Optimizing Mac OS X Server File Services

MacWorld, 2007.

Andrina Kelly
Nigel Kersten

Optimizing Mac OS X Server File Services

- FTP
  - Basic Configuration
  - Debugging

- AFP, NFS, SMB
  - Basic Configuration
  - Debugging
  - Optimizing

- Automounts
  - Dynamic versus Static mounts
  - NetInfo, LDAP, Flat Files

FTP: Background

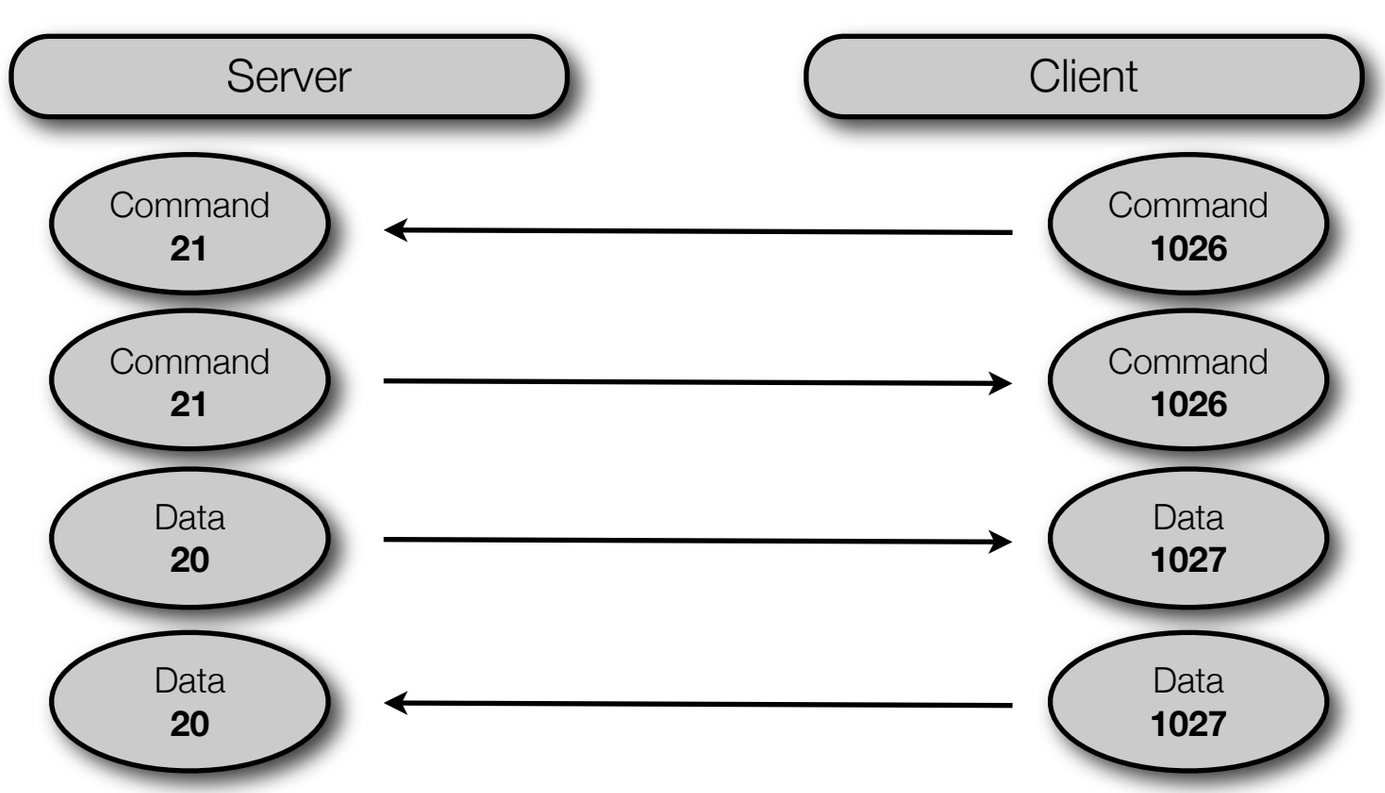
- Data transfer is **always** unencrypted
- Authentication, Standard versus Kerberos
  - Standard
    - Passwords are **always** unencrypted
    - Minimal client configuration
  - Kerberos
    - Passwords are secure
    - Requires client configuration
    - Requires GSSAPI capable FTP client
      - Fetch on OS X
      - Filezilla on Windows (anyone?)

FTP: Background

- TCP based service
- Uses two ports, a “command” port and a “data” port
  - Command port: 21
  - Data port: 20 (depends on “mode”...)
- Two modes: Passive and Active
  - “Active FTP vs. Passive FTP, a Definitive Explanation”
    - http://slacksite.com/other/ftp.html
- Cannot be used for network home directories
- Finder only offers read-only FTP connections

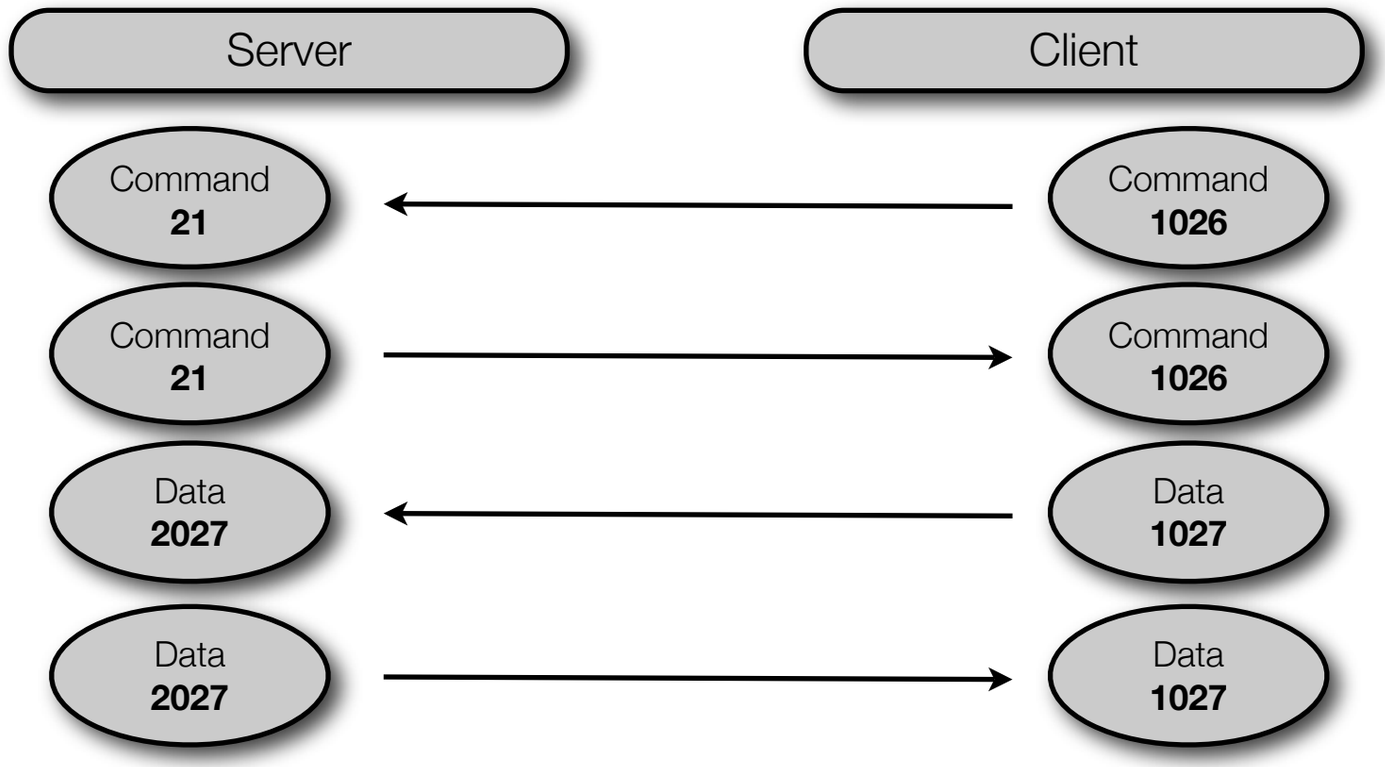
FTP: Active FTP

- Server: Command port: 21, Data port: 20
- Client: Command port > 1023, Data port > 1023
- Data connection is initiated from Server to Client
  - Potential problem for **client** side firewalls and NAT



FTP: Passive FTP

- Server: Command port: 21, Data port: > 1023
- Client: Command port > 1023, Data port > 1023
- Data connection is initiated from Client to Server
  - Potential problem for **server** side firewalls



FTP: Mac OS X Server

- Based on wu-FTPd (has been modified)
- Configuration:
  - Primary config file:
    - /Library/FTPServer/Configuration/ftpaccess
  - File conversion settings:
    - /Library/FTPServer/Configuration/ftpconversions
  - Per-host access settings:
    - /Library/FTPServer/Configuration/ftphosts
  - Per-user access settings:
    - /Library/FTPServer/Configuration/ftpusers

FTP: Defining Passive Port Ranges

- Allows more precise server-side firewall configuration
  - Edit /Library/FTPServer/Configuration/ftpaccess
  - Add this line **above** the “shutdown” line:

    ```
    passive ports   0.0.0.0/0  65500 65600
    ```

  - “0.0.0.0/0” is a wild-card
  - Different client addresses can be given different port ranges

FTP: Restricting Access

- Service Access Control Lists in Server Admin



WebDAV: Background

An extension of the HTTP protocol to allow reading *and* writing

OS X Server's Apache install provides WebDAV services.

Configure using "Realms" tab in Server Admin.

Can generally be seen as a more secure replacement for FTP

Cannot be used for network home directories

WebDAV: Considerations

- Use SSL/HTTPS locations for security.
- The Web Server process needs read/write access.
  - Traditionally difficult to mix POSIX/WebDAV privileges.
  - Use filesystem ACLs instead.
    - Give Full Access to the ‘www’ user.
- Can be used with traditional htpasswd authentication.

AFP: Background

Apple Filing Protocol. Currently the most “native” of the OS X network file service options. Modern AFP works exclusively over TCP, and should not be confused with the service discovery protocol “AppleTalk”.

AFP: Optimizing for heavy loads

- Increase server RAM
- Disable Secure Connections
- Minimize number of automounts
- Configure client-side afp_wan_threshold/quantum
- Increase AppleFileServer maxThreads setting
- Disable Session Caching/Reconnects
- Disable Network .DS_Store creation
- Network Home Directories
  - Redirect ~/Library/Caches for clients
  - Consider Portable Home Directories

AFP: Configure client-side afp_wan_threshold

- AFP clients check their latency to the AFP server with:
  - **afp_wan_threshold** Threshold before WAN connection assumed
  - **afp_wan_quantum** Packet size for WAN connections
- Default values can be too low
  - Default afp_wan_threshold: 30 (ms)
  - Default afp_wan_quantum values:
    - OS X 10.4.8 and up : 32768
    - OS X 10.4.7 and lower : 8192

AFP: Configure client-side afp_wan_threshold

- Changing the defaults:

```
defaults write -g com.apple.AppleShareClientCore -dict-add \
afp_wan_threshold -int 120

defaults write -g com.apple.AppleShareClientCore -dict-add \
afp_wan_quantum -int 131072
```

- LoginHook or MCX using Workgroup Manager.

AFP: Increasing maxThreads

- 10.4 spawns more AFP threads for each connection
- Default values can be far too low
  - Defaults:
    - OS X Server 10.4 (PPC) : 40
    - OS X Server 10.4 (Universal) : 200
- Apple suggest a maximum setting of 400

```
sudo serveradmin settings afp:maxThreads=400
```

AFP: Session Caching/Reconnects

- Allows sleeping clients to transparently reconnect
- Can be problematic in heavy load scenarios
- To disable:

```
sudo serveradmin settings afp:reconnectFlag=”none”
```

AFP: Disable Network .DS_Store creation

- .DS_Store files contain window view settings
- Corrupted versions can affect performance
- To disable on a per-user basis:

```
defaults write com.apple.desktopservices DSDontWriteNetworkStores true
```

- Alternatively use Workgroup Manager/MCX.

AFP: Redirect ~/Library/Caches

- Redirecting ~/Library/Caches for network home directories to a local location improves client performance significantly
- Network Home Redirector:
  - http://jochsner.dyndns.org/NHR/Welcome.html
- LoginHook method:
  - See macos-x-server list archives

AFP: Client-side debugging

- Server-side debug logs are inadequate
- Client-side debug logs are comprehensive
- To enable (debug levels range from 1 to 6):
- Edit /etc/syslogd.conf and add the line:

```
*.debug                                                    /var/log/debug.log
```

- Enable AFP debugging, create log file and restart syslogd

```
defaults write -g com.apple.AppleShareClientCore -dict-add afp_debug_syslog -int 1
defaults write -g com.apple.AppleShareClientCore -dict-add afp_debug_level -int 6
sudo touch /var/log/debug.log
sudo killall syslogd
```

AFP: Privilege Mapping

- AFP translates remote “effective” permissions to local “real” permissions
  - Comes into play with:
    - Local client users
    - No shared directory service
  - Was problematic for 10.4 clients with 10.3 AFP servers
  - To disable:

```
sudo serveradmin settings afp:noNetworkUsers = yes
```

  - Excellent Apple Techinfo article at:

    http://docs.info.apple.com/article.html?artnum=107485

AFP: More Tuning Articles

- Check http://www.afp548.com
- Look for:
  - “AFP: It ain’t so bad”
  - “AFP: It’s really quite good”
- Check Apple Techinfo articles
  - http://search.info.apple.com
- Check Apple Developer Connection
  - http://developer.apple.com
- Apple’s documentation is getting **much** better

SMB: Optimizations

- /etc/smb.conf changes

```
use kerberos keytab = true
oplocks = 1
strict locking = 0
socket options = TCP_NODELAY IPTOS_LOWDELAY
```

- Check “man smb.conf” for more socket options

- Important!
  - The OS X SMB **client** is **not** Samba.

SMB: Samba Web Administration Tool (SWAT)

- Add file "swat" in /etc/xinetd.d

```
# default: off
# description: SWAT is the Samba Web Admin Tool. Use swat \
#              to configure your Samba server. To use SWAT, \
#              connect to port 901 with your favorite web browser.
service swat
{
        port    = 901
        socket_type     = stream
        wait    = no
        only_from = localhost
        user    = root
        server  = /usr/sbin/swat
        server_args = -a /etc/smb.conf
        log_on_failure  += USERID
        disable = no
}
```

SMB: Samba Web Administration Tool (SWAT)

- Modify /etc/services to use swat
  - Original:

```
smpnameres      901/udp     # SMPNAMERES
smpnameres      901/tcp     # SMPNAMERES
```

  - Replacement:

```
swat      901/udp     # Samba Web Admin Tool
swat      901/tcp     # Samba Web Admin Tool
```

- /etc/smb.conf -> map to guest = Bad User
  - also found in Server Admin as "Allow Guest Access"
- Connect in web browser - http://localhost:901

NFS

The abbreviation for Network File System, NFS is a protocol suite developed and licensed by Sun Microsystems that allows different makes of computers running different operating systems to share files and disk storage

Daemons

- nfsd
- nfsiod
- mountd
- automount

Other Items of Note

- Portmap
- exports
- nfsstat
- iostat
- lsof
- showmount -e
- rpcinfo -p

Config Files

- sysctl

```
sudo sysctl -w net.inet.tcp.sendspace=65536
sudo sysctl -w net.inet.tcp.recvspace=65536
```

  Some additional gain can sometimes be seen by increasing these values to 262144. If one increases these values to 262144, one must also increase kern.ipc.maxsockbuf to something larger, such as 524288

- netinfo

```
nicl . -list /exports
nicl . -read /config/nfsd
```

- nfsd

Using with other *nix's

NFS Reserved Ports need to be configured - ports 1024 and above - on the non-OS X NFS Server

- On FreeBSD add to /etc/rc.conf -> **`nfs_reserved_port_only="NO"`**
  also -> **`sysctl vfs.nfs.nfs_privport=0`**

  $ rpcinfo -p server.example.com | grep nfs
  100003 2 udp 2049 nfs
  100003 3 udp 2049 nfs
  100003 2 tcp 2049 nfs
  100003 3 tcp 2049 nfs

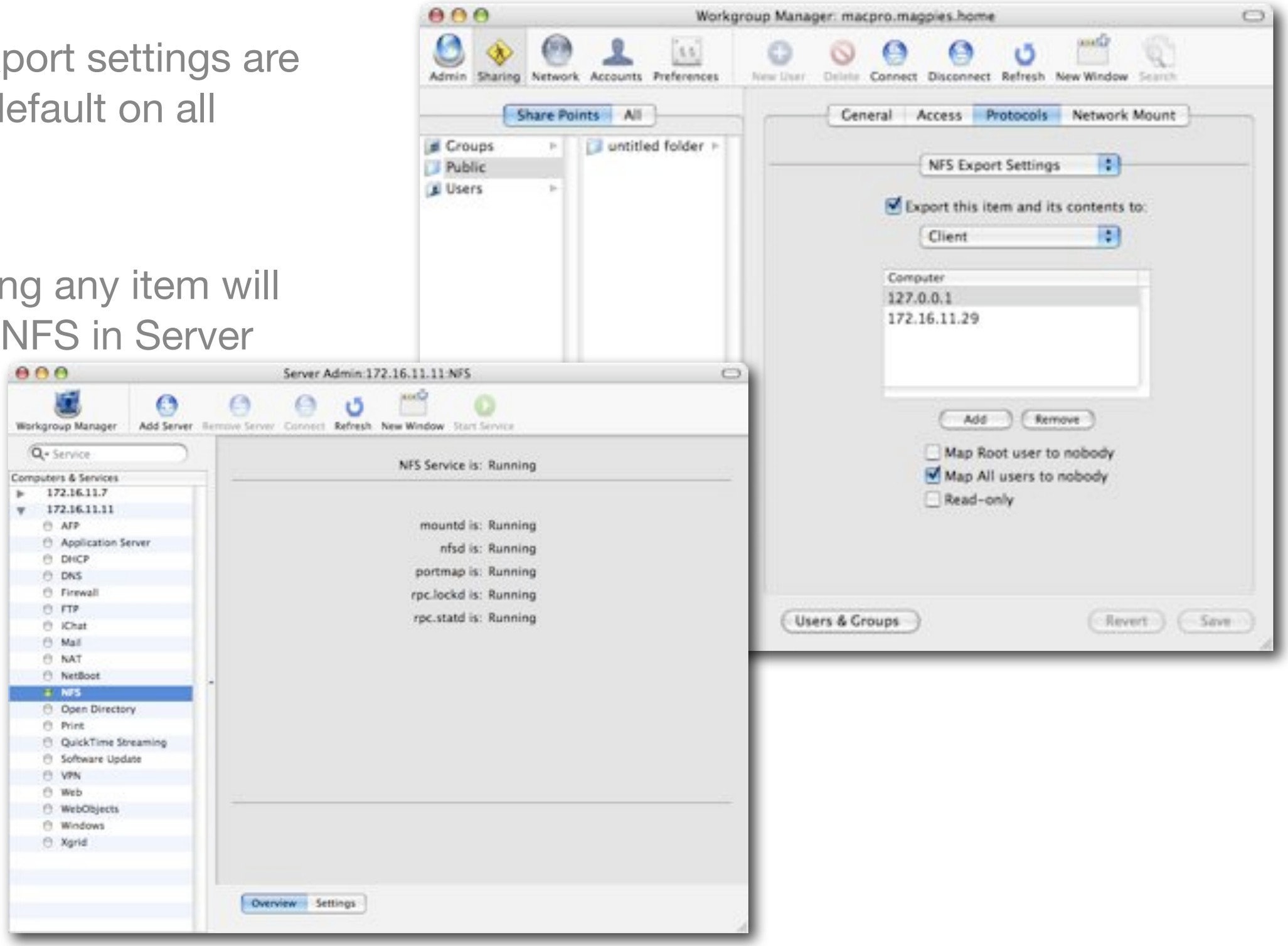
- On Linux distros add insecure to /etc/exports
  `/share 192.0.2.0/255.255.255.0(rw,root_squash,`**`insecure`**`)`

Logging

- /var/log/system.log
- nfsstat
- tcpdump
- ethereal / aqua ethereal / wireshark... whatever you want to call it now...

Enabling NFS

- NFS export settings are off by default on all shares
- Exporting any item will enable NFS in Server Admin



Some NFS (in)Security Notes...

- NFS was not designed with security in mind
- Configure NFS so that requests are only accepted from privileged system programs - i.e. map root to root
- Export file systems read-only wherever possible
- At your firewall, no **not** pass TCP port 111, UDP port 111, TCP port 2049 or UPD port 2049 (111 is the portmappers, 2049 is nfsd)
- Don’t enable NFS unless you really need it!

NFS Automounts on OS X

- Range of options...
  - Workgroup Manager
  - mountd mountmaps
  - NetInfo

Workgroup Manager

- Easily set up
- Easily modifiable
- Requires all machines bound to OD
- Advised that server with NFS mount is also bound to OD



mountd mountmap Automounts

- Has to be set up client-side
- Uses the built in NFS automount scrupt on all OS X machines in `/System/Library/StartupItems/NFS`

The mountmap file

This is a separate file that can be stored anywhere on your client file system

```
server1  -rw,intr,hard,timeo=600,-T,-r=32768,-w=32768   server:/mount
server2  -rw,intr,hard,timeo=600,-T,-r=32768,-w=32768   server2:/mount
```

that is:

```
name_of_the_mount  options_for_your_mount   Mount_point_on_NFS_Server
```

Look at man mount_nfs for more explanation of options

Telling mountd about your mountmap

- Make a backup of `/System/Library/StartupItems/NFS/NFS`
- Modify `/System/Library/StartupItems/NFS/NFS` to look to your mountmap file

Find the section that looks like this:

```
if [ "${AUTOMOUNT:=-YES-}" = "-YES-" ]; then
   automount -m /Network -nsl -mnt ${AUTOMOUNTDIR}
   ln -s /automount/Library /Network/Library
   automount -m /automount/Servers -fstab -mnt /private/Network/Servers \
            -m /automount/static -static -mnt ${AUTOMOUNTDIR}
   ln -s /automount/Servers /Network/Servers
```

And modify it to look like this:

```
if [ "${AUTOMOUNT:=-YES-}" = "-YES-" ]; then
   automount -m /Network -nsl
   automount -m /automount/Servers -fstab -mnt /private/Network/Servers \
              -m /automount/static -static -mnt /private/automount
   ln -s /automount/Servers /Network/Servers
   automount -m /mnt /System/Library/StartupItems/NFS/mountmap
```

NetInfo Automounts

- Has to be done client-side
- Fairly simple to administer - can be controlled with NetInfo Manager GUI or through `nicl` command line
- If you already have SSH or ARD enabled on client machine, deployment can be a scripted implementation (Send UNIX Command in the case of ARD)

nicl

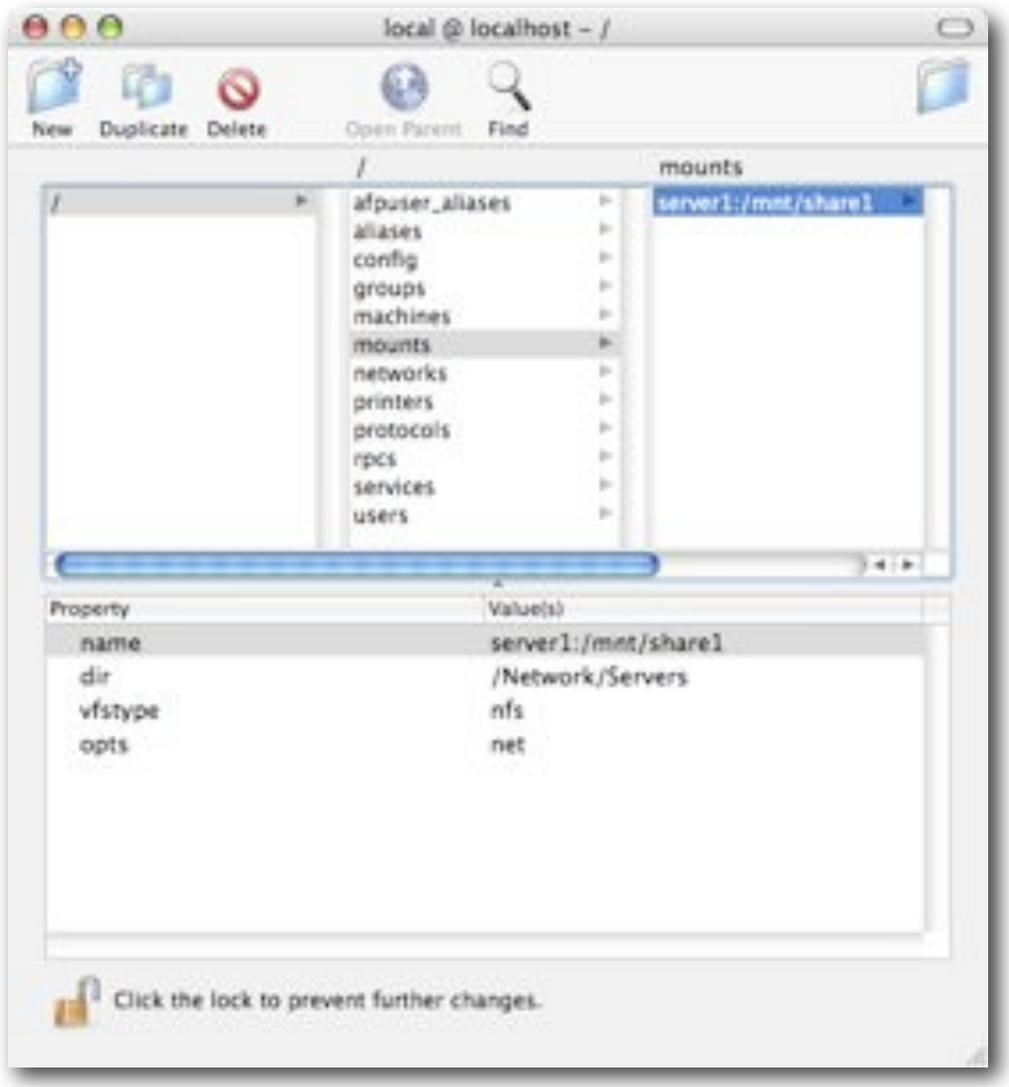
- Creating a new mount:

```
nicl . -create /mounts/$mount
nicl . -append /mounts/$mount vfstype nfs
nicl . -append /mounts/$mount opts net
nicl . -append /mounts/$mount dir /Network/Servers
```

- Be aware of escaping your “/’s” - i.e.:

```
nicl . -create /mounts/server1:\\/mnt\\/share1
nicl . -append /mounts/server1:\\/mnt\\/share1 vfstype nfs
nicl . -append /mounts/server1:\\/mnt\\/share1 opts net
nicl . -append /mounts/server1:\\/mnt\\/share1 dir /Network/Servers
```

NetInfo Manager



Restarting the automount process

```
kill -1 `cat /var/run/automount.pid`
```

Automounts

The automount daemon is responsible for automatically mounting network filesystems, and optionally to unmount them when idle.

Static versus Dynamic mounts

- Static mounts are mounted in /Network
  - Available immediately
  - Appropriate for content such as /Network/Library

- Dynamic mounts are mounted in /Network/Servers/server.name
  - Available when first accessed
  - Appropriate for Network Home Directories

- Network Home Directory automounts must either have:
  - Guest Access enabled *or*
  - Kerberos Authentication enabled

- Both kinds can be defined in:
  - NetInfo
  - Open Directory
  - Flat Files