updating rpc.byname...
Updating rpc.bynumber...
Updating services.byname...
Updating netid.byname...
Updating protocols.bynumber...
Updating protocols.byname...
Updating mail.aliases...
Updating shadow.byname...
# shadow publickey # networks ethers bootparams printcap \
# amd.home auto.master auto.home passwd.adjunct
gmake[1]: Leaving directory `/var/yp/bosley'
```

Schon steht die Datenbank. Auf jedem Slaveserver muss man jetzt das folgende Kommando ausführen:

```
root@linux / # ypinit -s charly
```

Um sicherzustellen, dass alles korrekt läuft, reicht es aus, einen Server in einen Client zu verwandeln, egal ob

Master oder Slave, und einen Request abzusetzen. Zum Beispiel auf **charly**:

```
ypcat passwd mulder:x:500:100::/home/mulder:/bin/csh
scully:x:501:100::/home/scully:/bin/bash
```

Man kann nebenbei auch feststellen, ob die Shadowpasswörter korrekt funktionieren.

4.5 Installation eines NIS-Servers

1. portmap initialisieren.
2. Den NIS-Domänenname festlegen.
3. Die Konfigurationsdatei des NIS-Servers vorbereiten: `/etc/ypserv.conf`
4. Den **ypserv** Dämon starten.
5. In `/var/yp/Makefile`, die Maps auswählen, die von **ypserv** verwaltet werden sollen und dann die Kompilierungsoptionen einstellen.
6. Die Zugangsrechte für den NIS-Server in der Datei `/var/yp/securenets` festlegen.
7. Die NIS Masterdatenbank erstellen mit Hilfe des **ypinit -m** Kommandos.
8. Die Slavedatenbanken erstellen mit **ypinit -s <master server>**

4.6 Aktualisierung der NIS-Datenbank

Sobald man eine Map modifizieren möchte, um zum Beispiel einen neuen Slaveserver oder einen neuen User hinzuzufügen, muss man die NIS-Datenbank aktualisieren. Dies geht wie folgt.

Um einen Slaveserver hinzuzufügen genügt es, **ypinit -s charly** auf dem neuen Slaveserver auszuführen, und seinen Namen in die Datei `/var/yp/ypservers` des Masterservers einzufügen.

Wenn man einen neuen User anlegt, können sich mehrere Maps verändert haben (passwd, shadow, alias, etc ...).

Sobald man eine Map modifiziert hat, darf man nicht vergessen, ein **make** im Verzeichnis `/var/yp/` des Masterservers zu machen: dies aktualisiert seine Datenbank, indem es die Information integriert und auf die Slaves verteilt (mit dem Kommando **yppush**).

Das Program **rpc.ypxfrd** erlaubt es, die Transaktion zwischen einem Masterserver und seinen Slaves zu beschleunigen. Es gestattet einem Slave, die Datenbank des Master-Servers einfach zu kopieren, anstatt sie komplett neu zu erstellen. **rpc.ypxfrd** muss zur selben Zeit gestartet werden wie **ypserv** und nur auf dem Master. Dieses Programm ist notwendig für sehr grosse Maps.

5 Benutzungsratschläge zum Schluss

Jederman weiss, dass NIS nicht gesichert ist. Trotzdem sind die Dienste, die es zur Verfügung stellt so nützlich, dass es schade wäre, es nicht zu benutzen. Man muss deshalb einige Vorsichtsmassnahmen ausserhalb von NIS treffen, wenn man es benutzen will.

Genauso wie es einige Passwörter gibt, die man leicht erraten kann, werden auch NIS-Domänennamen benutzt, die vorhersehbar sind. Offensichtliche Kandidaten sind, wenn man einmal den (Maschinen-) Namen des NIS-Servers herausbekommen hat, der komplette oder teilweise Name des Servers oder auch der Name der Organisation, zu der der Server gehört. **ypwhich** gestattet es, den Namen der Domäne zu testen!

Der NIS-Domänenname erscheint an mehreren Stellen, besonders im Verzeichnis `/var/yp`, oder auch in einem Unterverzeichnis, das während der NIS-Installation angelegt wurde (auf dem(n) Server(n), aber auch auf den Clients) und den Namen der NIS-Domäne trägt. Man muss deshalb die Zugangsrechte zu diesem Verzeichnis genau festlegen. Man darf auf es keinen Fall, nicht einmal read only, mit **NFS** exportieren. Jeder kann dann dieses Verzeichnis auf seiner eigenen Maschine mounten, um den Domänennamen herauszufinden.

Darüber hinaus schadet es nicht, **tcp_wrapper** zu benutzen. Damit kann man nämlich den **portmap**-Prozess kontrollieren, und verhindern, dass jederman RPC-Requests auf die eigene Maschine absetzt.

Es ist ebenfalls von Vorteil, die Defaultroute nicht über Ihren NIS-Server zu legen, sondern statisches Routing zu den Clients und den Slaveservern zu benutzen. Der Server kennt auf diese Art nur Routen zu genau festgelegten Maschinen und kann deshalb nicht auf Requests von unbekannten Maschinen antworten.

Auf dem Router-Level erlaubt eine **Firewall**, den Zugang zu den NIS-Servern effektiv zu kontrollieren.

Diese Ratschläge ergeben sich aus dem gesunden Menschenverstand. Sie verändern nicht die Sicherheit von NIS selbst, sondern nur den Rahmen darum. Trotz dieser Probleme bleibt NIS ein effektives und praktisches Werkzeug.