

# Project #2, Part II: Implementing NIS

- NIS
- Daemons
- Limitations of NIS
- How We Will Use NIS
- NIS Domain Name
- NIS Software
- Setting Up NIS on `it20`
- `/etc/nsswitch.conf`
- Creating New Accounts on Ubuntu
- `/etc/passwd`
- `/etc/shadow`
- `/etc/group`
- `chown` and `chgrp`

# NIS

- One of the things you did when you installed Linux was to create the *sysadmin* user account
  - This account is only valid on your virtual machine – **not** on any other virtual machine
  - So, if you want to connect to another virtual machine, you must use a different user account
- However, setting up the same user account on every machine that you might possibly want to use on a network would be a nightmare for system administrators

# NIS

- Thus, we need a software service that allows us to configure many machines without going to each machine
- ***Network Information Service (NIS)*** is one such service. The idea behind NIS was to have:
  - One place, where configuration files for an entire network could be stored
  - ...which could then be copied automatically to the individual machines on the network

# NIS

- The information that NIS provides is stored in files that NIS calls *maps*
  - These map files are not text
  - They are in a special binary format. Therefore, to read these files you need special NIS tools
- The maps exists on the NIS server, and all other machines on the network get copies of these files from the server

# NIS

- NIS always has a ***master*** server, but there may be additional servers that have copies of the master server files – and can stand in for the master server if it is not available
- These servers are called ***slave*** servers
- NIS was developed by Sun and was originally called Yellow Pages
- For legal reasons, the name had to be changed, but many of the commands used in NIS start with "yp"

# Daemons

- NIS, like many network services, relies on daemons to provide its services
  - Daemons are processes that provide services and run in the background, never interacting directly with any user (even root)
  - Daemons are often started when the machine is booted
  - The programs that are run in these processes often have names ending in "d". (*Just in case you've ever wondered about that...*)
  - You use them in many contexts...

# Daemons

- For example:
  - The daemon that a web server runs is called `httpd`
  - To be able to log into your VM, you will need to have the `sshd` daemon running
- NIS uses several daemons, but the two most important daemons are
  - `ypserv` : runs on the master server
  - `ypbind` : runs on the other machines that use NIS services
- These other machines are called *clients*

# Limitations of NIS

- Since all the information that NIS provides is in the form of files, there is a practical limit to the size of the networks it can support
- Other databases architectures, like SQL, have a hierarchical **structure**, which makes searching for an individual entry much faster
- NIS uses a **flat file** architecture, in which each record is stored one right after another
- To search for a given entry, you have to start at the beginning of the file and continue until you reach the record you need



# Limitations of NIS

- In the worst case situation, you might have to scan the entire file to get the record you want, so flat file databases are not efficient for storing large amounts of data
- Another important limitation is **security**
- Without going into the details, NIS is not very secure
  - We will deal with this limitation by confining the NIS server to our little network
  - The firewall will not allow outside users to access the NIS ports
  - **Using NIS with the dynamic addresses in DHCP is a major security threat**
  - This is why, when using DHCP in this lab, we use static IP addresses instead of addresses from the pool
- NIS also creates a lot of network traffic which also makes it unsuitable for large networks

# How We Will Use NIS

- With all that NIS has going against it, why are we using it?
- For *small* networks, where
  - NIS is running behind a firewall which blocks access to the NIS ports from the outside, and
  - IP addresses are *static*
- NIS is efficient at distributing four configuration files

*/etc/passwd*

*/etc/group*

*/etc/shadow*

*/etc/hosts*

- ***/etc/passwd*** is where user account data is stored

```
$ head -3 /etc/passwd
```

```
root:x:0:0:root:/root:/bin/bash
```

```
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
```

```
bin:x:2:2:bin:/bin:/bin/sh
```

- ***/etc/shadow*** is where encrypted passwords are stored

```
$ sudo head -3 /etc/shadow
```

```
root:MTx$PIjueBrD5oAvilboAinqjLeJgOzZUHH$S6bkuU$JRbKYpXrc7Jp4DgnA
```

```
BkOJWuRLlPsgDit6hpARE/WAJ.nHaw0mSadntL:15770:0:99999:7:::
```

```
daemon:*:15722:0:99999:7:::
```

```
bin:*:15722:0:99999:7:::
```

- ***Why is ***sudo*** needed to view shadow but not passwd?***

- `/etc/group` is where data about groups is stored

```
$ head -3 /etc/group
```

```
root:x:0:
```

```
daemon:x:1:
```

```
bin:x:2:
```

- Check out `/etc/group` on `it20` and on your own VM. *Does it contain anything of note?*

- `/etc/hosts` is used to associate hostnames with IP addresses

```
$ head -20 /etc/hosts
# /config/IT/root/etc/hosts installed as
#           it20:/etc/hosts.
# If you modify this file, please notify sysprog
# so your mods can be fetched back to repository.

# Associate ip addresses with names.

# Myself (loop back)
127.0.0.1    localhost
#127.0.1.1  it20.it.cs.umb.edu it20
10.0.0.1    it20.it.cs.umb.edu it20

# Inside vm's for s13, per abird.
10.0.0.128  itvm21-1a.it.cs.umb.edu itvm21-1a
10.0.0.129  itvm22-1a.it.cs.umb.edu itvm22-1a
10.0.0.130  itvm23-1a.it.cs.umb.edu itvm23-1a
10.0.0.131  itvm24-1a.it.cs.umb.edu itvm24-1a
10.0.0.132  itvm25-1a.it.cs.umb.edu itvm25-1a
10.0.0.133  itvm26-1a.it.cs.umb.edu itvm26-1a
```

# NIS Domain Name

- When you configure NIS, you will be asked for a domain name
- This is **not** the same as an Internet domain name
- The NIS domain name is used by NIS only
- It can be any value, but all machines on the network need to use the same NIS domain name
- For simplicity, we will use our network domain ( it.cs.umb.edu ) for our NIS domain name

# NIS Software

- There are two components of the NIS package
- Server software runs on a server and provides copies of the configuration file
- Client software runs on all machines that use the software provided by the NIS Server

# Setting Up NIS on **it20**

- The same packages are used for both the server and the client versions of NIS
- On either machine you run

```
sudo apt-get update
sudo apt-get install nis
sudo apt-get install sysv-rc-conf
```
- On the server, **it20**, the NIS configuration file **/etc/default/nis** must be changed from its default values
  - We set the value of **NISSERVER** to **master**
  - This tells **it20** that it will be the master NIS server



# Setting Up NIS on **it20**

- NIS allows you to set up slave servers which get their data from the master
- We will not be using slave servers
- After the NIS software was installed on **it20**, the configuration file `/etc/yp.conf` had to be customized
  - The value of the `ypserver` parameter was set to `10.0.0.1`, which is the IP address of **it20**
  - This allows **it20** to use the NIS service it provides other machines for itself

# Setting Up NIS on it20

- To set up the directory structure that NIS would use, an initialization script had to be run  
`/usr/lib/yp/ypinit -m`
- The service was started using  
`sudo service ypserv start`
- Some daemons have to be assigned a runLevel
- A runlevel is one of seven states in the booting up and shutting down of a Linux machine

# Setting Up NIS on it20

- Each runlevel has access to different system processes
- The following command was used to set the proper runlevel for NIS  

```
sudo sysv-rc-conf ypserv on
```
- The next task in setting up NIS was to build the database
- Doing this required going to the `/var/yp` directory  

```
cd /var/yp
```
- and running  

```
sudo make
```

# Setting Up NIS on *it20*

- ***make*** is a utility that runs a compiler with the proper options to create a working software installation
- The instructions that ***make*** uses are contained in a file named ***Makefile***
- You can look at the contents of this file by running  

```
less /var/yp/Makefile
```
- after you connect to ***it20***

# /etc/nsswitch.conf

- **/etc/nsswitch.conf** tells a machine the order to get information from NIS or other information sources
- Each entry in this file represents a source of information that the machine uses to obtain configuration data
- For example the entry

**hosts: files dns**

tells the machine to *first* look in **/etc/hosts** when resolving a domain name into an IP address – and *then* to use DNS

# /etc/nsswitch.conf

- The most common values for information sources are

Value	Information Source
<b>files</b>	Local configuration file
<b>nis</b>	NIS server
<b>ldap</b>	LDAP server
<b>compat</b>	Local configuration file with some special options

- For more information run  
**man nsswitch.conf**

# Creating New Accounts on Ubuntu

- One of the most basic tasks of a system administrators is creating an account for a new user
- On Ubuntu there are two commands that will do this
  - ***adduser***
  - ***useradd***
- **useradd** is an executable file, while **adduser** is a Perl script that uses **useradd**, but is more user-friendly

# Creating New Accounts on Ubuntu

- These programs make entries in the following configuration files
  - [/etc/passwd](#)
  - [/etc/shadow](#)
  - [/etc/group](#)
- I will use **adduser** to create accounts for each of you on [it20](#)
- Your account on [it20](#) will have the same Unix ID as your account on the CS LAN





# Creating New Accounts on Ubuntu

- Using NIS, this account information will be put on every machine so that you can log in using this account
- I am using adduser to do this because it has the --no-create-home option, which tells adduser **not** to create a home directory for this account
  - The reason I am using this option is that your home directory will be on your virtual machine
  - In the next project, we will use NFS to make this directory available on **every** virtual machine

# /etc/passwd

- **/etc/passwd** contains basic information about each account on the machine
- At one time, this file did contain passwords – but no longer
- Passwords have been moved to **/etc/shadow**, where they are encrypted
- **/etc/passwd** can be read by anyone, but only root can change it
- Each line is a entry for a specific user account

# /etc/passwd

- Each record consists of 7 fields, separated by colons 
  - 1 - Username
  - 2 - Password
  - 3 - User ID (UID)
  - 4 - Group ID (GID)
  - 5 - Comment
  - 6 - Home directory
  - 7 - Command/Shell
- An  in the Password field means the password is in */etc/shadow*

# /etc/passwd

- The User ID (**UID**) is the internal representation of a Linux/Unix account
- The Username is just a convenience for human beings
  - UID 0 is assigned to root
  - UIDs 1-99 are reserved for other predefined accounts
  - UIDs 100-999 are reserved for other administrative purposes

# /etc/passwd

- Group ID (**GID**) identifies the default group for this user
- The **Comment** field is optional, but it is sometimes used to add additional information about the user – such as the user's full name, phone number, etc.
- The **Command/Shell** field
  - Usually contains the **absolute address** of the default shell for the user
  - But, it can be any Linux/Unix command

# /etc/shadow

- **/etc/shadow** stores passwords in an encrypted form
- Again, each line represents an account, and individual fields are separated by colons :

**1** - Username

**2** - Password

**3** - Last password change

**4** - Minimum days

**5** - Maximum days

**6** - Warning days

**7** - Inactive days

**8** - Expire

## /etc/shadow

- The *Password* field is encrypted
- *Last password change* is the number of days – since January 1, 1970 – that the password was last changed
- *Minimum days* is the number of days that must pass before the user can change the password
- *Maximum days* is the number of days, since the last password change, before a new password must be chosen

## /etc/shadow

- *Warning days* is the number of days since the last password was changed that must occur before the user is warned that the password must be changed
- *Inactive days* is the number of days after the password expires that the account will remain active
- *Expire* is the number of days since January 1, 1970 after which the account can no longer be used



# /etc/group

- A group is a collection of user accounts
- Groups can only be created by root
- **/etc/group** contains group data. There is a line in this file for each group on the machine, and individual fields are separated by a colon :
- Group name
- Password
- Group ID
- Group List

# /etc/group

- The *Group name* is for the benefit of human users.
  - The machine uses the *Group ID (GID)*.
  - GID is the number the system uses to refer to a group
- The *Password* field may contain an encrypted password for the group. Most system admins do not assign passwords to groups
- The *Group List* field contains a list of usernames, which are included in the group
- A unique group is usually created automatically for each new user. The group name is the same as the username

# chown and chgrp

- Every file or directory has an owner and an assigned group
- By default the owner is the account that created the file or directory, and the group is the one *created for that user*

- To change the owner use **chown**

```
sudo chown UNIX_USERNAME FILE_OR_DIRECTORY
```

- To change the group use **chgrp**

```
sudo chgrp UNIX_GROUPNAME FILE_OR_DIRECTORY
```

- Only root can run **chown** or **chgrp**