File sharing with Samba

ComS 252 — Iowa State University

Barry Britt and Andrew Miner

Theoretical background

Sharing files — in general

Idea:

- Have a trusted LAN
  - Firewall to the outside
  - Everyone on this side is trusted
- Want ability to share files from a central location
  - I.e., a "shared drive" simultaneously connected to all machines
- With all the comforts of a modern, multi–user filesystem
  - Hierarchical directory structure
  - Files have owners and groups and permissions. . .

How can we make this happen?

Sharing by Remote Transfers

One solution:

- Have a central server machine with lots of disk space
- Users do the following cycle:
  1. "Download" file(s)
  2. View or edit file(s)
  3. "Upload" file(s)
- Choose some download/upload mechanism, e.g.,
  - `scp`, or `rsync`, or subversion, or git...
    (subversion and git actually work quite nicely for this)

Sharing by Remote Transfers

One solution:

- Have a central server machine with lots of disk space
- Users do the following cycle:
  1. "Download" file(s)
  2. View or edit file(s)
  3. "Upload" file(s)
- Choose some download/upload mechanism, e.g.,
  - `scp`, or `rsync`, or subversion, or git...
    (subversion and git actually work quite nicely for this)

Problems with this solution:

- This does not act much like a "shared drive"
- What if the shared drive is huge with huge files?
  - E.g., filesystem in *terabytes*
  - E.g., files in *gigabytes*

What we really want

- Shared drive on server acts like a local drive on client
  - But clients interact with the server using some protocol
    - Not at the level of disk sectors
    - Details of file storage on the server are hidden
- Can open and edit files "directly"
- Call this "network file sharing" to stay generic

What we really want

- Shared drive on server acts like a local drive on client
  - But clients interact with the server using some protocol
    - Not at the level of disk sectors
    - Details of file storage on the server are hidden
- Can open and edit files "directly"
- Call this "network file sharing" to stay generic

Location transparency

A file's name does not reveal any hint of its physical storage location

What we really want

- Shared drive on server acts like a local drive on client
  - But clients interact with the server using some protocol
    - Not at the level of disk sectors
    - Details of file storage on the server are hidden
- Can open and edit files "directly"
- Call this "network file sharing" to stay generic

Location transparency

A file's name does not reveal any hint of its physical storage location

Location independence

A file's name does not need to be changed when its physical storage location changes

Network File Sharing Issue: Performance

`head -n 1 file.txt` on a local disk

1. System call for "read first block of file.txt"
2. Wait for I/O request in disk scheduler
3. Perform I/O
4. Return from system call

Network File Sharing Issue: Performance

`head -n 1 file.txt` on a remote disk

1. System call for "read first block of file.txt"
2. Send request to server and wait for reply
3. Wait for I/O request in disk scheduler (on server)
4. Perform I/O (on server)
5. Send block back to client
6. Return from system call

Network File Sharing Issue: Performance

`head -n 1 file.txt` on a remote disk

1. System call for "read first block of file.txt"
2. Send request to server and wait for reply
3. Wait for I/O request in disk scheduler (on server)
4. Perform I/O (on server)
5. Send block back to client
6. Return from system call

How fast is data transfer on the network compared to disk?

Network File Sharing Issue: Performance

`head -n 1 file.txt` on a remote disk

1. System call for "read first block of file.txt"
2. Send request to server and wait for reply
3. Wait for I/O request in disk scheduler (on server)
4. Perform I/O (on server)
5. Send block back to client
6. Return from system call

How fast is data transfer on the network compared to disk?

- SATA revision 1.0 transfer rate: $\approx$ 1.5 Gbit/s
- USB 2.0: 480 Mbit/s in theory, 280 Mbit/s in practice
- 100 Mbit/s ethernet is 100 Mbit/s

Network File Sharing Issue: Caching

- Files (or parts of files) are cached on the client
  - This includes directories, also
  - Improves performance: reduces network traffic
  - Your web browser does this too, by the way
- But now we have the cache consistency problem
  - Need to keep cached file consistent with file on the server

  Client–initiated : client handles this
  Server–initiated : server handles this
- Cache update policy: when to send writes back to the server

  write through : writes sent to server immediately
  delayed write : writes not sent immediately
  write on close : send changes when file is closed
- Consider how choices here affect reliability
  - What if the client crashes?

Network File Sharing Issue: File Sharing Semantics

File Sharing Semantics

- Specifies what happens if two processes open the same file
- Single machine, local disk:

Network File Sharing Issue: File Sharing Semantics

File Sharing Semantics

- Specifies what happens if two processes open the same file
- Single machine, local disk:
  - Typically use sequential consistency:
  - All reads and writes are processed sequentially
  - A write from one process is immediately visible to the next read from any other process
- Multiple machines, shared disk:

Network File Sharing Issue: File Sharing Semantics

File Sharing Semantics

- Specifies what happens if two processes open the same file
- Single machine, local disk:
  - Typically use sequential consistency:
  - All reads and writes are processed sequentially
  - A write from one process is immediately visible to the next read from any other process
- Multiple machines, shared disk:
  - Sequential consistency will be expensive
    - Lots of network traffic for each write
  - May provide session semantics
    - Changes are visible after a process closes the file
  - Typically: all bets are off

Network File Sharing Issue: File Sharing Semantics

File Sharing Semantics

- Specifies what happens if two processes open the same file
- Single machine, local disk:
  - Typically use sequential consistency:
  - All reads and writes are processed sequentially
  - A write from one process is immediately visible to the next read from any other process
- Multiple machines, shared disk:
  - Sequential consistency will be expensive
    - Lots of network traffic for each write
  - May provide session semantics
    - Changes are visible after a process closes the file
  - Typically: all bets are off
- Fun realization: this is more likely to happen with directories

Network File Sharing Issue: Users

- Want files to have owners, groups, and permissions
- Need user and group information available over the network
  - E.g., what if different machines have different sets of users?
  - How will each machine know about the other users?
- This can be integrated into the file sharing system
- Or this can be handled separately by another protocol
  - Can "synchronize" users and groups on all machines
    - By hand — does not scale well
    - Using file transfers or custom software "hacks"
    - Using a protocol designed for this
  - Can move users and groups to a central server
    - Server provides "users, groups, passwords" service
    - Clients must support this
    - Part of a larger issue: "Network Identity"
    - We will discuss this some, in a few minutes

Network File Sharing Issues: Conclusion

- Enough choices, just tell me what everyone does in practice

Network File Sharing Issues: Conclusion

- Enough choices, just tell me what everyone does in practice

Elegant and efficient file sharing solution that everyone uses

Network File Sharing Issues: Conclusion

- Enough choices, just tell me what everyone does in practice

Elegant and efficient file sharing solution that everyone uses

There isn't one. Sorry. All current solutions make design decisions in an attempt to balance out performance and usability.

Network File Sharing Issues: Conclusion

- Enough choices, just tell me what everyone does in practice

Elegant and efficient file sharing solution that everyone uses

There isn't one. Sorry. All current solutions make design decisions in an attempt to balance out performance and usability.

Take away lessons from this:

Network File Sharing Issues: Conclusion

- Enough choices, just tell me what everyone does in practice

Elegant and efficient file sharing solution that everyone uses

There isn't one. Sorry. All current solutions make design decisions in an attempt to balance out performance and usability.

Take away lessons from this:

- Sharing files is hard business
- DO NOT simultaneously edit files
  - That's what file locking is for
  - That's what subversion and git are for

Examples of Network File Sharing Systems

NFS (Network File System)

- By Sun Microsystems, 1985
- First widely–used IP–based network file sharing
- We will discuss this in another lecture

AFS (Andrew File System)

- Designed for WAN–scale sharing

AFP (Apple Filing Protocol)

SMB (Server Message Block)

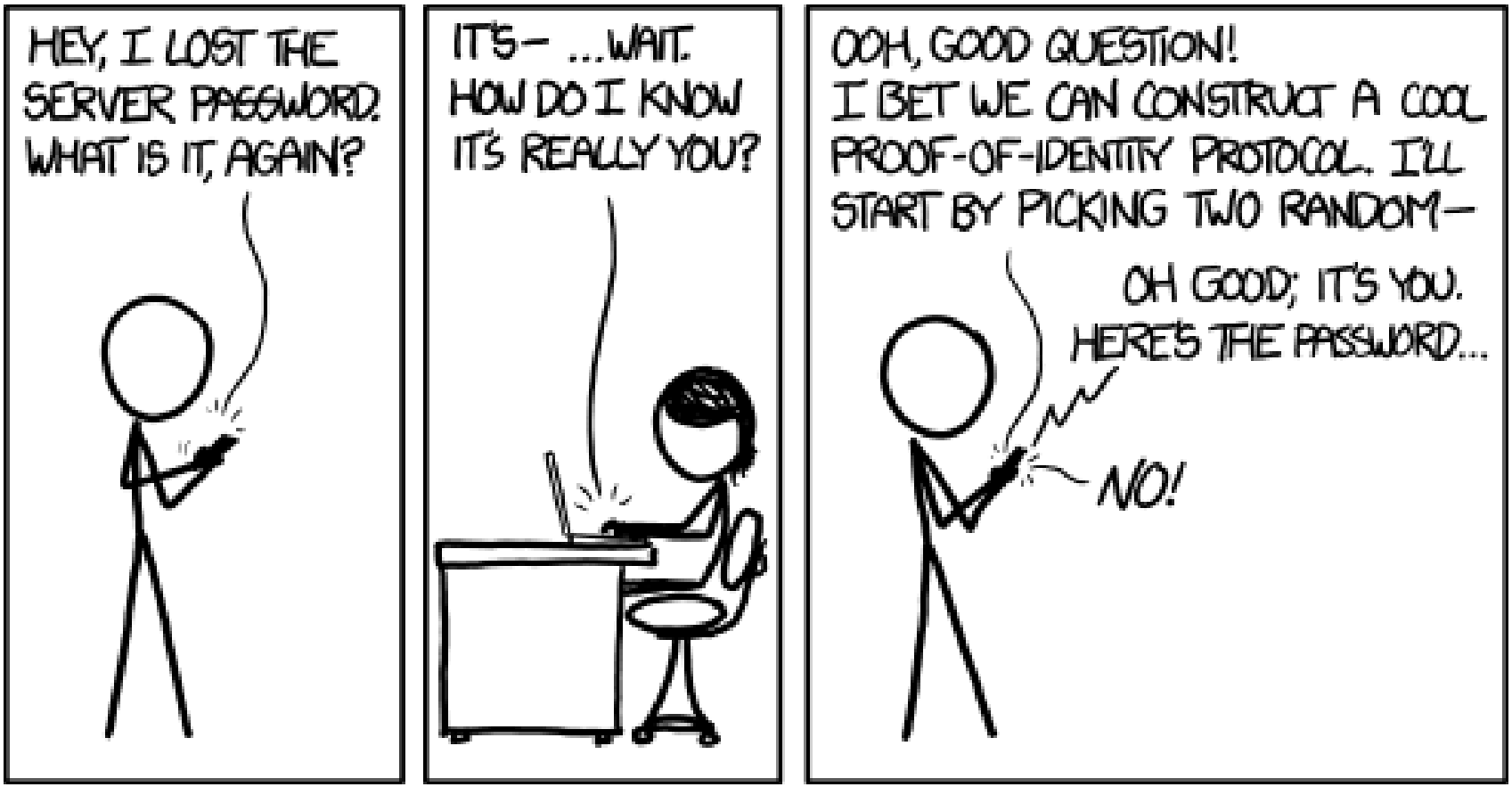
- Also known as CIFS
  (Common Internet File System)
- Will be discussed in this lecture

Network Identity — in general

What is this all about?

- How does a computer know that you are who you claim?
  - In particular, over a network
- Need these two things (at least):
  1. A unique identifier
     - Will be how users are identified
     - E.g., username, email address, account number...
     - Sometimes is aggregation of several things;
       E.g., username + domain gives unique email address
  2. An authentication scheme
     - How we verify users' identities
     - Can be based on what you know
     - Can be based on what you have

A relevant xkcd comic http://xkcd.org/1121



Authentication schemes: typical examples

- Passwords — something you know
  - Simple to implement
  - "Easy" to crack
  - More on this when we discuss security
- Tickets / certificates — something you have
  - Can be tricky to implement
  - Much harder to crack
  - Need "Authoritative broker" to assign certificates
    - A Trusted third party
- Host based
  - Simple, but easy to fake
- Host + port
  - Simple, but still possible to fake
- Two–factor authentication
  - E.g., ATM card + PIN

Network Identity Protocols

- NIS (Network Information Service)
  - Protocol for synchronizing information among hosts
  - Fairly easy to set up
- LDAP (Lightweight Directory Access Protocol)
  - Similar idea to NIS but more secure
    - . . . and more complex to set up
  - Version 3 is current: RFC 4511
  - Builds on dozens of other LDAP RFC's
- Active Directory. . .
- Kerberos
  - Developed at MIT
  - Uses "tickets" (provided by a trusted third party)
  - Allows machines to prove their identity to one another
  - Version 5 is current: RFC 4120
    - But there are several related RFC's

Protocols used for Windows Networking

Server Message Block

- Is a proprietary protocol — no RFC's
- Provides shared access to files, printers, and other devices
- Originally designed at IBM for a networked file system
- Originally designed to run on top of NetBIOS/NetBEUI
  - PC networking API used in 1980's
- 1996: Microsoft tried renaming it CIFS
  - Common Internet File System
  - Also added features (symbolic and hard links, . . .)
- Since Windows 2000: SMB can run directly on top of TCP/IP
- SMB 2.0 introduced with Windows Vista in 2006
- SMB 2.1 introduced with Windows 7
- SMB 3.0 introduced with Windows 8

SMB/CIFS services

1. File and print services
2. Authentication and Authorization
   - In a minute...
3. Name resolution
   - Originally: broadcast
     - Client broadcasts "where is machine Gamera?"
     - Gamera replies with an IP address
   - Later: machines send their name and IP address to a server
     - A NetBIOS Name Service (NBNS) server
     - Windows Internet Name Service (WINS): Microsoft's implementation of NBNS
   - Since Windows 2000: DNS can be used
4. Service announcement
   - So other users can browse a list of services (i.e., available shared folders and printers)

Active Directory

- A directory service created by Microsoft
  - Again, as in "telephone directory"
  - In use since 2000
- Uses LDAP versions 2 and 3
- Uses Kerberos
- Can use DNS
- What does it do, and why?
  - Need some terminology first...

Some Terminology

principal : an entity that can be authenticated

- Can be users, machines, services, processes
- Or groups of users, machines, services, processes
- Microsoft uses term security principal
- Are usually given a unique identifier for life
  - Microsoft: Security Identifier (SID)

Some Terminology

principal : an entity that can be authenticated

- Can be users, machines, services, processes
- Or groups of users, machines, services, processes
- Microsoft uses term security principal
- Are usually given a unique identifier for life
  - Microsoft: Security Identifier (SID)

Windows domain : a collection of security principals

- Share a central directory (a database)
- Idea:
  - Authenticate once when connecting to the domain (usually, when you login)
  - Then the SID is used to determine permissions within the domain

More Terminology

Active Directory : directory service for a Windows domain

- I.e., protocols to manage the central directory

Domain Controller : a server that "controls" a Windows domain

- I.e., a server for the Active Directory service
- Responds to authentication requests

Active Directory Domain Services (since 2008)

- The software that the Domain Controller runs
- From Microsoft:

. . . you configure the server with the role of domain controller by installing AD DS.

Some features of Active Directory

- Resources and security principals may be grouped into units
  - Makes administration easier
- Auditing
- Load balancing of Domain Controllers
- Support for MS Exchange
- Support for white page services

Active Directory: Summary

Sound complicated?

Active Directory: Summary

Sound complicated?

- It is.

Active Directory: Summary

Sound complicated?

- It is.

Good news:

Active Directory: Summary

Sound complicated?

- It is.

Good news:

- You will only use a tiny fraction of this for homework
- Not even close to setting up a full–fledged domain controller

Protocol Implementations

Samba

`http://www.samba.org`

What is Samba?

Samba

`http://www.samba.org`

What is Samba?

- An open–source implementation of SMB/CIFS

Samba

`http://www.samba.org`

What is Samba?

- An open–source implementation of SMB/CIFS
- Allows Linux machines to be clients in a "Windows Network"

Samba

`http://www.samba.org`

What is Samba?

- An open–source implementation of SMB/CIFS
- Allows Linux machines to be clients in a "Windows Network"
- Allows Linux machines to be servers in a "Windows Network"

Samba

`http://www.samba.org`

What is Samba?

- An open–source implementation of SMB/CIFS
- Allows Linux machines to be clients in a "Windows Network"
- Allows Linux machines to be servers in a "Windows Network"
- You will do this for homework

Samba History

1992 Andrew Tridgell (Australian)

- Wanted to connect DOS PC to UNIX server
- AND needed PC to run NetBIOS
  - Choice 1: use NFS (UNIX protocol)
    and get IP and NetBIOS to run together under DOS
  - Choice 2: write a server that works with NetBIOS
    and have it run on the UNIX machine
- Went with Choice 2
  - Wrote a packet sniffer
  - Reverse engineered the SMB protocol
  - Implemented it on UNIX machine
- UNIX machine behaved like PC file server
- Published code
  - No formal name, Tridgell called it
    "UNIX file server for DOS Pathworks"
  - Eventually renamed it "NetBIOS for UNIX"
- After a few updates, set it aside and ignored it

Samba History (2)

1994 Tridgell again

- Wanted to connect his wife's Windows machine to his Linux machine
- Dusted off his old code and tried it out . . .

1

Samba History (2)

1994 Tridgell again

- Wanted to connect his wife's Windows machine to his Linux machine
- Dusted off his old code and tried it out . . . it worked!
- Discovered NetBIOS and SMB documentation
  - Renamed it "SMBServer"
- Was contacted by a company about the software name
  - They held the trademark
- Tridgell checked the dictionary file[1] for words with "`smb`"
  - "Samba" was the first match

[1] Typically, `/usr/share/dict/words`

Samba History (2)

1994 Tridgell again

- Wanted to connect his wife's Windows machine to his Linux machine
- Dusted off his old code and tried it out . . . it worked!
- Discovered NetBIOS and SMB documentation
  - Renamed it "SMBServer"
- Was contacted by a company about the software name
  - They held the trademark
- Tridgell checked the dictionary file[1] for words with "`smb`"
  - "Samba" was the first match

Every good work of software starts by scratching a developer's personal itch.

[1] Typically, `/usr/share/dict/words`

Samba History (2)

1994 Tridgell again

- Wanted to connect his wife's Windows machine to his Linux machine
- Dusted off his old code and tried it out . . . it worked!
- Discovered NetBIOS and SMB documentation
  - Renamed it "SMBServer"
- Was contacted by a company about the software name
  - They held the trademark
- Tridgell checked the dictionary file[1] for words with "`smb`"
  - "Samba" was the first match

Every good work of software starts by scratching a developer's personal itch.

—Eric Raymond, "The Cathedral and the Bazaar"

[1] Typically, `/usr/share/dict/words`

Samba History (3)

1999 Samba 2.0 released

2003 Samba 3.0 released

- IT Week Labs:
  Samba 3 is 2.5x faster than Windows 2003 Server
- Veritest (hired by Microsoft):
  Windows 2003 Server is 1.6x faster than "Linux"
- See `http://www.kegel.com/nt-linux-benchmarks.html`

2012 Samba 4.0 released

- Complete reworking of the Samba code
- Samba can be an Active Directory Domain Controller

Samba project: summary

- Is large and active
  - Around 50 developers on the team
  - Including Andrew Tridgell
- License: GPL
- Goal: do anything a Windows Server can do
  - Windows Servers can do lots of things
  - That means things can get complicated
- Has an lots of documentation

- Good news — setup for homework is "simple"
  - Far from a full–fledged Domain Controller

Samba client (1)

How to access SMB/CIFS shared folders in Linux?

Samba client (1)

How to access SMB/CIFS shared folders in Linux?

File managers in Gnome and KDE

- GUI — do the usual

Samba client (1)

How to access SMB/CIFS shared folders in Linux?

File managers in Gnome and KDE

- GUI — do the usual

`smbtree`

- Command–line tool
- Displays a hierarchical diagram of available shares

Samba client (1)

How to access SMB/CIFS shared folders in Linux?

File managers in Gnome and KDE

- GUI — do the usual

`smbtree`

- Command–line tool
- Displays a hierarchical diagram of available shares

`smbclient`

- Command–line tool
- Works like old–style `ftp`
- Can `get` and `put` files

Samba client (2)

For the ultimate in transparency:

Samba client (2)

For the ultimate in transparency:

```
mount
```

Samba client (2)

For the ultimate in transparency:

`mount`

- Filesystem type is "`cifs`"
- Will mount a SMB/CIFS shared folder
  - The shared folder is specified as the "device"
  - Use the syntax: `//server/share`
    - `server` can be an IP address
      or a hostname that we can find an IP address for
- After it is mounted, shared files act just like local ones
- Check `man mount.cifs` for more information
  - Lots of CIFS–specific options
    - E.g., setting the user (and password) for the connection

Samba client (2)

For the ultimate in transparency:

`mount`

- Filesystem type is "`cifs`"
- Will mount a SMB/CIFS shared folder
  - The shared folder is specified as the "device"
  - Use the syntax: `//server/share`
    - `server` can be an IP address
      or a hostname that we can find an IP address for
- After it is mounted, shared files act just like local ones
- Check `man mount.cifs` for more information
  - Lots of CIFS–specific options
    - E.g., setting the user (and password) for the connection

```
prompt$  mount -t cifs //10.9.8.7/SharedDocs /mnt/winshares
```

Samba server

- Requires 2 or 3 daemons
  - Described on the next slide . . .
  - Can be started and stopped as usual
  - Can be set to start at boot time as usual
- Can be configured using SWAT
  - Samba Web Administration Tool
  - Uses a Web–based Interface (WUI?)
- Can be configured by hand
  - Put the "magic text" in the configuration file
  - Usually the config file is `/etc/samba/smb.conf`
    - May vary by distribution
  - What you should do for homework
  - This lecture covers basic settings only

Samba server daemons

`nmbd` "`nmb.service`"

- Handles name registration and resolution
- Is involved in network browsing

`smbd` "`smb.service`"

- Handles file sharing
- Handles printer sharing
- Manages local authentication

`winbindd` "`winbind.service`"

- Optional daemon
- Required for Windows NT4 or ADS domains
- Needed for "trust relationships"

Samba configuration file: format

- Lines starting with "#" or ";" are ignored
  - Used for comments
  - Do not put comments after non-comments on the same line
- Configuration file is divided into sections
  - Start of a section is indicated by text: `[sectionname]`
  - A section ends when the next section starts, or at end of file
- Remaining lines are configuration options
  - Apply to the section containing them
  - Indented for readability (not required)
  - Format is: `option name = value`

Example configuration file

/etc/samba/smb.conf

```
[global]
;    These options are part of section "global"
     workgroup = HOMEGROUP
     wins support = yes
[printers]
;    These options are part of section "printers"
     path = /var/tmp
     printable = yes
     min print space = 2000
[myshare]
#    These options are part of section "myshare"
     browsable = yes
     read only = yes
     path = /usr/local/samba/tmp
```

Sections in the configuration file

`[global]`

- Server-wide settings
- Default settings for everything else

`[printers]`

- Settings for sharing printers

`[homes]`

- Settings for sharing home directories
- Creates a share for each user's home directory
- Share name matches username

other sections

- Define shares (shared directories)
- Share name matches the section name

Some useful options

Global options

`workgroup`:	the workgroup name
`netbios name`:	server name (15 characters max)
`hosts allow`:	the hosts allowed to connect

Share options

`path`:	Directory that will be shared (for a disk share)
`read only`:	Should the share be read-only?
`writable`:	Should the share be writable (not read-only)?
`force user`:	Pretend specified user is accessing share
`force group`:	Pretend specified group is accessing share

See `/etc/samba/smb.conf.example` for more information

Samba authentication

- Need to specify users and their passwords to Samba server
  - For the authentication and authorization part of SMB
- These are stored securely in a password database (usually)

- Use `smbpasswd` utility to update this database
  - As ordinary user: to change their SMB password
    - Requires that `smbd` is running
  - As root: to add SMB users

```
prompt$ smbpasswd -a chuck
```

  - As root: to remove SMB users

```
prompt$ smbpasswd -x chuck
```

  - Other changes are possible — check your `man` pages

And now for something completely different

Automatically mounting SMB shares in Linux

Suppose we want to set up a Linux (client) machine as follows

- Mount SMB share `//server/share` to `/mnt/server`
- Have this happen "automatically at boot time"
- Have this "fail cleanly"
  - E.g., if the server is down, the client can still function but of course cannot access files under `/mnt/server`

How can we do this?

Automatically mounting SMB shares in Linux

Suppose we want to set up a Linux (client) machine as follows

- Mount SMB share `//server/share` to `/mnt/server`
- Have this happen "automatically at boot time"
- Have this "fail cleanly"
  - E.g., if the server is down, the client can still function but of course cannot access files under `/mnt/server`

How can we do this?

Add entry to `/etc/fstab`, e.g.,

```
//server/share    /mnt/server    cifs  rw,<options>  0 0
```

Automatically mounting SMB shares in Linux

Suppose we want to set up a Linux (client) machine as follows

- Mount SMB share `//server/share` to `/mnt/server`
- Have this happen "automatically at boot time"
- Have this "fail cleanly"
  - E.g., if the server is down, the client can still function but of course cannot access files under `/mnt/server`

How can we do this?

Add entry to `/etc/fstab`, e.g.,

```
//server/share    /mnt/server    cifs  rw,<options>  0 0
```

Is this a good idea?

Automatically mounting SMB shares in Linux

Suppose we want to set up a Linux (client) machine as follows

- Mount SMB share `//server/share` to `/mnt/server`
- Have this happen "automatically at boot time"
- Have this "fail cleanly"
  - E.g., if the server is down, the client can still function but of course cannot access files under `/mnt/server`

How can we do this?

Add entry to `/etc/fstab`, e.g.,

```
//server/share    /mnt/server    cifs  rw,<options>  0 0
```

Is this a good idea? NO

Why not `/etc/fstab` with network shares?

- Those entries are automatically mounted at boot time
- But what if the server is down at boot time?
  - Client will hang at boot time
  - Either indefinitely, or
  - Until the server comes up, or
  - Until the client gives up trying

Why not `/etc/fstab` with network shares?

- Those entries are automatically mounted at boot time
- But what if the server is down at boot time?
  - Client will hang at boot time
  - Either indefinitely, or
  - Until the server comes up, or
  - Until the client gives up trying
- So, bring the server up first every time?

Why not `/etc/fstab` with network shares?

- Those entries are automatically mounted at boot time
- But what if the server is down at boot time?
  - Client will hang at boot time
  - Either indefinitely, or
  - Until the server comes up, or
  - Until the client gives up trying
- So, bring the server up first every time?
  - Not a clean solution
  - What if we have two servers?
  - And each server is a client of the other?

Why not `/etc/fstab` with network shares?

- Those entries are automatically mounted at boot time
- But what if the server is down at boot time?
  - Client will hang at boot time
  - Either indefinitely, or
  - Until the server comes up, or
  - Until the client gives up trying
- So, bring the server up first every time?
  - Not a clean solution
  - What if we have two servers?
  - And each server is a client of the other?
- From an earlier lecture:
  - `/etc/fstab` is for devices that are always connected

Why not `/etc/fstab` with network shares?

- Those entries are automatically mounted at boot time
- But what if the server is down at boot time?
  - Client will hang at boot time
  - Either indefinitely, or
  - Until the server comes up, or
  - Until the client gives up trying
- So, bring the server up first every time?
  - Not a clean solution
  - What if we have two servers?
  - And each server is a client of the other?
- From an earlier lecture:
  - `/etc/fstab` is for devices that are always connected
- Ugh, so do we need to `mount` these by hand every time?

Why not `/etc/fstab` with network shares?

- Those entries are automatically mounted at boot time
- But what if the server is down at boot time?
  - Client will hang at boot time
  - Either indefinitely, or
  - Until the server comes up, or
  - Until the client gives up trying
- So, bring the server up first every time?
  - Not a clean solution
  - What if we have two servers?
  - And each server is a client of the other?
- From an earlier lecture:
  - `/etc/fstab` is for devices that are always connected
- Ugh, so do we need to `mount` these by hand every time?
  - Fortunately, no ...

Automounting

What is automounting?

- Filesystems are automatically mounted on demand
  - As compared to automatically mounted *at boot time*
- Filesystems are automatically unmounted after inactivity
- This can repeat
  - A filesystem may be re-mounted and unmounted several times

Automounting

What is automounting?

- Filesystems are automatically mounted on demand
  - As compared to automatically mounted *at boot time*
- Filesystems are automatically unmounted after inactivity
- This can repeat
  - A filesystem may be re-mounted and unmounted several times

How does this happen?

- An automounter daemon
- Two systems available
  - amd : Berkeley Automounter
    - Implemented in user space
  - autofs : used in homeworks
    - Requires kernel support
    - Also uses a user–space daemon

Configuring autofs

- The autofs daemon can be managed as usual
  - Service name is autofs.service
  - E.g., systemctl is-enabled autofs.service
- Configuration:
  - Put "magic text" in the appropriate file(s)
  - auto.master
    - Normally in /etc
    - The "main" configuration file
  - Also: lots of map files
    - These will make more sense after we discuss the main configuration file

Format of `auto.master`

Example `auto.master`

```
/foo      /etc/auto.foo   --timeout 60
/bar      /etc/auto.bar
```

- First column: family of mount points
  - E.g., there can be several mount points under `/foo`
- Second column: where to obtain the map for those points
  - Generic entry is of the form "`type:name`"
  - Supported types include:

    `file, program, yp, nisplus, userdir`
  - `/etc/auto.foo` really means `file:/etc/auto.foo`
- Third column: options (optional)
- In this example:
  - File `/etc/auto.foo` explains how to mount things under `/foo`

Format of maps

- First column: key
  - Acts as a "relative mount point"
  - "Family" + "key" gives the complete mount point
- Second column: mount options
  - Same as for `/etc/fstab`, except...
  - Need to use `-fstype` to specify filesystem type (unless it is an NFS share)
- Third column: location to mount from
  - Add a ":" in front if location starts with "/"

- Pro tricks
  - Can use "*" in the key field
    - Matches all keys
  - Can use "&" in the location
    - Replaced by the key
    - Makes sense with "*"

Complete example (1)

/etc/auto.master

```
/foo      /etc/auto.foo   --timeout 60
/bar      /etc/auto.bar
```

/etc/auto.foo

```
cd        -fstype=iso9660,ro   :/dev/cdrom
floppy    -fstype=auto         :/dev/fd0
```

- Device `/dev/cdrom` will be automounted to `/foo/cd`
- Device `/dev/fd0` will be automounted to `/foo/floppy`

Complete example (2)

/etc/auto.master

```
/foo      /etc/auto.foo   --timeout 60
/bar      /etc/auto.bar
```

/etc/auto.bar

```
*         -fstype=cifs,<options>  ://server/&
```

- SMB share `//server/xyz` will be automounted to `/bar/xyz` for any string `xyz`

How this should work

```
prompt$
```

(Insert CD into drive)

How this should work

```
prompt$ ls -aF /foo
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$
```

Automounter mounts `/dev/cdrom` to `/foo/cd`

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ sleep 90
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ sleep 90
```

Will wait here for 90 seconds

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ sleep 90
prompt$
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ sleep 90
prompt$ ls -F
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ sleep 90
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$
```

Device is still busy so it is was not unmounted

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ sleep 90
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ cd ..
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ sleep 90
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ cd ..
prompt$
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ sleep 90
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ cd ..
prompt$ ls -aF
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ sleep 90
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ cd ..
prompt$ ls -aF
./   ../  cd/
prompt$
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ sleep 90
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ cd ..
prompt$ ls -aF
./   ../  cd/
prompt$ sleep 90
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ sleep 90
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ cd ..
prompt$ ls -aF
./   ../  cd/
prompt$ sleep 90
```

Another wait for 90 seconds

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/         System/      mach_kernel
prompt$ sleep 90
prompt$ ls -F
EFI/         System/      mach_kernel
prompt$ cd ..
prompt$ ls -aF
./   ../  cd/
prompt$ sleep 90
prompt$
```

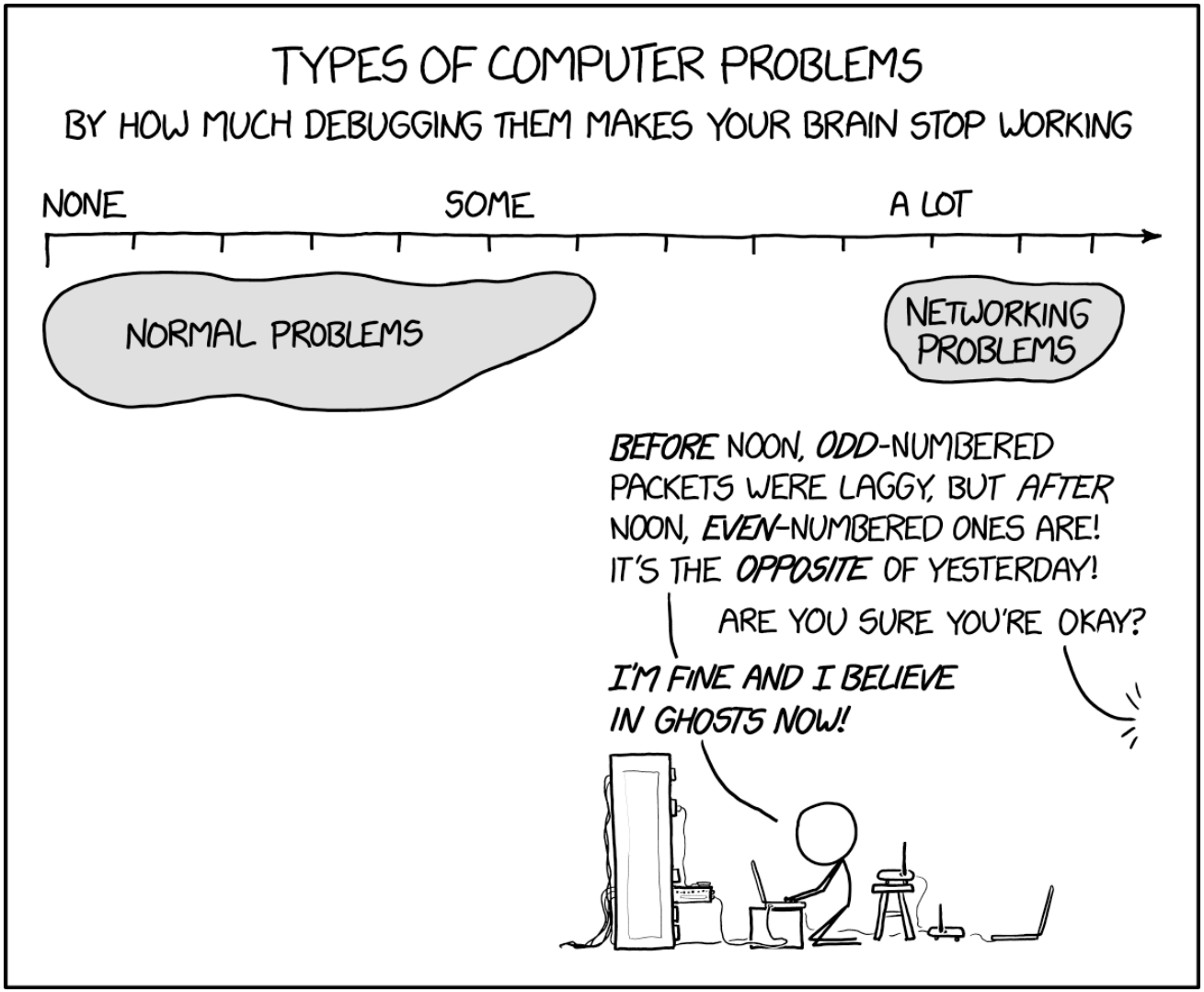
How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ sleep 90
prompt$ ls -F
EFI/          System/       mach_kernel
prompt$ cd ..
prompt$ ls -aF
./   ../  cd/
prompt$ sleep 90
prompt$ ls -aF
```

How this should work

```
prompt$ ls -aF /foo
./   ../
prompt$ cd /foo/cd
prompt$ ls -F
EFI/         System/      mach_kernel
prompt$ sleep 90
prompt$ ls -F
EFI/         System/      mach_kernel
prompt$ cd ..
prompt$ ls -aF
./   ../  cd/
prompt$ sleep 90
prompt$ ls -aF
./   ../
prompt$
```

An appropriate xkcd comic: http://xkcd.com/2259



End of lecture