Project #3, Part I: Implementing NFS

- Distributed File Systems
- NFS
- Ports and Network Conversations
- Destination and Return Ports
- RPC-based Services

- Configuring Server Daemons
- /etc/exports
- autofs
- Sharing home directories in the Lab
- /etc/auto.home

Distributed File Systems

- Most networks have a central place where files are kept and are available to any machine on the network
  - This is a necessity whenever people work together
  - If you work in an office, you may have some files on your own PC
  - However, the files for your projects may reside on a file server where *anyone* on your team can access them
- There are technologies that allow machines to make their files available to anyone on the network, forming a ***`distributed file system`***

Distributed File Systems

- Our CS LAN has such a distributed file system
  - You can log into any machine on the network, and you will always land in your home directory…
  - *…even though you are not connected to the machine* whose hard drive has your home directory
  - Your home directory *lives* on the hard disk of some network machine…
  - …but this disk is **`mounted`** whenever you log in to *any* machine on the network!

Distributed File Systems

- There are two major file server packages that are commonly used on Linux:
  - Network File System (NFS)
  - Samba
- **Samba** allows files to be shared between *Linux* and *Windows* machines
- **NFS** only works on *Linux*

NFS

- Again, "NFS" stands for ***Network File System***, and it *only* works on Unix and Linux
  - NFS allows each machine on the network to make some of its directories available to other machines
  - Unix and Linux have NFS as part of the kernel
- After you finish the first part of Project 3, every VM will be both
  - an NFS *server* and…
  - an NFS *client*

NFS

- This means that each team will perform the setup for *both* the client *and* server components of NFS
  - Each machine will make the home directories for its users available to other machines when you log into them
  - Therefore, the home directories created on one virtual machine will be available on all the other virtual machines
- NFS makes it *seem* like all shared directories are part of one big file system.
- It does this through mount points

NFS

- A ***`mount point`*** is a directory on your machine where other shared directories will appear *as if* they were actually part of your local file system
- *Directories used as mount points should be* ***empty***
- Shared NFS directories can be
  - mounted automatically when the machine is booted ***or***…
  - made available from the command line – using the ***`mount`*** command

Ports and Network Conversations

- Every machine on a network needs services from other machines
  - Web pages, file access, configuration info, etc.
  - All of these services involve *communication* over the network
- This communication takes place using **packets**
  - Packets are *chunks of information* – sent out over a network – that allow two computers to communicate
  - When a file is transferred over the network, it is *broken up* into multiple packets

Ports and Network Conversations

- On the Internet, there are two major packet protocols
    - ***TCP***: *Transmission Control Protocol*
    - ***UDP***: *User Datagram Protocol*
- UDP is simpler than TCP
    - **TCP** establishes a connection between two machines
    - **UDP** does not
- "Connection" means that packets can keep going back and forth, as long as one machine needs the other's services.
- So most network services use ***TCP***

Ports and Network Conversations

- Knowing the IP address of the machine providing a service is *not enough* because you may need *multiple* services from that machine
- For example ***it20*** provides *several* services to each machine on the IT Lab network
  - ***DHCP***
  - ***DNS***
  - ***NIS***
  - ***ssh***
- All these services are available *SIMULTANEOUSLY*
- When you need a particular service from ***it20***, it is not enough to simply send a packet to ***10.0.0.1***

Ports and Network Conversations

- How is a machine like *it20* supposed to know what to do with the packet it gets from your virtual machine?
- In order to allow a given host to provide more than one service, you must provide *additional* information in the packet -- a ***port***
- A **port** is
  - A communication channel
  - At a **specific** IP address
  - On a specific machine.

Ports and Network Conversations

- Ports are identified by their port number
  - To get a service from another machine, you need its IP address and the port number
  - The combination of an IP address, a port, and the protocol used is called a **`socket`**
- Port numbers from 0 to 1023 are special
  - They are known as the **`well-known ports`**
  - The well-known ports are given to widely used network services, such as port 80 for a web server
  - On Linux, only root can start a process that uses a well-known port

Destination and Return Ports

- Two computers communicate by exchanging packets, addressed to a specific port, at a specific IP address
- This is called a *socket*
- For most network services, the communication goes in *two directions*
    - Therefore, when a client sends a packet to a server, it has to have a destination socket, consisting of ***the server's IP address*** and ***the port number for that particular service***
    - But it also has to provide a return socket, where it can receive packets back from the server

Destination and Return Ports

- The port number for the return socket is NOT the same as the destination port number
  - This allows your machine to get packets from multiple servers
  - For example, your browser can be receiving packets simultaneously from many different web servers, even though each server is listening on port 80
- Server ports are standardized, but clients pick their return ports at random to receive packets from the server

RPC-based Services

- NFS has been around in the Unix world for a long time – so long in fact that it does not use network ports, which came into common use long after Unix was created
- NFS uses a technology called **remote procedure calls**, which allow a user on one computer to run a program on another computer
- Instead of port numbers, RPC services use **program numbers**, each of which is assigned to a specific **daemon**

RPC-based Services

- On modern versions of Unix, these program numbers have to be converted to Internet port numbers
- This is done through a an RPC port map program, which runs as a background process on all Linux/Unix machines
- When an RPC-based service like NFS is started, it tells the port mapper what RPC program number it is using
- When another machine needs to use the service, it contacts the port mapper on the machine providing the service to learn the port number it should use

RPC-based Services

- Once the machine knows the port number, it creates a remote procedure call and wraps it up in a TCP packet sent the server
- This is knows as *tunneling*
- To find out what RPC program numbers are mapped to which ports use the command **`rpcinfo   -p`**

Configuring Server Daemons

- NFS provides its services through a number of daemons, using software contained in two Ubuntu packages

  `nfs-kernel-server`

  `nfs-common`

- The **`nfs-kernel-server`** package contains software for daemons that provide the basic NFS services
- The **`nfs-common`** package contains software for the daemons that help the *`nfs-kernel-server`* daemons do their job

Configuring Server Daemons

- The configuration file for these daemons is `/etc/default/nfs-kernel-server`
- The installations, that you will perform for NFS, will automatically configure all these daemons to run when you boot your virtual machine
- In addition, you will need to configure your VM…
  1. As a ***server***, to share *your* personal homes to the network
  2. As a ***client***, to mount *other* users' homes

/etc/exports

- Here, you are configuring your VM as an NFS *server.*
- If a machine wants to share a directory through NFS, it must make an entry in `/etc/exports`
- Therefore, there will be a line in this file for each directory you are sharing
- Each line will have the same format:

```
DIRECTORY_PATHNAME   HOST(OPTIONS)
```

- `DIRECTORY_PATHNAME` is the **absolute pathname** of the shared directory on your virtual machine

/etc/exports

- **`HOST`** is the machine or machines that can access the shared directory – i.e., the ***whom***
- You can specify the host in many ways
  - By the hostname (only for machines inside the network)
  - By the IP address for a machine
  - All the machines in a specific network or subdomain
- We will use the last format in today's project
- **`OPTIONS`** specify ***how*** the directory is to be shared

/etc/exports

- Some of the more important options are:

`ro`	Files in directory are read only
`rw`	Files in directory can be changed if you have the proper permissions
`root_squash`	The root account on another machine does not have root powers in this directory
`sync`	Changes to the files are written to disc before another uses accesses the file
`no_subtree_check`	Does not check permissions on parent directories of shared directories

- ***There must be*** no spaces ***between the options for any host!***

autofs

- Here, you configure your VM as an NFS *client*.
- Although you can mount shared directories at the command line, you will usually want this to happen *automatically* -- when the machine boots
- This is done using the **`autofs`** package, which you will install on your virtual machines in this project
- The **`autofs`** software is smart
  - By default, it will only mount a directory when you "ask for it" – i.e., *try to access it*
  - This conserves network resources

autofs

- The main configuration file for ***`autofs`*** is **`/etc/auto.master`**
- Each line in this file lists a mount point and points to another file with information about the directories to be mounted there
- You will make the following entry in **`auto.master`**

```
/home   /etc/auto.home
```

- This line tells ***`autofs`*** to get the information it needs to mount shared home directories from the file **`/etc/auto.home`** on your virtual machine

autofs

- You will copy this file from it20, where I have made an entry in **/etc/auto.home** on it20 for you and your teammate
- Each entry lists:
  - A directory (that can be mounted)
  - and where it can be found
- For example, if the **gh** account's home was located on itvm25-3b, the entry for the user's home directory in **/etc/auto.home** would be

```
gh   itvm25-3b:/home.itvm25-3b/&
```

autofs

- **gh** is the name of the directory in the NFS share and **itvm25-3b:/home.itvm25-3b/&** tells NFS where to find the source
- **itvm25-3b** is the hostname, and **/home.itvm25-3b/&** gives the absolute address of the shared directory
- There really isn't anything named **&**. The **&** stands for the name of whatever is the first thing on the line – which in this case is "gh"
- As such, on that line, /home.itvm25-3b/& is **really** **/home.itvm25-3b/gh**

autofs

- You must copy `/etc/auto.home` from `it20` to your VM, using the `scp` command
- You will need to set up the command *very carefully!*
- On your virtual machine, you will be able to
  - Log on to other virtual machines in the Lab and…
  - …see your home directory – as long as ***they*** have set up NFS properly
- Once you get ***`autofs`*** working, you will have access to ***your*** home directory for your account on the ***other*** virtual machines (as long as they have ***`autofs`*** working)

autofs

- However, at startup, if you run **`ls /home`** now, you would see nothing – even if your config is correct.
- That's because ***`autofs`*** does not really mount a directory *until it's needed*
- The moment you access (or "ask for") your home directory it will be there
- So if you ran **`cd ~`** to go to your home directory, that directory would be automatically mounted
- If you now ran **`ls /home`** again, you would see your home directory

autofs

- You can tell ***autofs*** to mount the directories without waiting for someone to access them if you use the "ghost" option in for the entry in **/etc/auto.master**
- To make the home directory I mentioned above visible (without going to that directory using cd), the entry in **auto.master** should be

```
/home   file:/etc/auto.home   --ghost
```

- However, you are advised *not* to do this in the project because it can complicate troubleshooting!
- Just be aware of the option, for future reference...

Sharing home directories in the Lab

- In the 2nd half of ***Project 2***, you created home directories for each person on your team
- Now, we will use NFS to share those home directories on the IT Lab LAN so that
  - you can log on to *any* virtual machine in the lab with your personal username and password…
  - …and your home directory will be available
- In order to make this work, we are going to have to make some changes

Sharing home directories in the Lab

- Before Project 3, `/home` should have *three* directories
  - One for sysadmin
  - One for the username of each team member
- In this project, you are asked to rename the current `/home` directory and then create a new `/home`
- This new directory will serve as the mount point for all the home directories of all students in this class – but ***not*** for the `sysadmin` account

Sharing home directories in the Lab

- We **don't** want the **`sysadmin`** account to be shared
  - We want it to be a unique account for each machine – so that only that one account can be used to make changes on that machine
  - So, while your individual home directories will be mounted in the new **`/home`**, the local **`sysadmin`** account will reside in the *old* **`/home`** directory – which you will have renamed in Project 4
- But you have to *tell* the Ubuntu running on the virtual machine to look in a *different* place to find the home directory for **`sysadmin`**

Sharing home directories in the Lab

- The location for the home directory for each user is found in **`/etc/passwd`**. Thus, you will have to edit the entry for **`sysadmin`** there.

- There is one further wrinkle in our setup…

- When you log in with your account on *another* virtual machine, you will access your home directory through ***NFS***, using the mount point **`/home`** on that VM

- But, you will get access to these home directories the same way on your *own* virtual machine

Sharing home directories in the Lab

- Although your real home directory *lives* in the directory you renamed, you will *access* it through NFS, at the local mount point **`/home`**
- Admittedly, it may seem weird to access a directory via NFS when it is already on your local machine.
- It does not have to be this way, but it makes things simpler for me

`/etc/auto.home`

- The file **`/etc/auto.home`** has the information that lets NFS mount the home directories for each student in the class
- For this to work, each machine must have a copy of this file from **it20**
- You'll use **`scp`**, which stands for "secure copy"
- It allows you to copy a file from one machine to another – with the data stream encrypted

`/etc/auto.home`

- You will need to set up the command carefully, supplying certain pieces of information....
  - A ***source***, consisting of a remote host (along with a valid user on that host) and the file path on that host
  - A ***destination***, a file path where the copied file should go
- You will be doing this from your VM's command line
- The remote host will be `it20`, and you can use the `it341` user account, or your own.
- Also, remember you are creating a new file in `/etc`