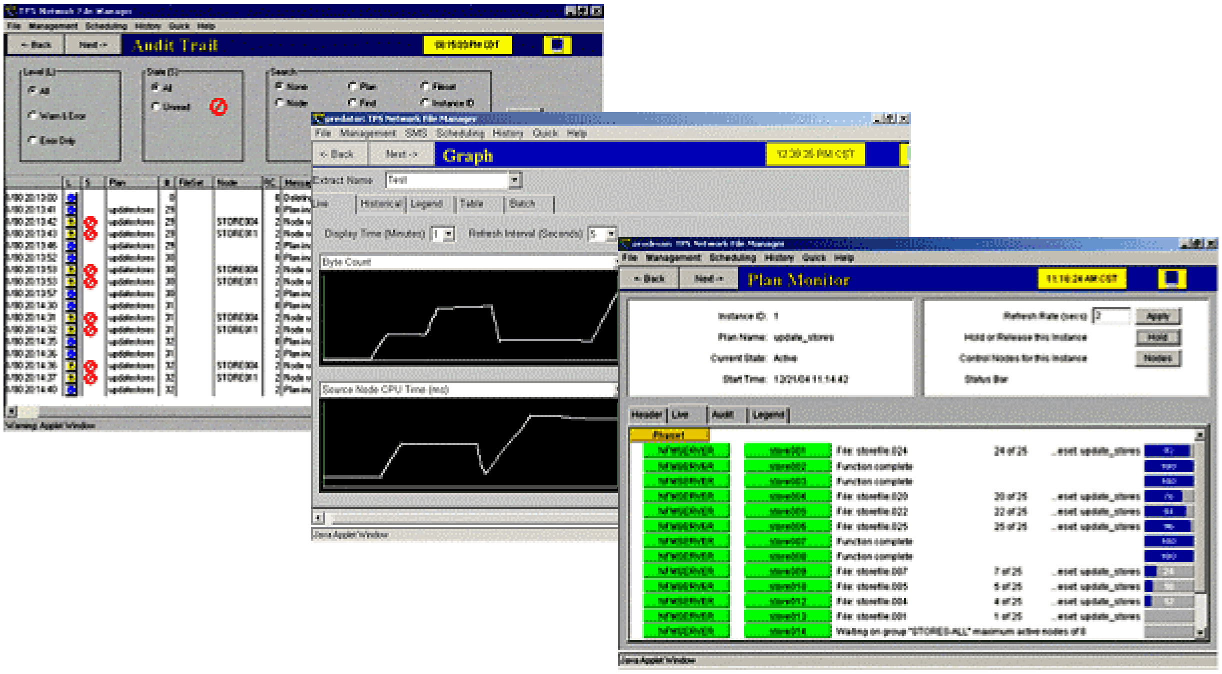
TPS®/NetWork File Manager

A Network Solutions Product



Training Guide

TPS® Systems, Inc.
14100 San Pedro Avenue, suite 600
San Antonio, TX 78232-4399
(210) 496-1984 voice
(210) 490-6805 fax
http://www.tps.com
support@tps.com

Software version 2.5.0
Manual revised 09-14-09

NFM Course Syllabus

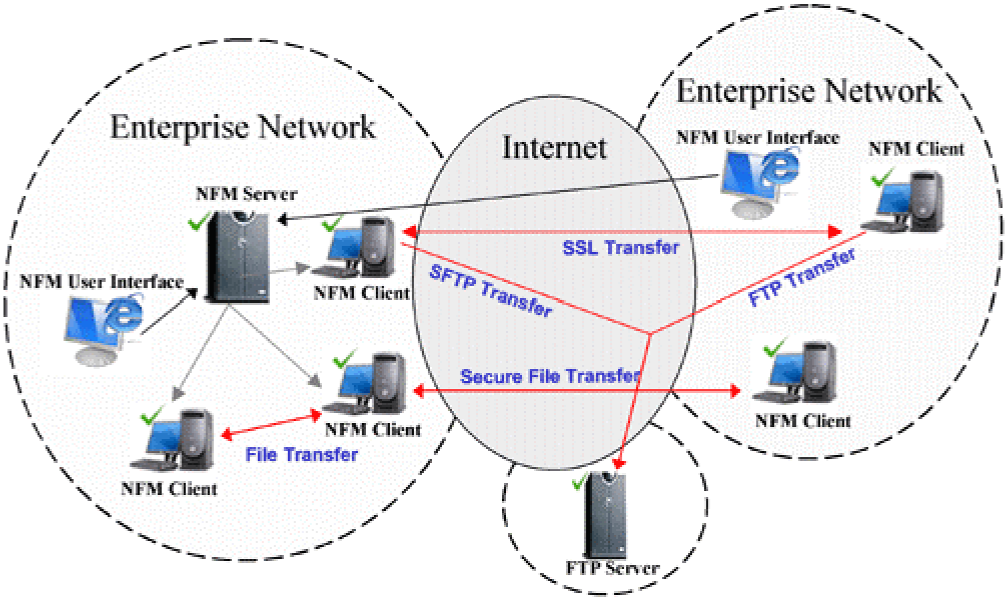
What is NFM?

TPS®/NetWork File Manager (NFM) is a sophisticated client-to-client file distribution system used to distribute files and execute scripts throughout a corporation's network. This robust, but easily, manageable system provides:

- Auditing capabilities
- Monitor and configure NFM using a browser interface
- Detailed error reporting
- Extensive logging capabilities.
- Secure (Encrypt) & compressed peer to peer file transfers
- Automated file transfers and program execution
- Checkpoint restart on file transfers
- Sophisticated scheduler to automate transfers and execution
- Script or batch execution
- Plan builder to control file transfers and program execution
- Monitoring capabilities for viewing plans in progress
- System security

NFM Topography

The NFM system can span across different networks to provide a single centralized control, auditing and monitoring while still benefiting secure peer-to-peer file distribution. This is an example of one NFM Data Delivery scheme.



NFM Components

There are five primary components to the NFM system (outlined in the diagram above):

NFM Server

This is the main “engine” of the NFM System. It contains the databases that comprise all NFM configurations (including node definitions, filesets records, plans, etc.) and audit records. The NFM Server is also responsible for initiating the communication between the NFM Clients, though files are transferred in a peer-to-peer environment. (more on NFM Server Components)

Supported Operating Systems: AIX, Windows, Linux
** Requires TPS Command Server be installed **

TPS Command Server

This service module enables the NFM GUI Interface to communicate with the NFM Server. (more on TPS Command Server)

Supported Platforms: Installed everywhere the NFM Server is installed

NFM Client

This refers to any computer, commonly referred to as an NFM node, that is running the NFM Client software. The NFM Client receives requests from the NFM Server to send and receive files as well as perform local program execution. Communication between the NFM Server and NFM Clients is socket based and occurs on TCP/IP port 8008 (default).

Supported Operating Systems: AIX, Windows, Linux, z/OS, OS/390, 4690, Sun, Stratus

Why use the NFM Client?

When possible, it is best to install the NFM Client and use it versus an NFM Non-Client (described further down). The NFM Client has several advantages these include: checkpoint restart, data encryption, command line execution, compression, remote command line, etc. (these features will be described later in this guide). NFM supports FTP and SFTP type access to computers. This is useful, in cases, where it is not possible to install the NFM Client (Internet site, an outside vendor, etc.). (more on NFM Client Component).

NFM Non-Client

This refers to any computer that is defined by the NFM server but is not running the NFM client software (for example: an FTP site). NFM supports certain types of FTP Servers. NFM Non-Clients rely on a different form of communication than NFM Clients based sockets.

Supported Non-Clients: FTP, SFTP (SSH), Kermit FTP

NFM GUI Interface

Once the NFM system has been properly installed, it is accessible through the NFM JAVA GUI Interface by either the NFM stand-alone windows program or via a JAVA enabled HTML browser (which requires setting up an HTTP Server). The NFM GUI communicates directly with the NFM Server to configure the system, schedule plans, view audits, etc. (more on choosing and setting up NFM GUI Interface).

Supported Platforms: OS capable of running Java; either via HTML Browser or a stand alone Windows application

NFM Terminology

There are some words or phrases that users new to the NFM system might not be familiar with. Before we get into the NFM system, here is a list of NFM terms that are used throughout this guide.

Node A network entity (usually a computer) that is a target of NFM activity.

Model Used to define parameters or options that are likely to be duplicated among several nodes. Each node references a particular model and inherits its default settings from it.

Fileset Contains a list of files to be transferred.

Plan A plan details an action or set of actions to be performed on a node or a group of nodes (for example: a file transfer, an execute, etc.). Each step or action is defined with a Plan Function and multiple functions grouped together in a Plan Phase. This provides a plan hierarchy and allows the user to develop conditional statements within the plan. For example, run this action if the action above it is successful or run this other action if the action above it is unsuccessful.

TPS®/NFM Training Guide

Section 1

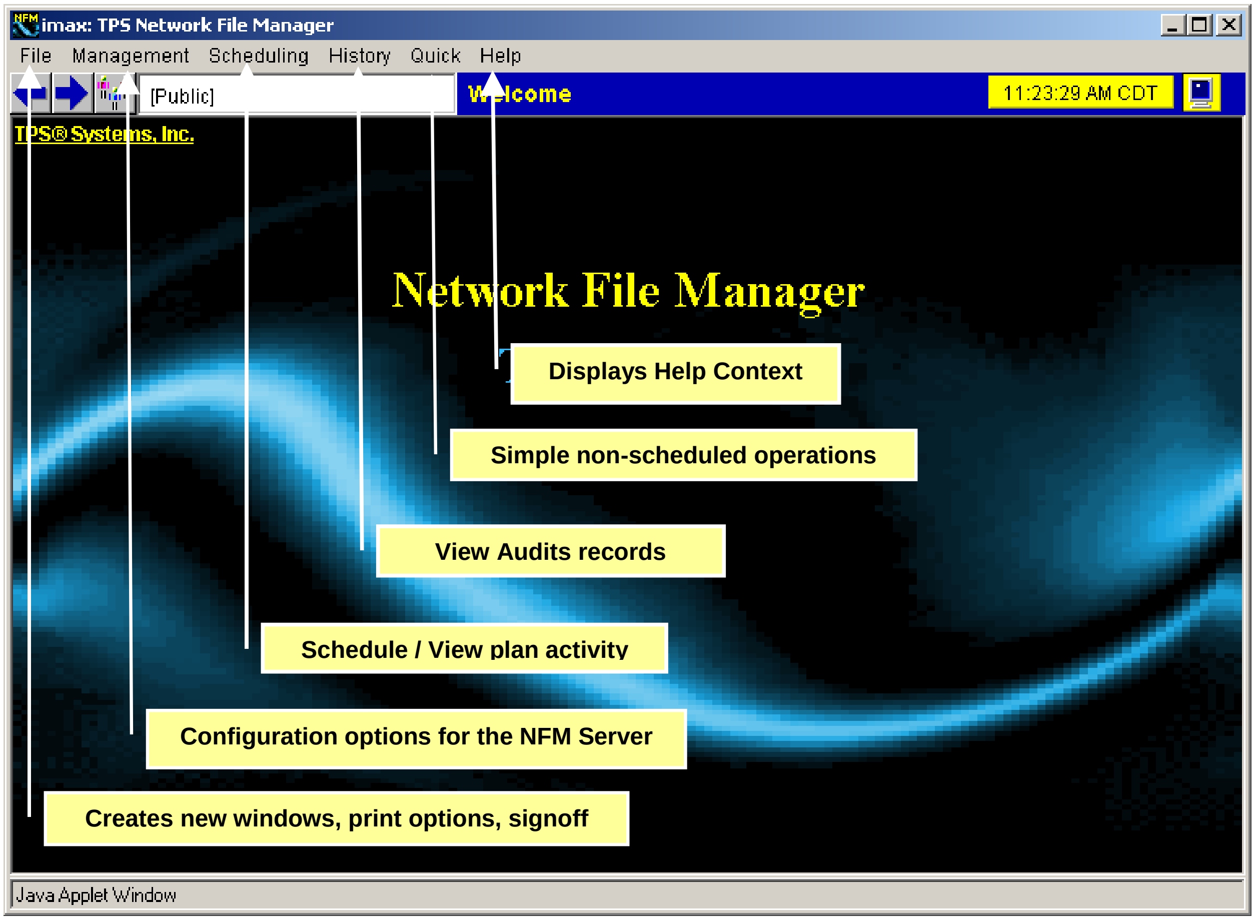
NFM GUI INTERFACE

Navigating the NFM GUI Interface

The NFM GUI Interface accesses the NFM Server through either a Java enabled browser or via the stand-alone Windows application (How to install and which one should I choose?).

NFM Toolbar Menus

Menus provide quick access to various NFM GUI screens. Each menu is grouped together in categories based on the menu items below it.



Displays Help Context

Simple non-scheduled operations

View Audits records

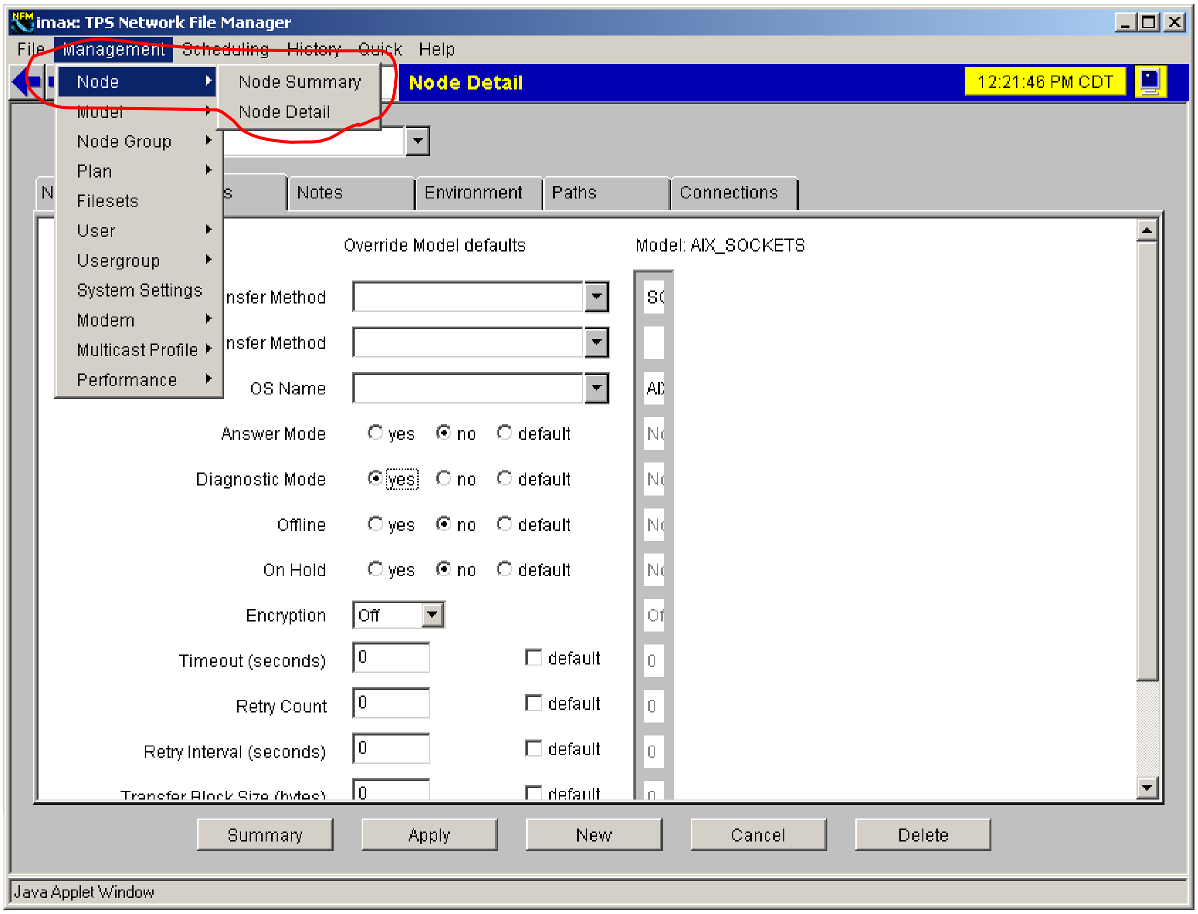
Schedule / View plan activity

Configuration options for the NFM Server

Creates new windows, print options, signoff

NFM Toolbar Sub Menus

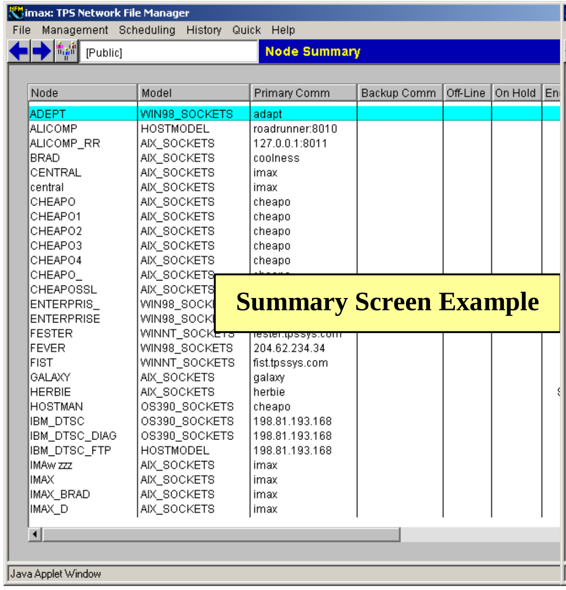
In general, most configurable items (Nodes, Users, Models, Plans, etc.) will offer a Summary Screen (lists all items) and a Detail Screen (for editing or creating a particular item).



Node Detail

Override Model defaults — Model: AIX_SOCKETS

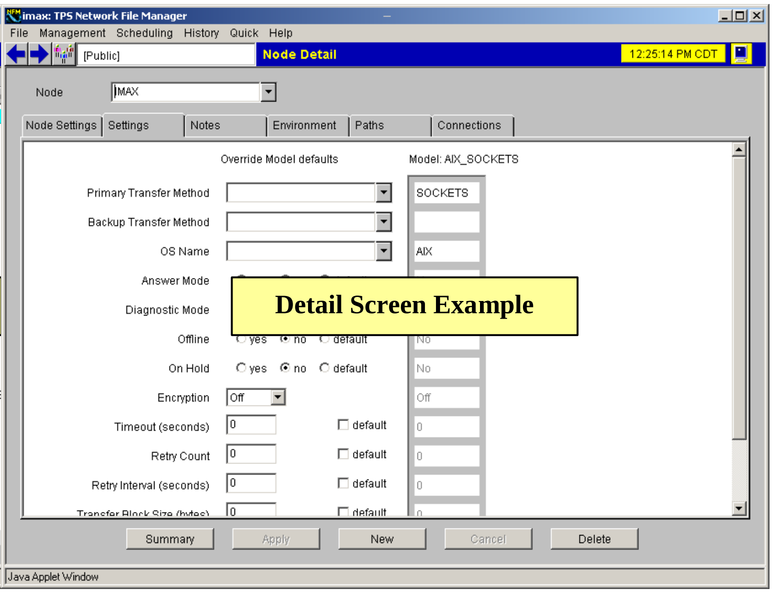
nsfer Method		SO
nsfer Method		
OS Name		AI
Answer Mode	○ yes ● no ○ default	No
Diagnostic Mode	● yes ○ no ○ default	No
Offline	○ yes ● no ○ default	No
On Hold	○ yes ● no ○ default	No
Encryption	Off	Of
Timeout (seconds)	0 □ default	0
Retry Count	0 □ default	0
Retry Interval (seconds)	0 □ default	0
Transfer Block Size (bytes)	0 □ default	0



Node Summary

Node	Model	Primary Comm	Backup Comm	Off-Line	On Hold	En
ADEPT	WIN98_SOCKETS	adapt				
ALICOMP	HOSTMODEL	roadrunner:8010				
ALICOMP_RR	AIX_SOCKETS	127.0.0.1:8011				
BRAD	AIX_SOCKETS	coolness				
CENTRAL	AIX_SOCKETS	imax				
central	AIX_SOCKETS	imax				
CHEAPO	AIX_SOCKETS	cheapo				
CHEAPO1	AIX_SOCKETS	cheapo				
CHEAPO2	AIX_SOCKETS	cheapo				
CHEAPO3	AIX_SOCKETS	cheapo				
CHEAPO4	AIX_SOCKETS	cheapo				
CHEAPO_	AIX_SOCKETS	cheapo				
CHEAPOSSL	AIX_SOCKETS					
ENTERPRIS_	WIN98_SOCK					
ENTERPRISE	WIN98_SOCK					
FESTER	WINNT_SOCKETS	fester.tpssys.com				
FEVER	WIN98_SOCKETS	204.62.234.34				
FIST	WINNT_SOCKETS	fist.tpssys.com				
GALAXY	AIX_SOCKETS	galaxy				
HERBIE	AIX_SOCKETS	herbie				
HOSTMAN	OS390_SOCKETS	cheapo				
IBM_DTSC	OS390_SOCKETS	198.81.193.168				
IBM_DTSC_DIAG	OS390_SOCKETS	198.81.193.168				
IBM_DTSC_FTP	HOSTMODEL	198.81.193.168				
IMAw zzz	AIX_SOCKETS	imax				
IMAX	AIX_SOCKETS	imax				
IMAX_BRAD	AIX_SOCKETS	imax				
IMAX_D	AIX_SOCKETS	imax				

Summary Screen Example



Node Detail

Node: IMAX

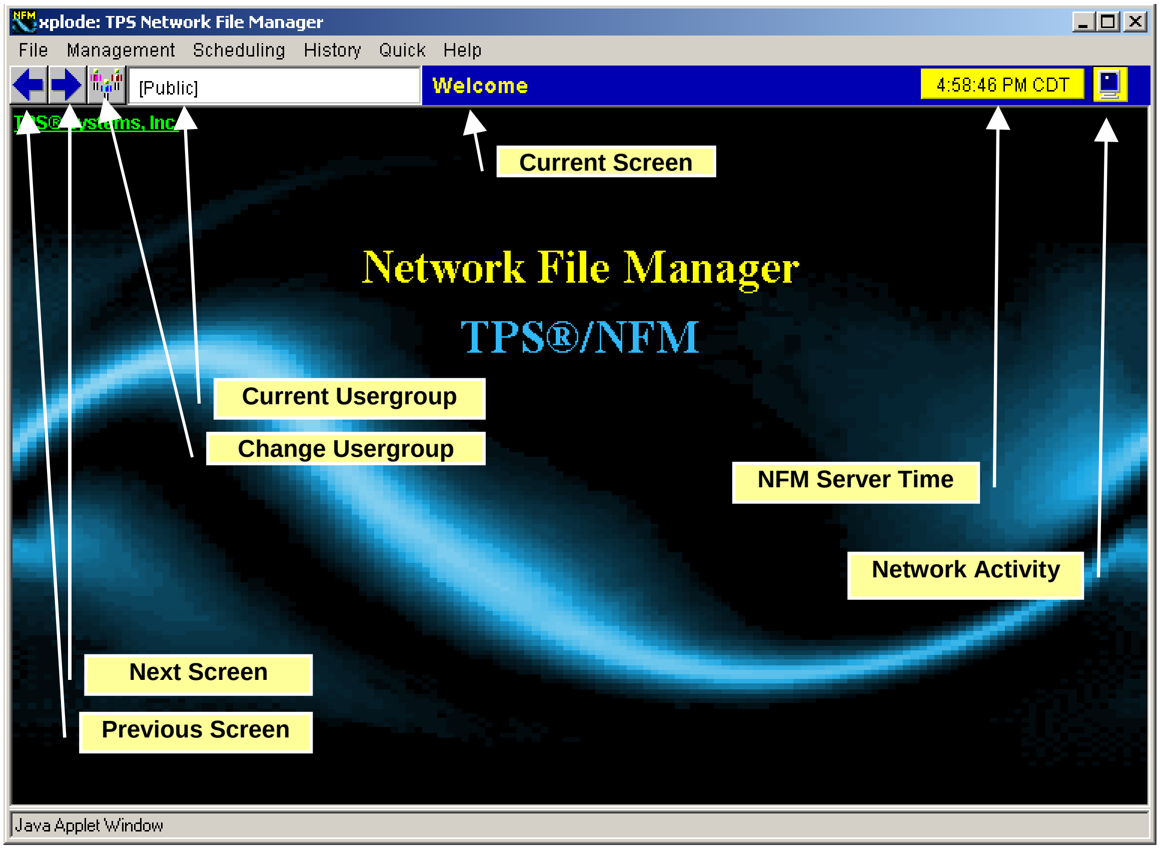
Override Model defaults — Model: AIX_SOCKETS

Primary Transfer Method		SOCKETS
Backup Transfer Method		
OS Name		AIX
Answer Mode		
Diagnostic Mode		
Offline	○ yes ● no ○ default	No
On Hold	○ yes ● no ○ default	No
Encryption	Off	Off
Timeout (seconds)	0 □ default	0
Retry Count	0 □ default	0
Retry Interval (seconds)	0 □ default	0
Transfer Block Size (bytes)	0 □ default	0

Detail Screen Example

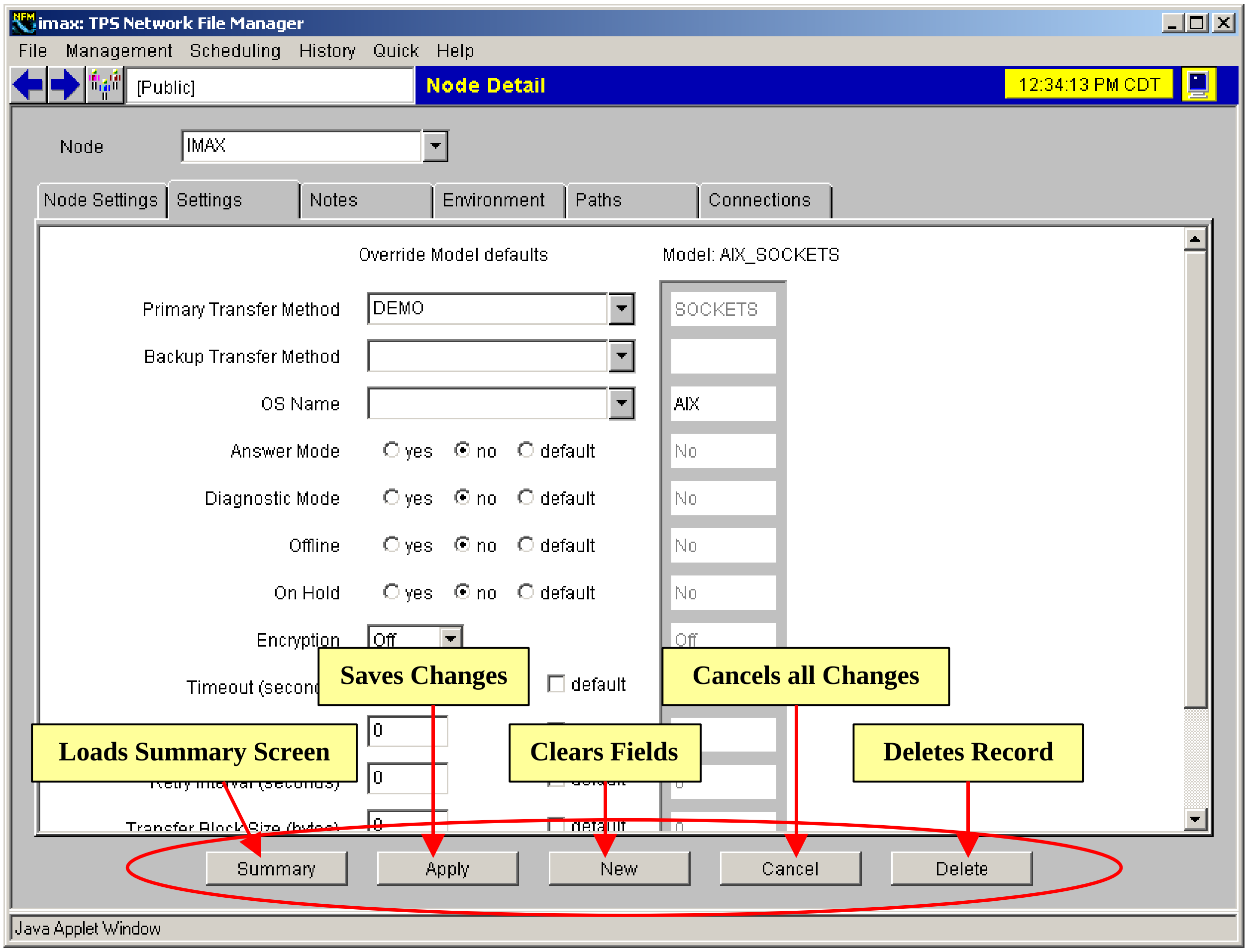
NFM Toolbar Icons / Buttons / Fields

Toolbar icons provide informational and navigation control



NFM GUI Buttons

Most GUI screens share buttons that are common to many NFM screens



Node: IMAX

Override Model defaults | Model: AIX_SOCKETS

Primary Transfer Method: DEMO | SOCKETS

Backup Transfer Method:

OS Name: | AIX

Answer Mode: ○ yes ● no ○ default | No

Diagnostic Mode: ○ yes ● no ○ default | No

Offline: ○ yes ● no ○ default | No

On Hold: ○ yes ● no ○ default | No

Encryption: Off | Off

Timeout (secon… ☐ default

0

Retry Interval (seconds): 0

Transfer Block Size (bytes): 0 ☐ default | 0

Saves Changes

Cancels all Changes

Loads Summary Screen

Clears Fields

Deletes Record

(Other buttons not shown here and specific to that screen will be described under that screen's description of fields / buttons)

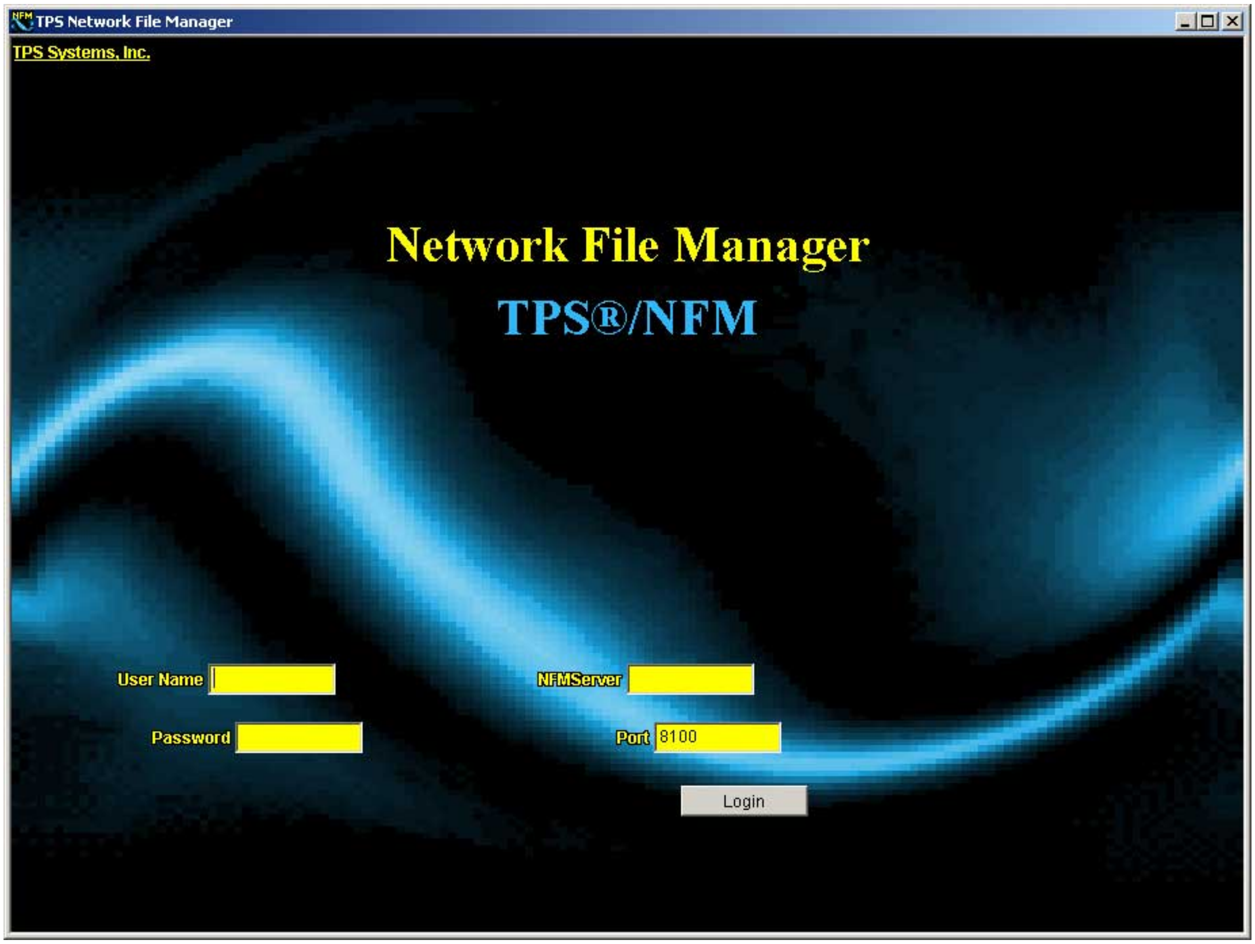
NFM GUI

Opening Screen

Once the NFM system has been installed, the NFM GUI provides an easy to configure interface to the NFM Server. There are two different GUI options, the Stand-alone Option and the Browser Option. The initial logging into the NFM system will vary slightly depending on which option you choose. (Installing the NFM GUI)

Stand-alone Option (Available only on Windows with Java installed)

After starting the Stand-alone NFM GUI program (*Start* | *Programs* | *TPS Systems* | *Network File Manager* | *NFM Java Server Interface*) an initial screen will prompt you for additional information.

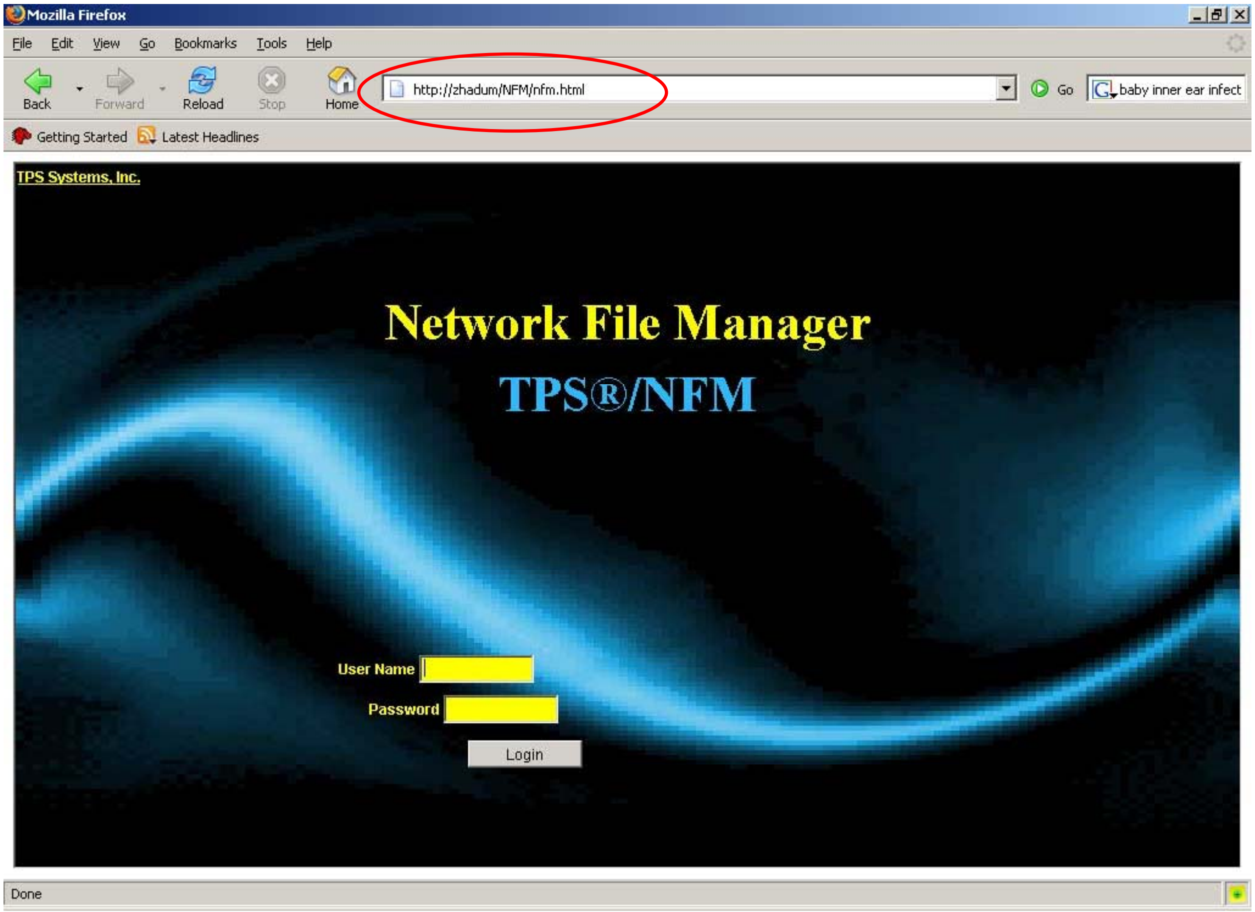


Port: 8100

Field Name	Field Description
User Name	Name of an existing user record that identifies an individual to the NFM system. This name identifies certain attributes and permissions for this user. (See "User" setup) ** NOTE: For first time users logging into the NFM System, NFM is shipped with a single user, 'root'. No password is required. You should change this password ASAP to prevent any one from logging in and having administrative access to the system.
Password	Password assigned to the user
NFM Server	IP address or name of the computer running the NFM Server.
Port	TCP/IP port used to communicate with the NFM Server. Default 8100.

Browser Option (Needs Java installed)

The browser should go to the URL that represents the NFM HTML page that exists on the NFM Server. The log in procedure is then identical to the Stand-alone Option, except the user will not need the NFM Server or Port fields. For the Browser Option, you may use any browser that is capable of running Java.



NFM GUI

BASIC CONFIGURATION

This section covers the minimum configuration needed to setup and use NFM. This covers the following GUI configurations / screens:

Nodes
Models
Node Groups
Filesets
Plans

Node

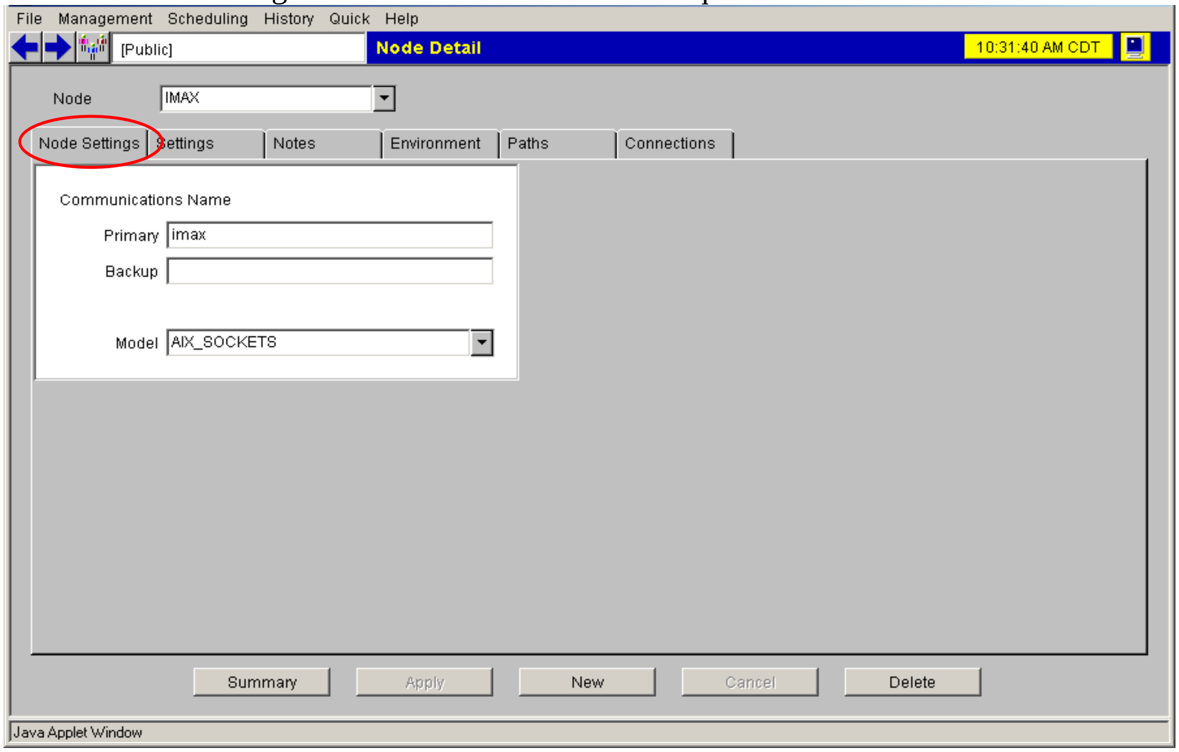
A node defines a network computer that can be used by the NFM system. It contains information that is unique to each computer; specifically the communications name, port number, IP address, etc. Each node belongs to a model, which is a template that will provide default values for many other fields in the node record (See 'Models').

Management | Node ▸ | Node Detail

(to access the Node screen)

Node Settings Tab

The Node Settings tab contains fields that are unique for each individual node.



Node Detail

10:31:40 AM CDT

[Public]

Node: IMAX

Communications Name

Primary: imax

Backup:

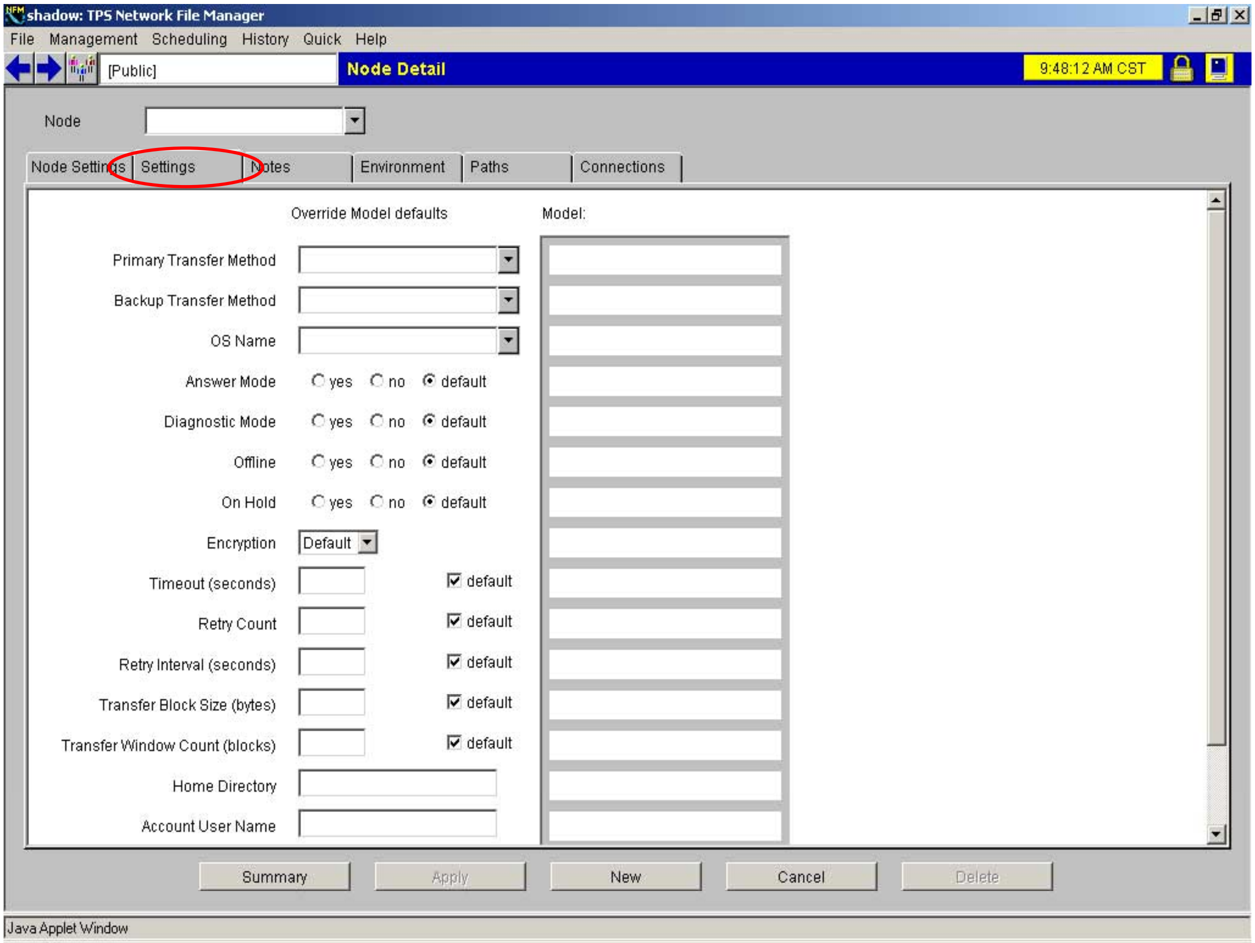
Model: AIX_SOCKETS

Field Name	Field Description
Primary	IP address or DNS hostname of the node. (*Required*)
Backup	IP address or DNS hostname of the node. This field is used if the Primary communications fails. The Backup Communications Name is used when the Retry Count and Retry Interval (defined under the Node's Settings tab) is exceeded.

Field Name	**Field Description**
Model	A model is a template used to define options of nodes that are likely to be duplicated among several nodes. For example: turning Encryption on for a model automatically turns it on for each node of that model type (this setting can be overridden in the node definition). (*Required*)

Settings Tab

The Settings tab allows for configuration of individual nodes.



Node

Override Model defaults | Model:

Primary Transfer Method

Backup Transfer Method

OS Name

Answer Mode ○ yes ○ no ● default

Diagnostic Mode ○ yes ○ no ● default

Offline ○ yes ○ no ● default

On Hold ○ yes ○ no ● default

Encryption Default

Timeout (seconds) ☑ default

Retry Count ☑ default

Retry Interval (seconds) ☑ default

Transfer Block Size (bytes) ☑ default

Transfer Window Count (blocks) ☑ default

Home Directory

Account User Name

Field settings on this screen can inherit values stored in the Node's Model. The 'Model: `AIX_SOCKETS` shows this Node's Model values. If the field in the 'Override Model defaults' column is blank or default is checked, the field will inherit the Model's corresponding field value. If you have a value in the 'Override Model defaults', the value will override whatever is in the Model.

Field Name	Field Description
Primary Transfer Method	Describes the primary type of connection used to communicate with the node (NFM Client nodes use socket based communications). (*Required*) Choices include: **SOCKETS** An NFM client node that is accessible on a TCP/IP LAN or virtual LAN. A node of this type is assumed to always be connected to the network. The NFM client software must be running on the node before any communications can be established. If this method is chosen, the Communications name field in the corresponding node record(s) should be set to the IP host name or IP address. **PPP** This method is also used for communicating to an NFM client node but in this case the node is not on a network. A dial-up connection must be made over a modem pair before communication takes place. The NFM server handles the dialing and disconnecting automatically to ensure proper sharing of single modem or modem pool when several connections are requested at the same time. If this method is chosen the Communications name field in the corresponding node record(s) should be set to a phone number. **FTP Server** The node is enabled as a standard FTP server. This can also designate an Internet ftp site if the Internet can be reached from the network. If this method is chosen the communications name in the corresponding node records should be set to the IP host name or IP address. Additionally, either the model or the node record must have the "Account User Name" and "Account Password" fields set to correspond with the FTP login values. **FTPS Server** The node is enabled as an FTPS server. This will automatically cause transfers with this node to use the FTPS protocol for doing secure FTP transfers. If this method is chosen the communications name in the corresponding node records should be set to the IP host name or IP address. Additionally, either the model or the node record must have the "Account User Name" and "Account Password" fields set to correspond with the F TP login values. **FTP** This node is enabled as a standard FTP

Field Name	Field Description
	server. When this selection is in use, the NFM client that is communicating to this node will use the external FTP client program that is native to the operating system. The "FTP Server" setting (above) uses the FTP protocol directly and is the preferred method to use, but this setting is also available for backward compatibility to existing customers. Also, use of the "Custom FTP" option (described in plan function settings), can only be used with this type of node. **SFTP (SSH)** The node is enabled as an SFTP (Secure FTP) server. This can be used to designate a secure Internet ftp site if the Internet can be reached from the network. If this method is chosen the communications name in the corresponding node records should be set to the DNS host name or IP address. Additionally, either the model or the node record needs the account name and password set to access the node. See the FTP and Secure FTP Nodes section for detailed information. **Kermit FTP** The node is enabled when using Kermit to secure server using Kermit FTP. This can be used to designate a secure Internet ftp site if the Internet can be reached from the network. If this method is chosen the communications name in the corresponding node records should be set to the DNS host name or IP address. Additionally, either the model or the node record needs the account name and password set to access the node. See the FTP and Secure FTP Nodes section for detailed information. **NVDM** The node is enabled as an NVDM remote node or catcher. Use of this type of node requires that the TPS®/SNA Primary product is also installed on the NFM server computer. (See "Appendix C" for more information on setting up an NVDM node.) **DEMO** Used for demonstration purposes only. No actual communications is established.

Field Name	Field Description
Backup Transfer Method	Describes the backup type of connection used to communicate with the node if the primary transfer method fails. Same choices as Primary Transfer Method apply.
OS Name	Operating System of the node. If the node is an FTP type node you may select generic (*Required*)
Answer Mode	The node will not initiate a socket connection. Very useful when a computer is outside a firewall and a file transfer needs to take place between this node and another node within a firewall. This will make the node within the firewall initiate a socket connection thus preventing the need to change firewall settings. (See Appendix B: Anatomy of a NFM File Transfer) (Does not apply to FTP/SFTP/Kermit)

<table>
<tr><th>Field Name</th><th>Field Description</th></tr>
<tr><td><b>Diagnostic Mode</b></td><td>Turns on the NFM trace feature. An excellent resource when trying to diagnose problems. Log files are kept in the following directories (See the Troubleshooting section for more info):
NFM Client logs:
<table>
<tr><th>Platform</th><th>Directory</th></tr>
<tr><td>Windows</td><td><code>C:\Program Files\TPS Systems\NFMClient\logs</code></td></tr>
<tr><td>UNIX</td><td><code>/var/tps/nfmc/logs</code></td></tr>
</table>
NFM Server logs:
<table>
<tr><th>Platform</th><th>Directory</th></tr>
<tr><td>Windows</td><td><code>C:\Program Files\TPS Systems\NFMServer\logs</code></td></tr>
<tr><td>UNIX</td><td><code>/var/tps/nfm/logs</code></td></tr>
</table>
</td></tr>
<tr><td><b>Offline</b></td><td>Disables the node from any plan activity.</td></tr>
<tr><td><b>On Hold</b></td><td>Puts the node on hold when a plan is run. Allowing a user to release the node manually from the Plan Activity or Plan Monitor screen.</td></tr>
<tr><td><b>Encryption</b></td><td>Automatically encrypt data going to, or coming from the node. This option is only available when transferring from NFM Client to NFM Client. If two nodes have this option set differently, the more secure setting is used. NFM currently supports the following encryption types: NFM (a TwoFish algorithm) and SSL. (See Encryption under Security Components)</td></tr>
<tr><td><b>Timeout (seconds)</b></td><td>Number of seconds a node has to reply to a request before communications is considered broken, thus generating an error. A value set too low can generate an error when the network might be busy. Try to avoid a timeout with less than 10 seconds unless you have a specific reason.</td></tr>
</table>

Field Name	Field Description
Retry Count	Number of times a connection to the node will be re-attempted after an initial failure. When the Retry Count has been exceeded, the Backup Communication Name (defined under the Node's Settings tab) will be tried.
Retry Interval (seconds)	How long to wait between connection attempts. It is only used if the retry count is non-zero (Not used for modem type connections). (***Dependant on Retry Count*)
Transfer Block Size (bytes)	The maximum block size used for sockets type file transfers to and from a given node (default is 32000). While transferring between nodes that have this field set different from one another, the smaller of the two is used.
Transfer Window Count (blocks)	This field along with "Transfer Block Size" above, can be set to enable a pacing feature of NFM that can help keep NFM from flooding a network with file transfer data. Normally, NFM transfers files as fast as the network will allow. This can have the adverse effect of choking off other network activity. This is especially visible when transferring to and from a node group.
Home Directory	Restricts access on a node for use with Quick Functions.
Node Password / Retype Password	An optional password may be used to restrict access to an NFM node. The password designated here must match a password that is configured on the NFM Client node (see "Setting a Node Password on the NFM Client" below).
Account User Name / Account Password	Used when the node type requires a login to be supplied when performing activity with the node. Account User Name applies to FTP and SFTP nodes. Account Password only applies to an FTP type node.
Nfmi User Name / Nfmi Password	This is the login to use for any remote commands issued from this node using the NFM remote server interface

Setting a Node Password on the NFM Client:

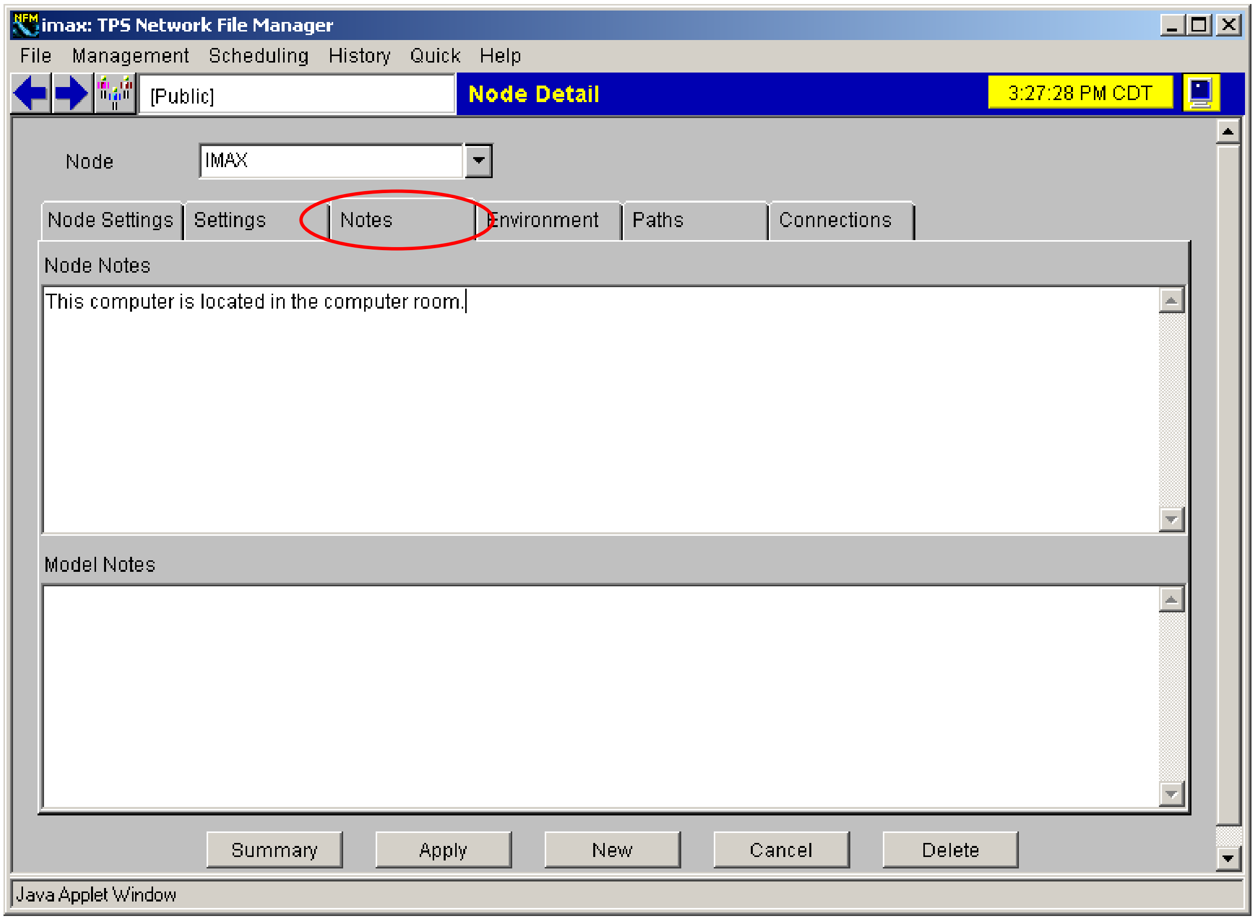
This task is accomplished using either the nfmi program at the NFM client node site using the following syntax:

nfmi -p
[or]
nfmi –P password

The first method will prompt you to enter and retype the password. After setting or changing the node password, the NFM Client MUST be stopped and restarted before the change will take effect.

Notes Tab

The Notes tab provides a text window for storing text information specific to this node.



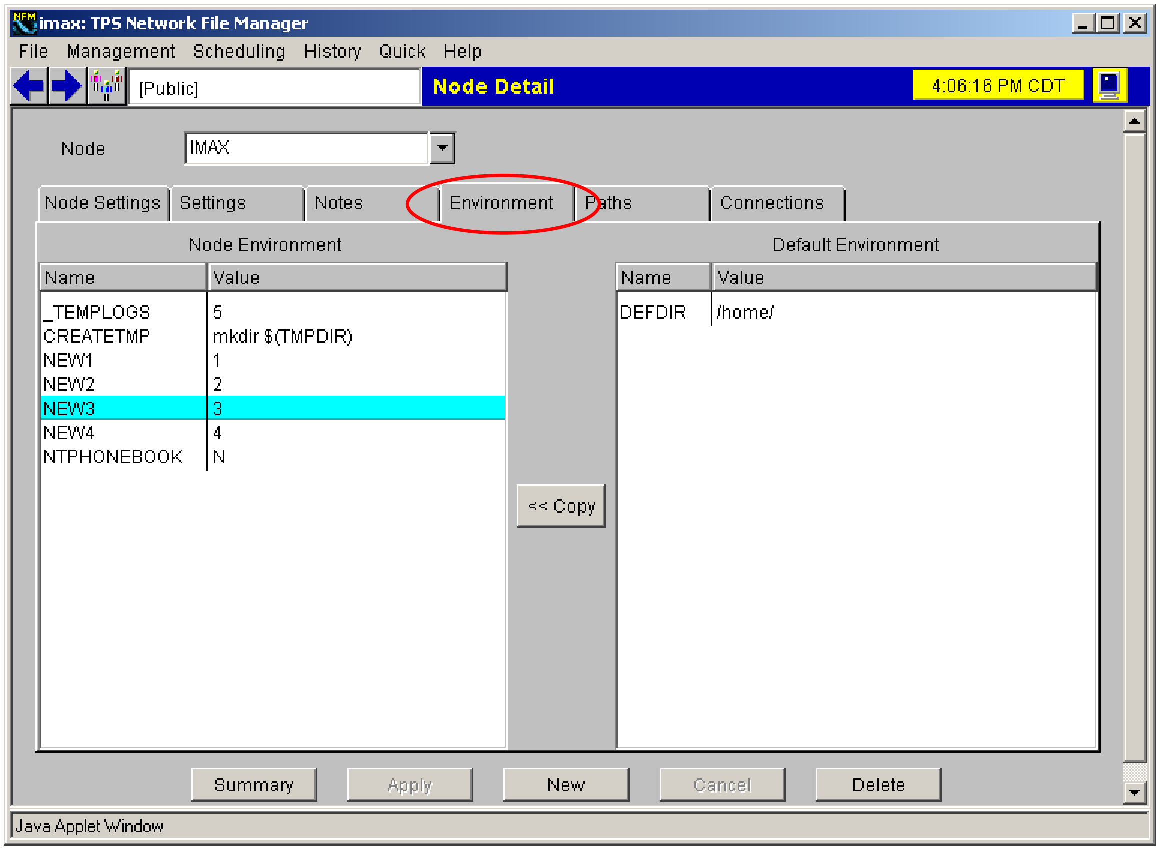
Node: IMAX

Node Notes: This computer is located in the computer room.

Field Name	Field Description
Node Notes	Text area for the user to type useful information pertaining to the node.
Model Notes	Text area the user typed in from the model screen.

Environment Tab

The Environment tab allows users to create custom environment variables.



Node: IMAX

Node Environment

Name	Value
_TEMPLOGS	5
CREATETMP	mkdir $(TMPDIR)
NEW1	1
NEW2	2
NEW3	3
NEW4	4
NTPHONEBOOK	N

Default Environment

Name	Value
DEFDIR	/home/

Environment variables are a powerful feature within NFM. They provide a flexible way to pass information to a Fileset or Plan. There are two types of environment variables available, user defined (defined on this screen or the Model screen) and NFM provided (for more information).

Environment values on this screen inherit values stored in the Node's Model. The 'Default Environment' shows this Node's Model Environment names and values. If an Environment name in the 'Default Environment' matches a 'Node Environment', the 'Node Environment' value will override whatever is in the Model.

Why use environment variables?

Environment variables provide a means of customizing filenames (within Filesets) and functions (within Plans) so they can perform different operations based on the current node they are working with. This gives the user the flexibility to write a generic plan (template) and reuse it for different purposes by changing environment variables.

How to use environment variables

Environment variables can be used in two different ways. Variables can be referenced in an executed command of a plan function, as well as in the prefix and filename fields of a fileset. This is done in the form $(VARIABLE) inserted anywhere in the text of these fields. In this case the substitution is done on the NFM server prior to the command or filenames being used during plan execution. All NFM environment variables are also passed down to the NFM client during a plan execute type function, and are available in local context as real operating system environment variables.

Example:

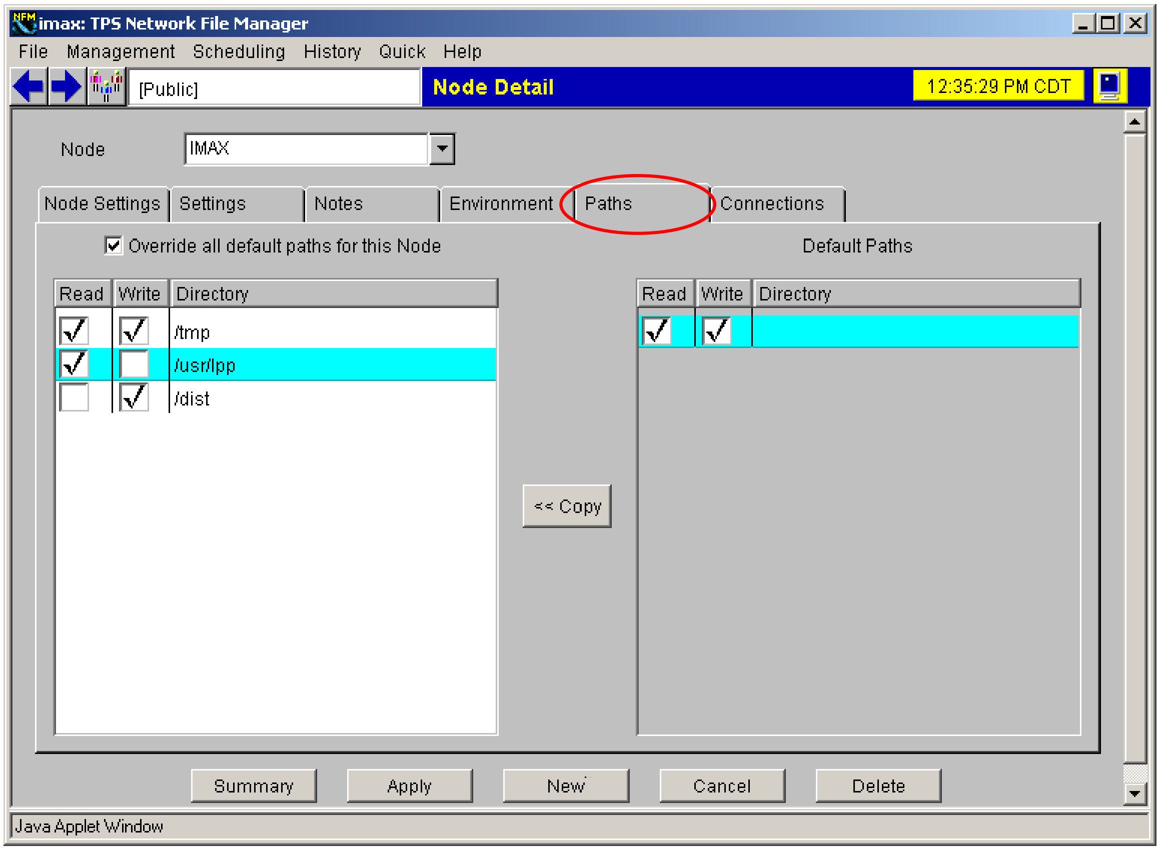
I define an environment variable called `DEFDIR` that I will use in my fileset to specify where to place files. On 10 Windows nodes, I define this variable as DEFDIR=/temp in the common model record used by all 10 nodes. On my 20 UNIX nodes, I define this variable as `DEFDIR=/home`. In the fileset, I specify $(DEFDIR) in my rename prefix field. When the plan is run, the `$(DEFDIR)` will be substituted with the appropriate value based on the node's `DEFDIR` environment variable.

The NFM provided environment variables are automatically generated during a plan execution.

(see Environment Variables for more information)

Paths Tab

The Paths tab maintains a list of path permissions associated with this node. Path permissions allow NFM to restrict or permit file transfer access to the specified directories.



[Public]

Node Detail

12:35:29 PM CDT

Node: IMAX

☑ Override all default paths for this Node

Read	Write	Directory
✓	✓	/tmp
✓	□	/usr/lpp
□	✓	/dist

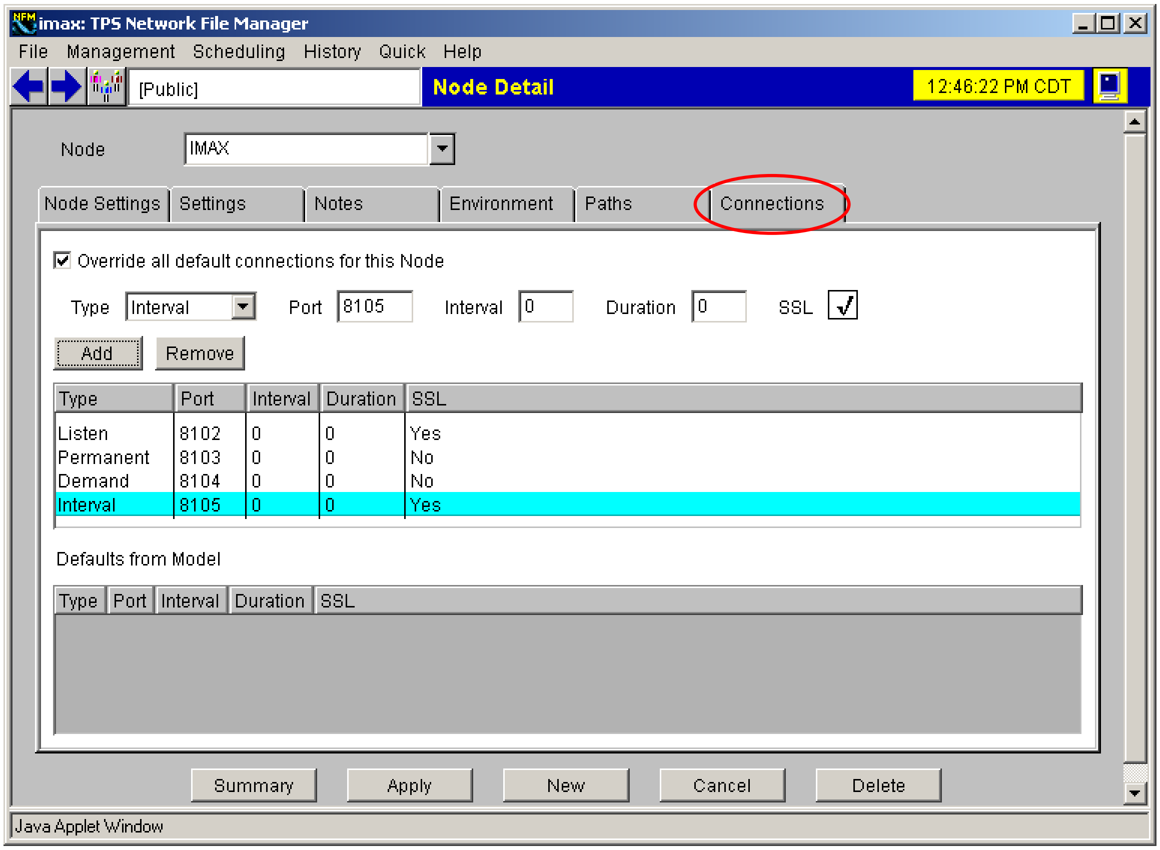
Default Paths

Read	Write	Directory
✓	✓	

See Path Permissions under Security Components for more detail.

Connections Tab

The Connections tab allows the user to customize socket type nodes (NFM Client).



Node: IMAX

☑ Override all default connections for this Node

Type: Interval Port: 8105 Interval: 0 Duration: 0 SSL: ☑

Type	Port	Interval	Duration	SSL
Listen	8102	0	0	Yes
Permanent	8103	0	0	No
Demand	8104	0	0	No
Interval	8105	0	0	Yes

Defaults from Model

Type	Port	Interval	Duration	SSL

It is not usually necessary to add connection records. NFM uses default connections for sockets type nodes as follows:

A non-SSL type node listens for a socket connection on the default port 8008.
An SSL type node listens for an SSL encrypted socket connection on the default port 8009.

NFM Clients, by default use a 'Listen Type' meaning the connection is initiated from the NFM Server to the node. Sometimes, you might need to use another type of connection, a different port or use SSL. Theses connections are:

Field Name	Field Description
Type	Choices include: **Listen** The node will listen for a socket connection on the specified port (Used when you want to override the default 8008). **Permanent** The node will make a permanent socket connection to the NFM server on the specified port when the NFM client program is started. **Demand** ** This connection type is not yet supported.** **Interval** ** This connection type is not yet supported.**
Port	Specifies which TCP/IP port to listen or connect to.
Interval	** This field is not currently implemented. **
Duration	** This field is not currently implemented. **
SSL	Indicates an SSL type connection.

If the "Override all default connections for this Node" is checked, then connections listed on the top will override the same connections on the bottom "Default from Model".

See SSL Type Connections under Security Components for more detail.

If you wish to use, add, or edit a connections entry, you will also need to configure or change that node's configuration file to match. See NFM Client Configuration File for more information

Model

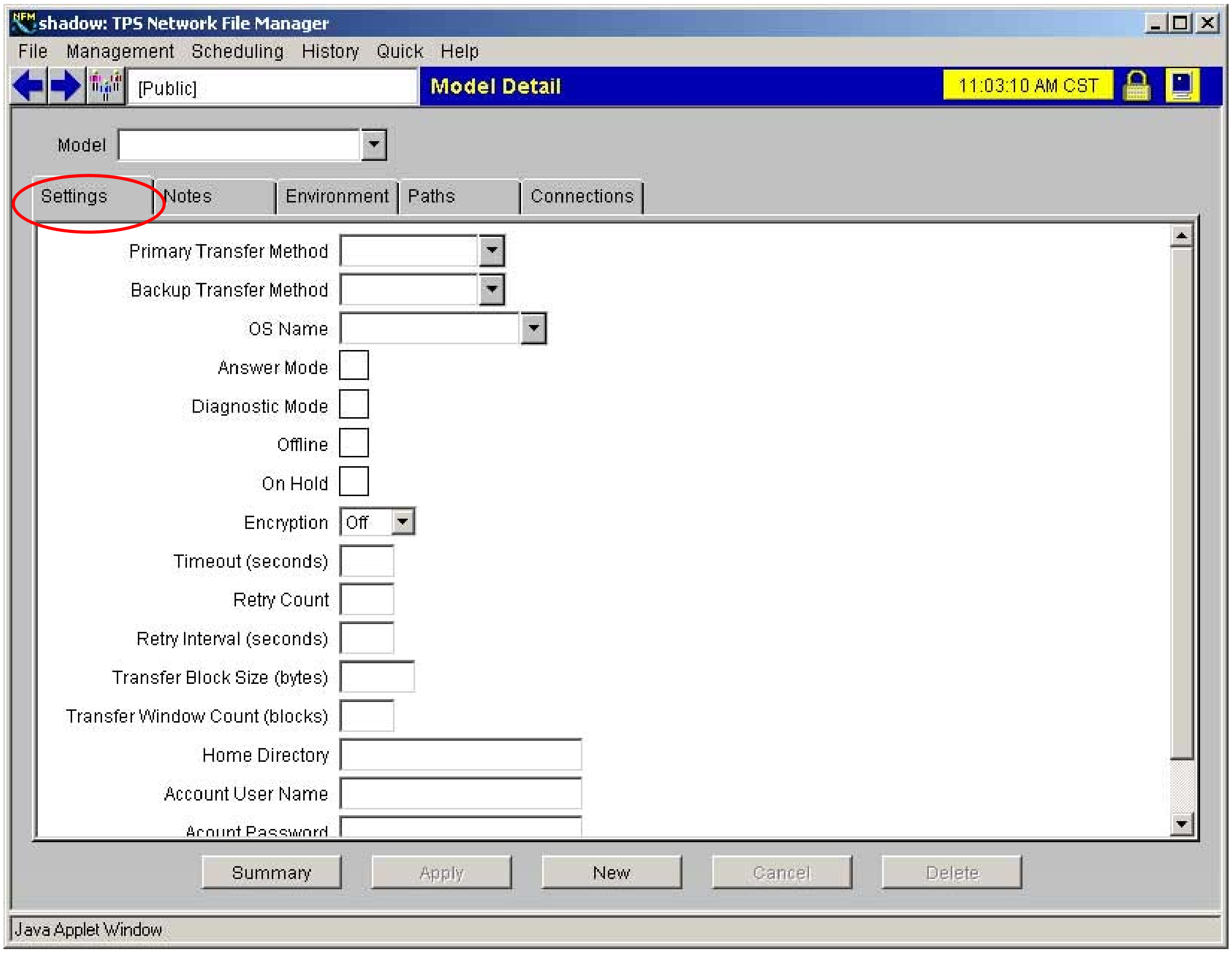
A Model is used to define parameters or options that are likely to be duplicated among several nodes. This gives the user the flexibility to make one change and affect several nodes without having to change each individual node, one at a time. Just like Nodes, a Model specifies options such as the Operating System, file transfer method, etc. The Model screen will look very similar to the Nodes screen.

Management | Model ▸ | Model Detail

(to access the Model screen)

Settings Tab

The Settings tab defines default characteristics for any node that will use this model as its default template (Each field may be overwritten in the Node screen).



Model

Primary Transfer Method

Backup Transfer Method

OS Name

Answer Mode □

Diagnostic Mode □

Offline □

On Hold □

Encryption Off

Timeout (seconds)

Retry Count

Retry Interval (seconds)

Transfer Block Size (bytes)

Transfer Window Count (blocks)

Home Directory

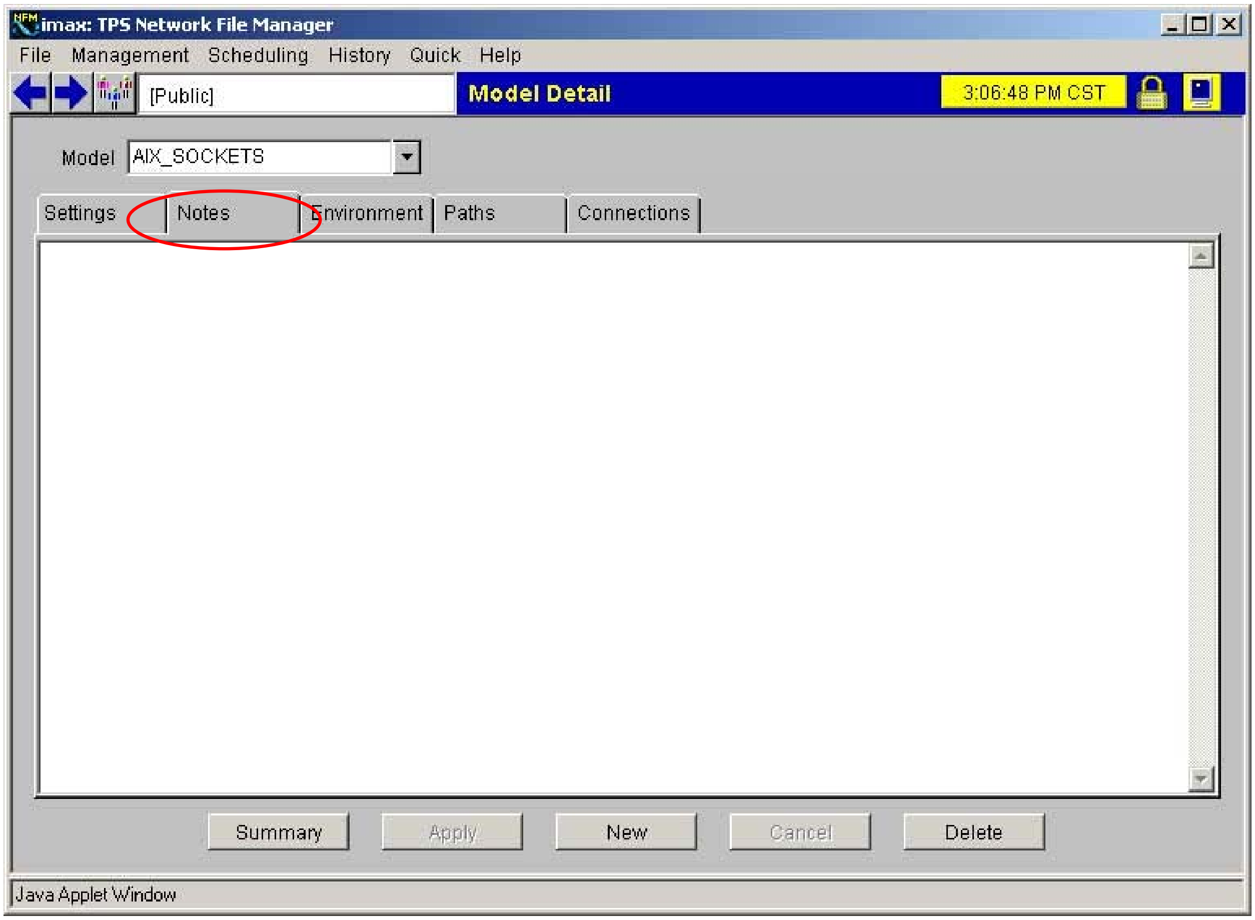
Account User Name

Acount Password

See the Node's Setting tab for field descriptions

Notes Tab

The Notes tab provides a text window for storing text specific to this model.

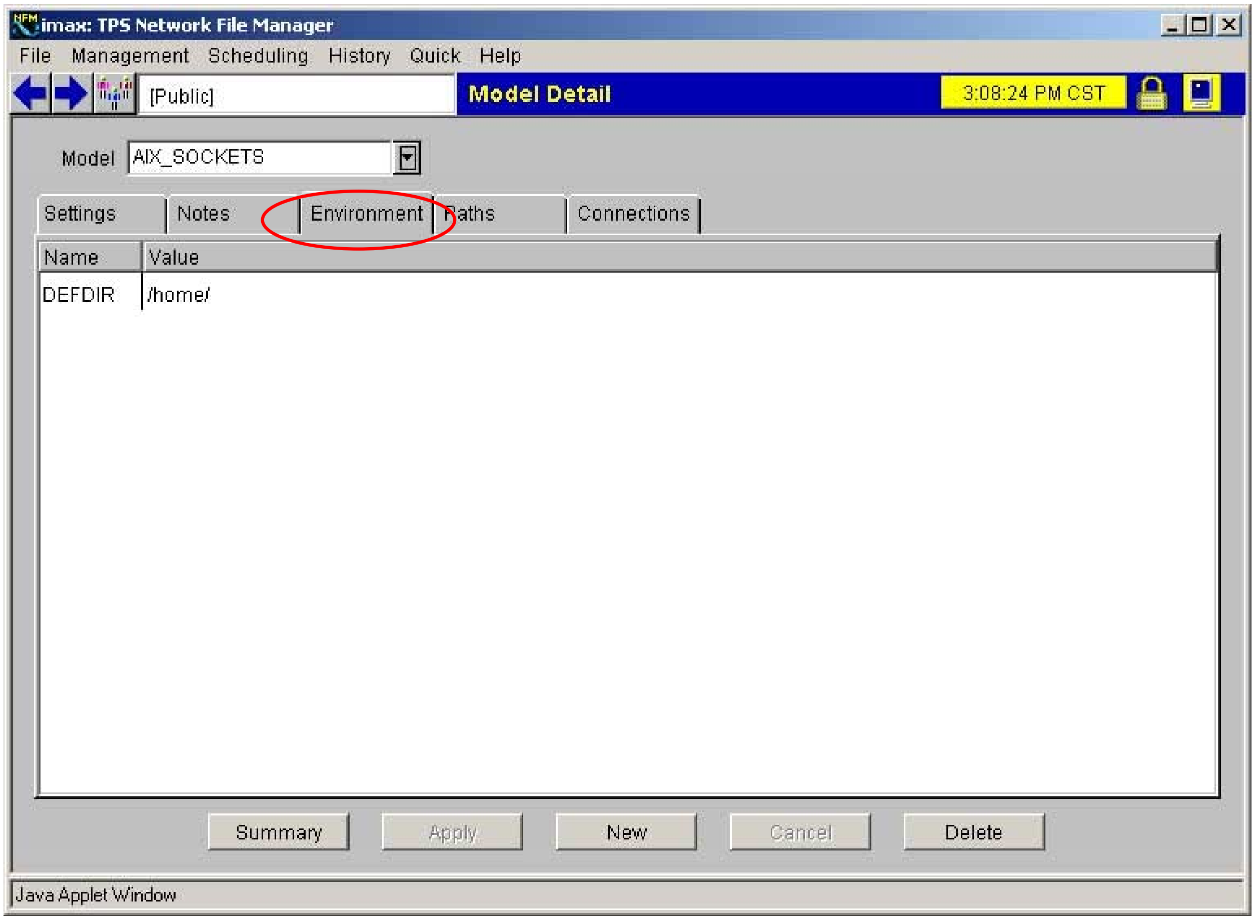


Model: AIX_SOCKETS

See Node’s Notes tab for field descriptions

Environment Tab

The Environment tab allows users to create custom environment, which will be inherited by any node that uses this model.


imax: TPS Network File Manager
File Management Scheduling History Quick Help
[Public]
Model Detail
3:08:24 PM CST
Settings Notes Environment Paths Connections
Summary Apply New Cancel Delete
Java Applet Window

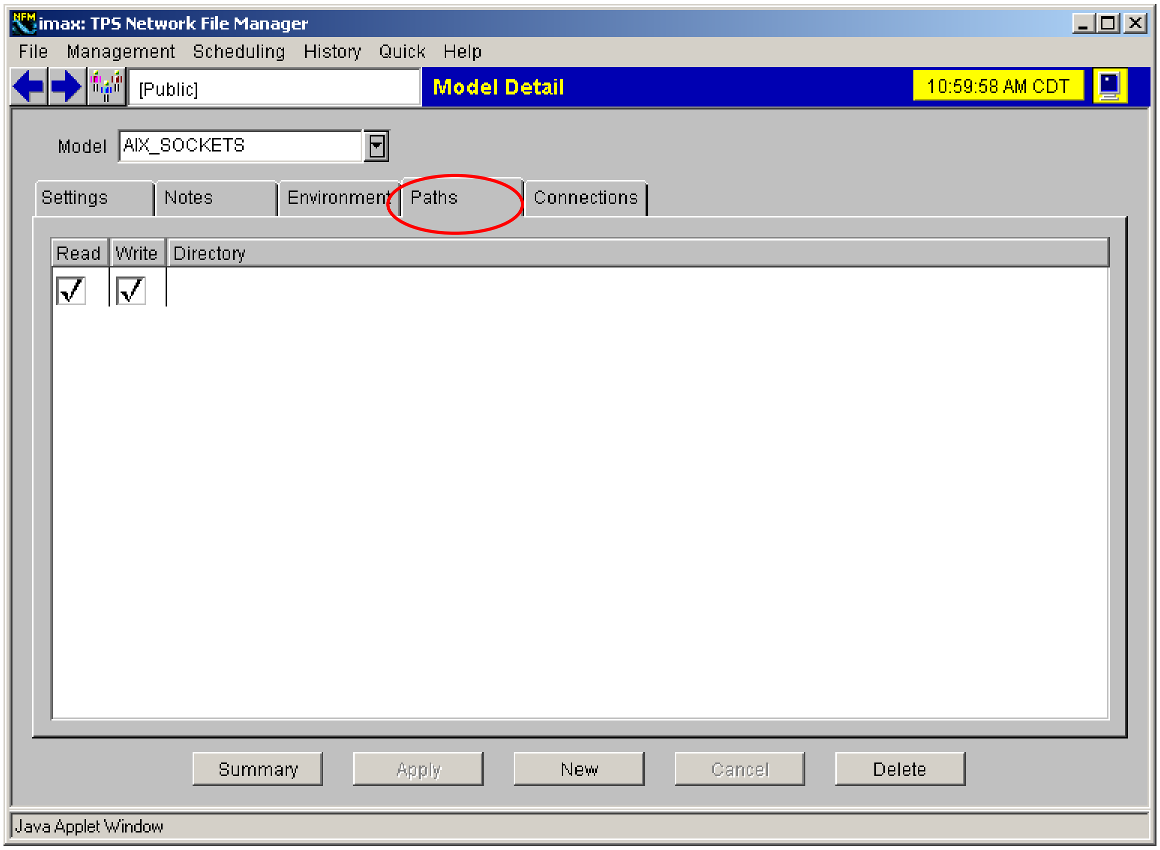
Model AIX_SOCKETS

Name	Value
DEFDIR	/home/

See Node’s Environment Tab for field descriptions

Paths Tab

The Paths tab maintains a list of path permissions associated with this model. Path permissions allow NFM to restrict or permit file transfer access to the specified directories on any nodes of this model type.



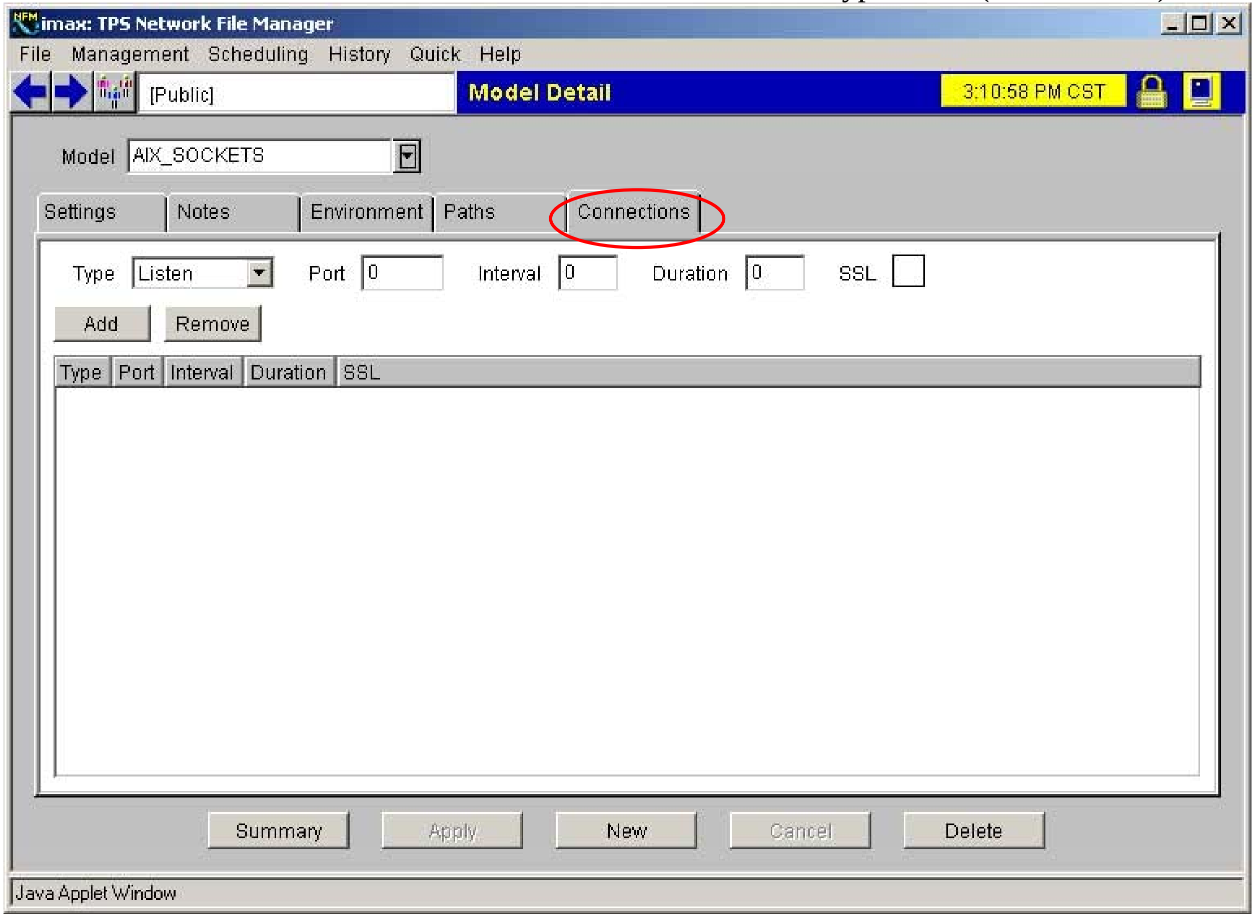
Model AIX_SOCKETS

Read	Write	Directory
☑	☑	

See Path Permissions under Security Components for more detail.

Connection Tab

The Connections tab allows the user to customize socket type nodes (NFM Clients).



Model: AIX_SOCKETS

Type: Listen
Port: 0
Interval: 0
Duration: 0
SSL: □

Type	Port	Interval	Duration	SSL

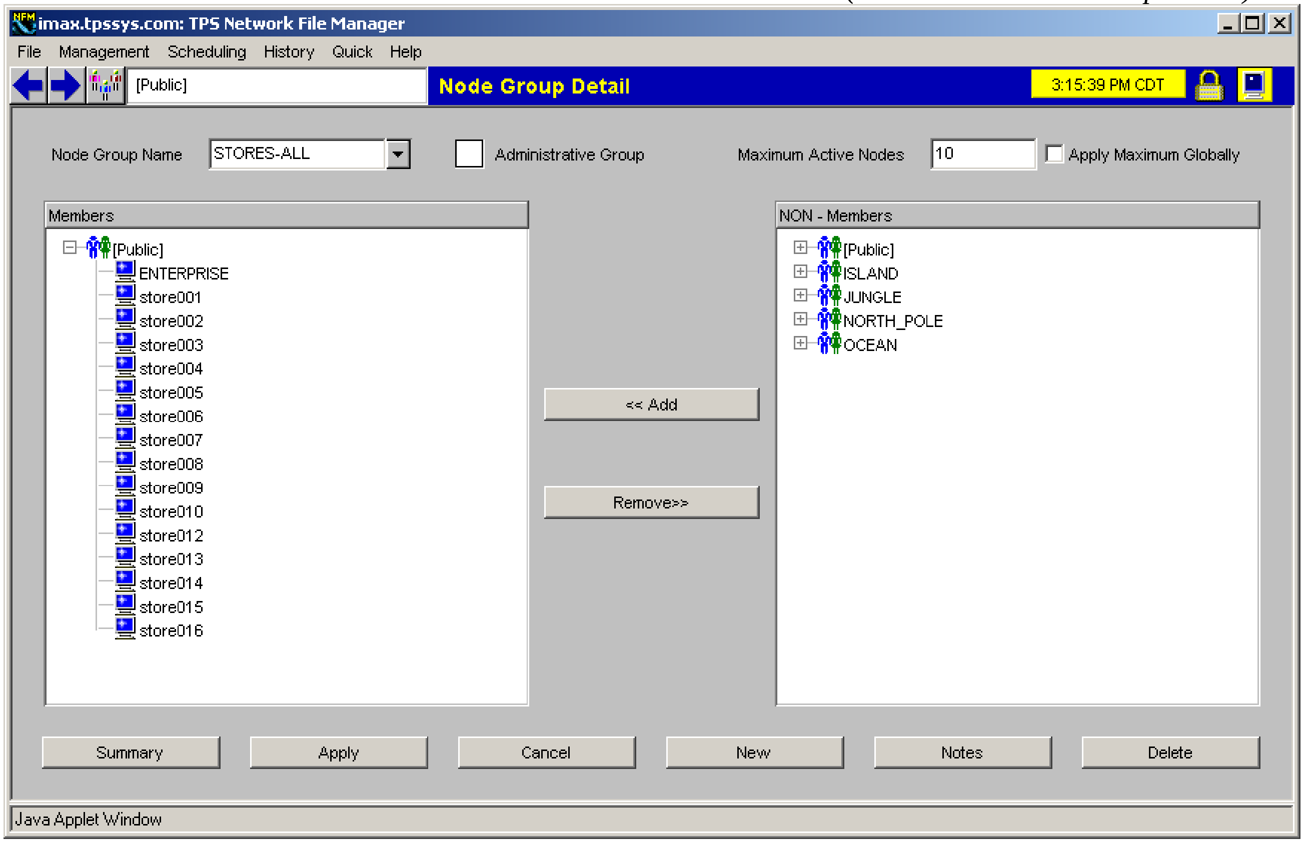
See Node's Connection tab for field descriptions

Node Group

A Node Group is a list of nodes used to allow actions to be performed on a group of nodes. A group many contain any combination of nodes and may contain other Node Groups. Nodes may belong to any number of groups. Nodes and Node Groups are also grouped together under Usergroups (What are Usergroups?).

Management | Node Group ▸ | Node Group Detail

(to access the Node Group screen)



imax.tpssys.com: TPS Network File Manager

Node Group Detail

[Public]

3:15:39 PM CDT

Node Group Name: STORES-ALL

□ Administrative Group

Maximum Active Nodes: 10

□ Apply Maximum Globally

Members
- [Public]
  - ENTERPRISE
  - store001
  - store002
  - store003
  - store004
  - store005
  - store006
  - store007
  - store008
  - store009
  - store010
  - store012
  - store013
  - store014
  - store015
  - store016

NON - Members
- [Public]
- ISLAND
- JUNGLE
- NORTH_POLE
- OCEAN

Java Applet Window

Field Name	Field Description
Administrative Group	This turns on administrative status for the node group so only an administrator or sub-administrator may add or remove node or node groups from the list. This should be turned on for groups that are used to configure individual users read write access to specific nodes in the User screen. This does not prohibit the use of this group with plan activity, it merely keeps a non-administrator from changing their own node access permissions.

Field Name	Field Description
Maximum Active Nodes	The maximum number of nodes from within this group that may be involved in plan activity at any given time. This is one of the bandwidth control features, which prevents NFM activity from over utilizing the network.
Apply Maximum Globally	If this box is checked, the "Maximum Active Nodes" setting (above) is in effect for all the nodes in this group, regardless of what other groups might have been used to initiate plan activity among these nodes. If this option is off, the limit will only take effect if this is the actual group used to specify activity.
Members	Lists all the nodes and node groups that are currently a member of the selected group (Groups and nodes are shown with different icons).
NON - Members	Lists all the nodes and node groups that are not currently a member of the selected group.

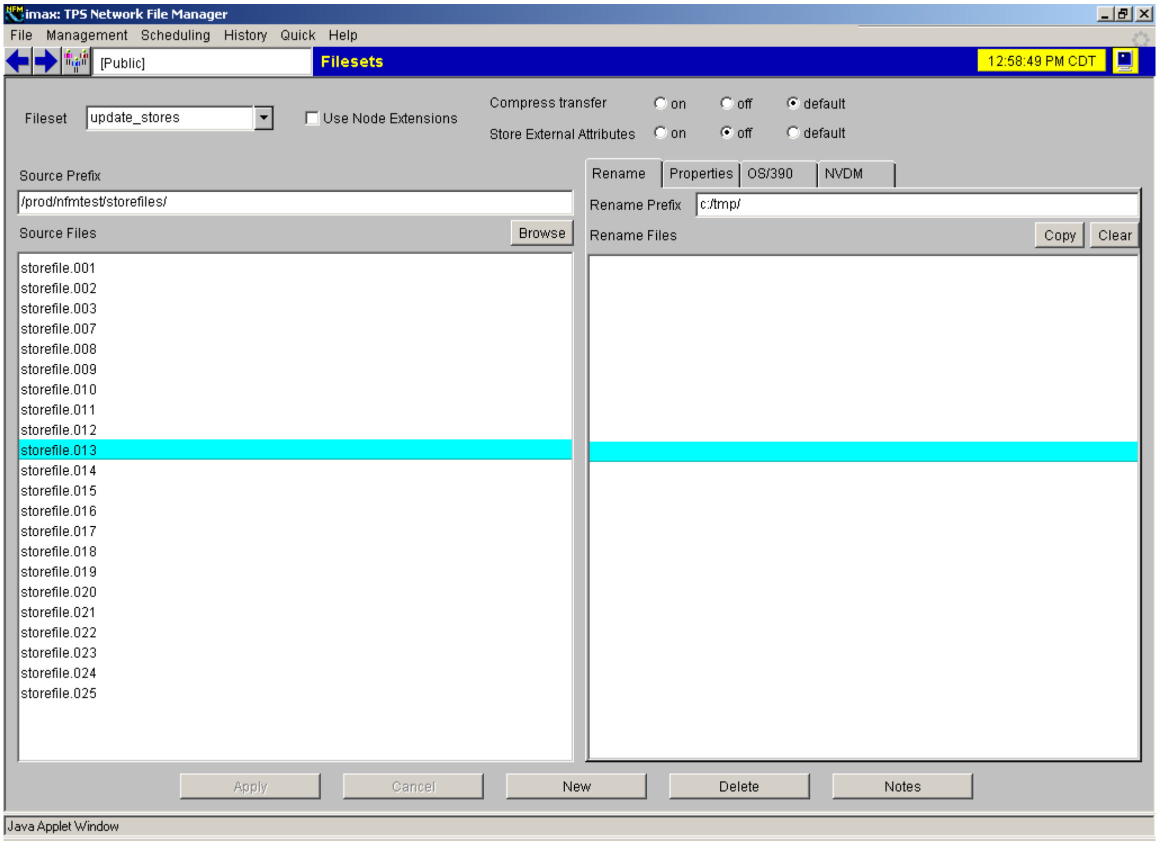
Icon	Icon Description
🖥	Node
🖧	Node Group
👥	Usergroup

Filesets

A Fileset contains a list of file(s). Filesets must be created for any file(s) that are to be transferred or otherwise manipulated by the NFM System. Filesets can contain wildcard expressions, such as * ?, which will be automatically cause all matching files to be processed during plan activity. Filesets can also contain environment variables; either NFM environment variables or user defined (What are environment variables?).

Management | Filesets

(to access the Filesets screen)



Fileset: update_stores

☐ Use Node Extensions

Compress transfer ○ on ○ off ● default

Store External Attributes ○ on ● off ○ default

Source Prefix: /prod/nfmtest/storefiles/

Rename Prefix: c:/tmp/

Source Files

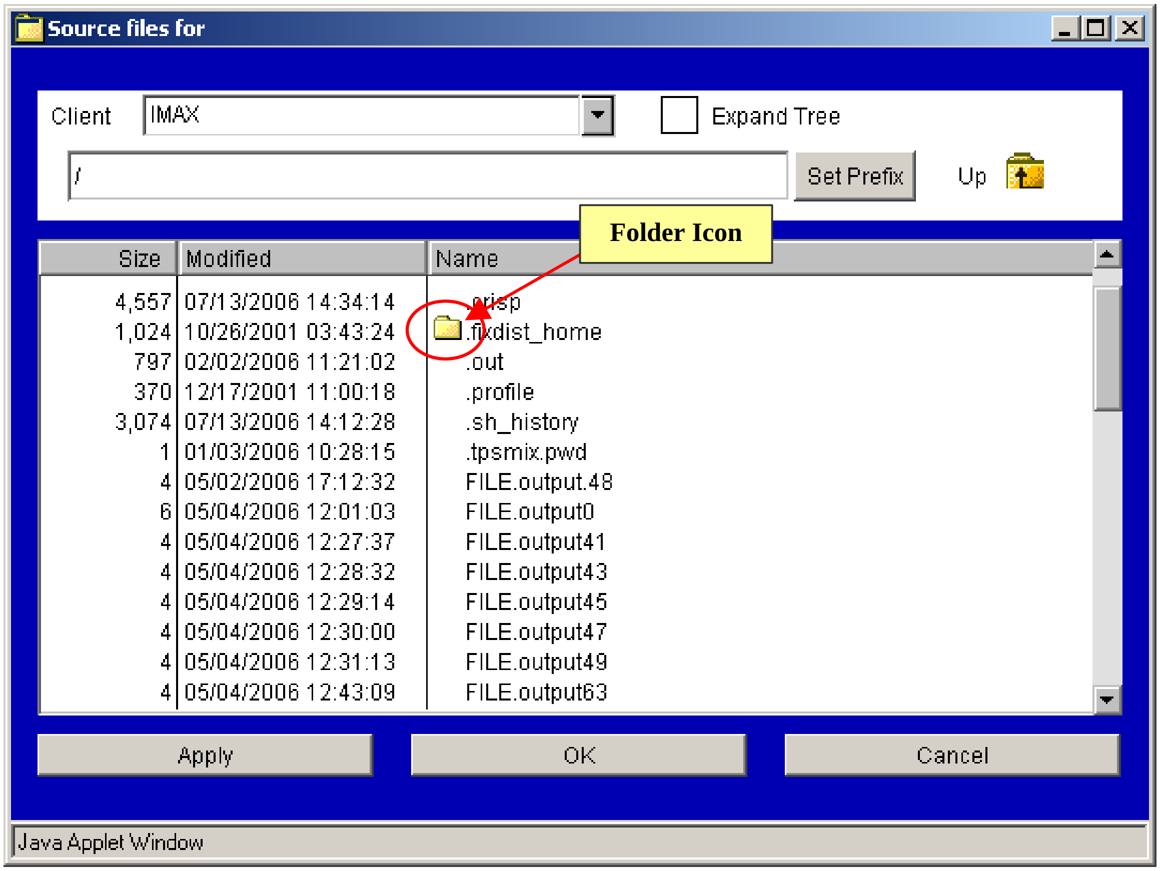
storefile.001
storefile.002
storefile.003
storefile.007
storefile.008
storefile.009
storefile.010
storefile.011
storefile.012
storefile.013
storefile.014
storefile.015
storefile.016
storefile.017
storefile.018
storefile.019
storefile.020
storefile.021
storefile.022
storefile.023
storefile.024
storefile.025

Rename Files

Field / Button Name	Field Description
Use Node Extensions	This mechanism is used for avoiding name collisions when transferring the same named files to and from a node group. Files transferring from a node group will be renamed on the target node filename.NODENAME. Transfers going to a node group will look for filename.NODENAME on the source node and strip off the extension during the transfer.
Compress transfer	All files defined in the fileset are compressed in transit (the actual sequence of events is: data is read from disk, compressed in memory using zlib algorithm, transmitted, received, uncompressed, written to disk). This can be especially beneficial for text files. If left on 'default' then compression will be attempted if the Compress Transfers option is checked on in System Settings. (** Compression can only occur between NFM Client type nodes **)
Store External Attributes	Enables NFM to retain attributes of a file that are operating system dependent even if that file is transferred to a different system type. If the file is transferred back to a system representing its original type, these attributes will still be intact. These attributes currently include the following items: UNIX: Owner name, Group name, and permissions. Windows: Attribute byte. File Distribution Attribute (FDA). If the default box is checked, the support is enabled based on the checkbox for External Attributes in the System Settings.
Source Prefix	Each file that makes up this Fileset record must have a full path specified to where it resides. This optional field allows any part of the path that is common to all the files to be specified here. The Source Prefix can be typed directly into this field or built from Browse and Set Prefix buttons. On Windows systems you can specify a UNC name to define a shared network drive (For example: \\computer_name\shared_folder) ** Source Prefix can contain environment variables **
Source Files	Specifies the names for individual files relative to the source prefix directory. If the Source Prefix is not specified this must be a full path to the file. If the Source Prefix is specified,

Field / Button Name	Field Description
	individual filenames may still specify a full path to a file that is different from the Source Prefix. Source Files can be typed directly into this field or built from the Browse button. On Windows systems you can specify a UNC name to define a shared network drive and file (For example: \\computer_name\shared_folder\filename) ** Source Files can contain wildcard expressions and environment variables **
Browse	If the files already exist on the server or client nodes, the browse button can be used to create the list from the actual directory.

Building a Fileset using the Browse button:



Source files for

Client: IMAX

□ Expand Tree

/

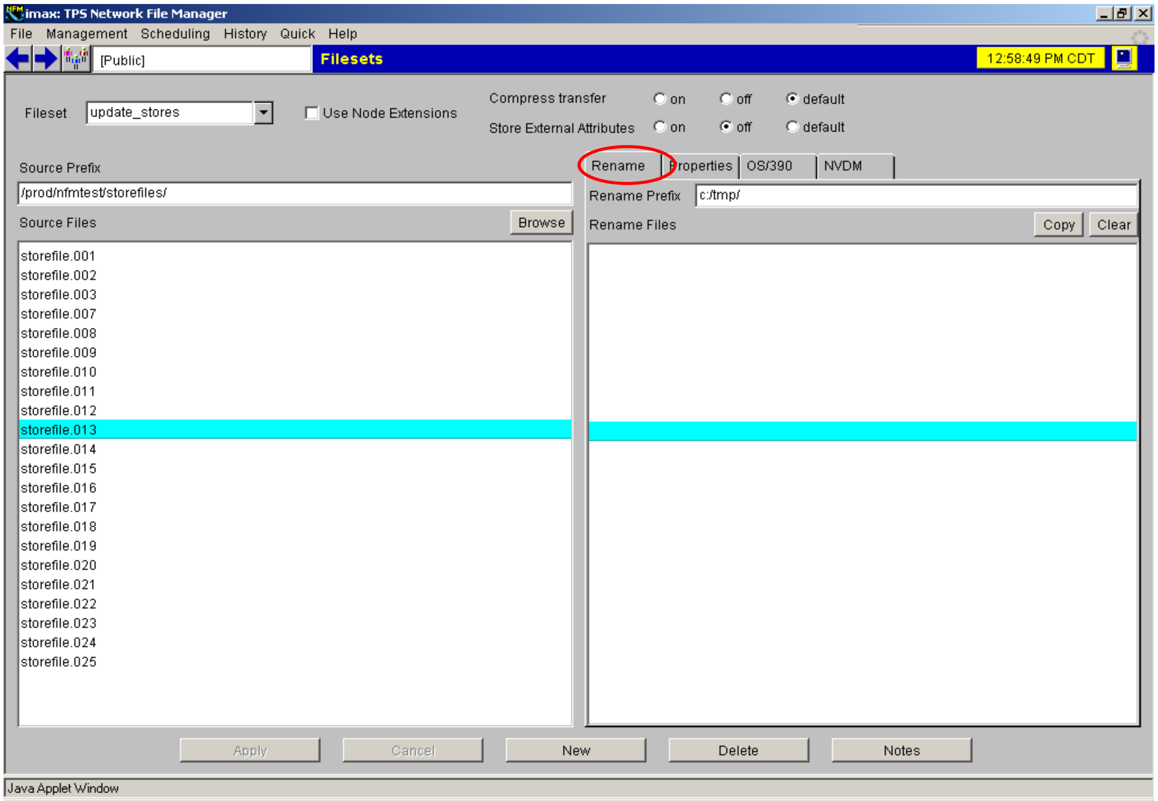
Size	Modified	Name
4,557	07/13/2006 14:34:14	.crisp
1,024	10/26/2001 03:43:24	.fixdist_home
797	02/02/2006 11:21:02	.out
370	12/17/2001 11:00:18	.profile
3,074	07/13/2006 14:12:28	.sh_history
1	01/03/2006 10:28:15	.tpsmix.pwd
4	05/02/2006 17:12:32	FILE.output.48
6	05/04/2006 12:01:03	FILE.output0
4	05/04/2006 12:27:37	FILE.output41
4	05/04/2006 12:28:32	FILE.output43
4	05/04/2006 12:29:14	FILE.output45
4	05/04/2006 12:30:00	FILE.output47
4	05/04/2006 12:31:13	FILE.output49
4	05/04/2006 12:43:09	FILE.output63

Field / Button Name	Field Description
Client	The computer or node name you want to browse.
Expand Tree	Expands all folders or directories in the current working directory.
Set Prefix	Takes the current directory and sets its value to the Fileset's Source Prefix field.
Up	Moves up one directory.

** A special note about the create directory functionality within NFM: NFM provides a way to create directories on the fly within a fileset. Under 'Source Files', provide a directory that exists on the source node (ex: C:\temp\) and under 'Rename Files', provide the directory you want to create (ex: C:\new_directory\).

Rename Tab

Rename tab on the right hand side, provides the ability to rename files on the target node(s).



Fileset update_stores

□ Use Node Extensions

Compress transfer ○ on ○ off ● default

Store External Attributes ○ on ● off ○ default

Source Prefix /prod/nfmtest/storefiles/

Rename Prefix c:/tmp/

Source Files

storefile.001
storefile.002
storefile.003
storefile.007
storefile.008
storefile.009
storefile.010
storefile.011
storefile.012
storefile.013
storefile.014
storefile.015
storefile.016
storefile.017
storefile.018
storefile.019
storefile.020
storefile.021
storefile.022
storefile.023
storefile.024
storefile.025

Rename Files

Field / Button Name	Field Description
Rename Prefix	Used to specify a target directory or portion of the target name common to all files. Rename Prefix combines with the individual target file name (either specified in ‘Rename Files’ or defaulted from the ‘Source Files’ fields) to form a complete path to each target file. On Windows systems you can specify a UNC name to define a shared network drive (For example: \\computer_name\shared_folder\) ** Rename Prefix can contain environment variables ** NOTE: The Rename folder has no meaning for file transfers to and from an OS/390 type node. In this case the Dataset or PDS name for the file is specified under the OS/390 folder along with other options described further in this chapter.
Rename Files	Specifies the names for individual files relative to the rename prefix directory. If the `Rename Prefix` is not specified this must be a full path to the file. If the Rename Prefix is specified, individual filenames may still specify a full path to a file that is different from the Rename Prefix. On Windows systems you can specify a UNC name to define a shared network drive and file (For example: \\computer_name\shared_folder\filename) ** Rename Files can contain wildcard expressions and environment variables **
Copy	Copies the highlighted line in the Source Files field to the Rename Files field.
Clear	Clears the Rename Prefix and Rename Files field.

More about Windows drive letters

The use of a drive letter when specifying a full path is not mandatory but it is recommended. A file name that begins simply with a forward slash will be interpreted to mean the current drive, which is not necessarily known to the user.

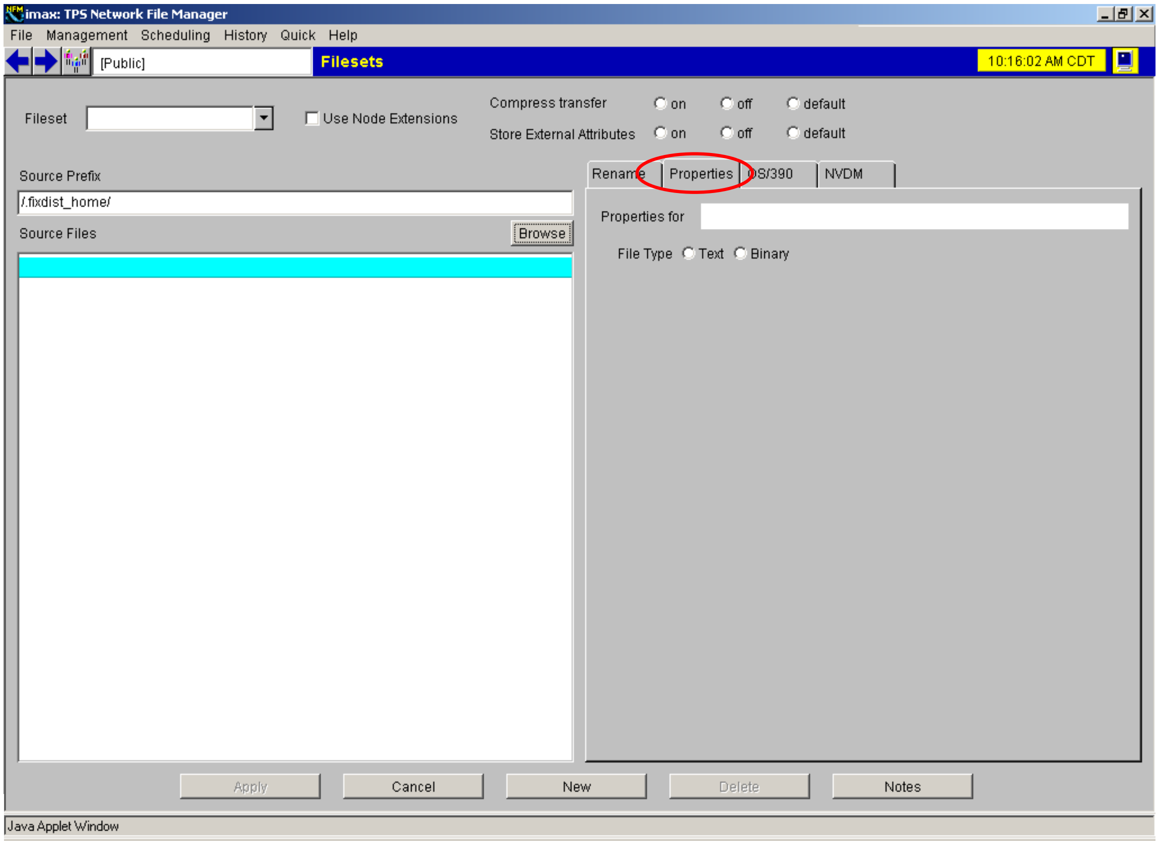
A mapped drive letter that references either local or network resources should never be used. This is because they are not global to the system, and are therefore not necessarily available to a services program (in this case the NFM client). The Universal Naming Convention (UNC) may be used instead to reference such a resource.

For more information on this please refer to the following document:

http://msdn2.microsoft.com/en-us/library/ms685143.aspx

Properties Tab

Properties tab contains additional information that may be configurable on a per file basis.



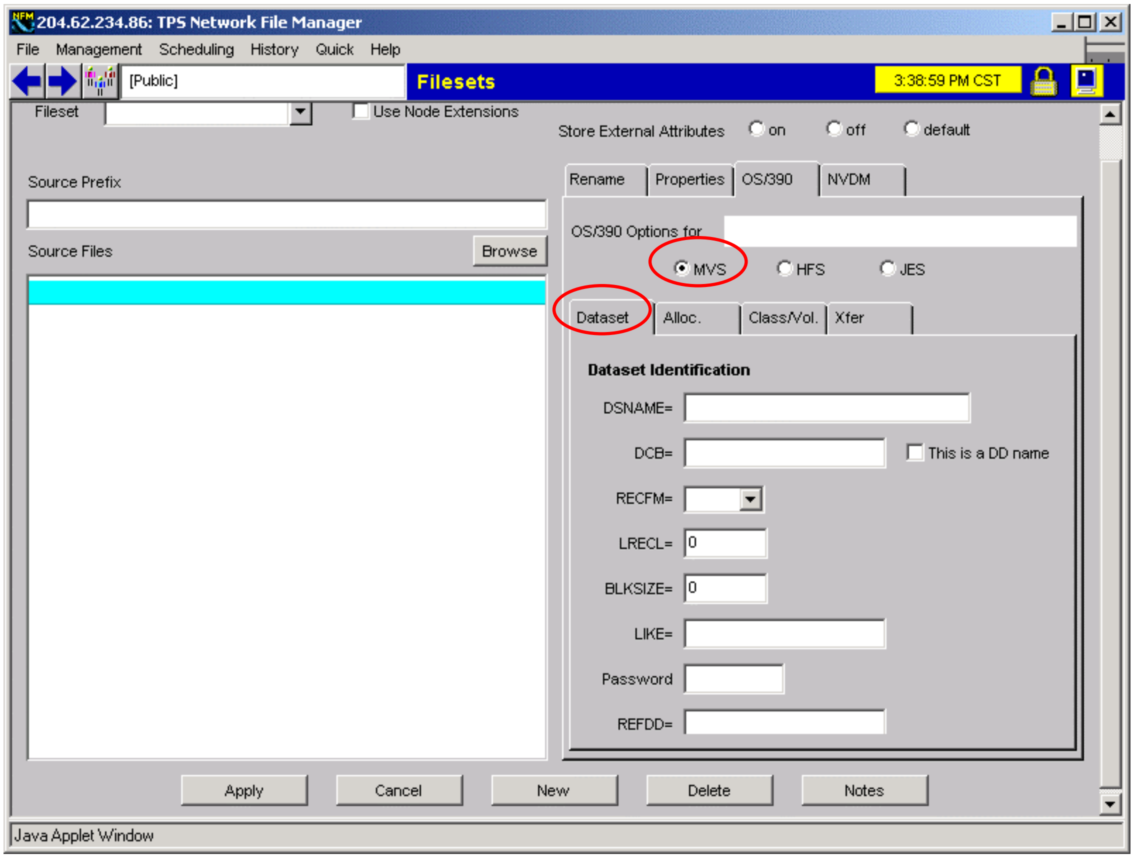
Source Prefix: /.fixdist_home/

Field Name	Field Description
Properties for	Contains the filename for the highlighted Source File on the left-hand side of the screen

Field Name	Field Description
File Type	Defines what type of file the file is Text will cause a translation to and from the native text type. This would be the following on the different platforms: • ASCII with new-line termination on most UNIX computers. • ASCII with new-line/carriage return termination on Windows and 4690 platforms. • Fixed or variable record length EBCDIC for OS/390 computers. Also used when determining transfer type for FTP type nodes.

OS/390 Tab | MVS Fields

Most of the fields described below are used exclusively for the creation of a dataset when the OS/390 or z/OS client system is the source or target of a file transfer.



□ Use Node Extensions

Store External Attributes ○ on ○ off ○ default

OS/390 Options for ● MVS ○ HFS ○ JES

DSNAME=

DCB= □ This is a DD name

RECFM=

LRECL= 0

BLKSIZE= 0

LIKE=

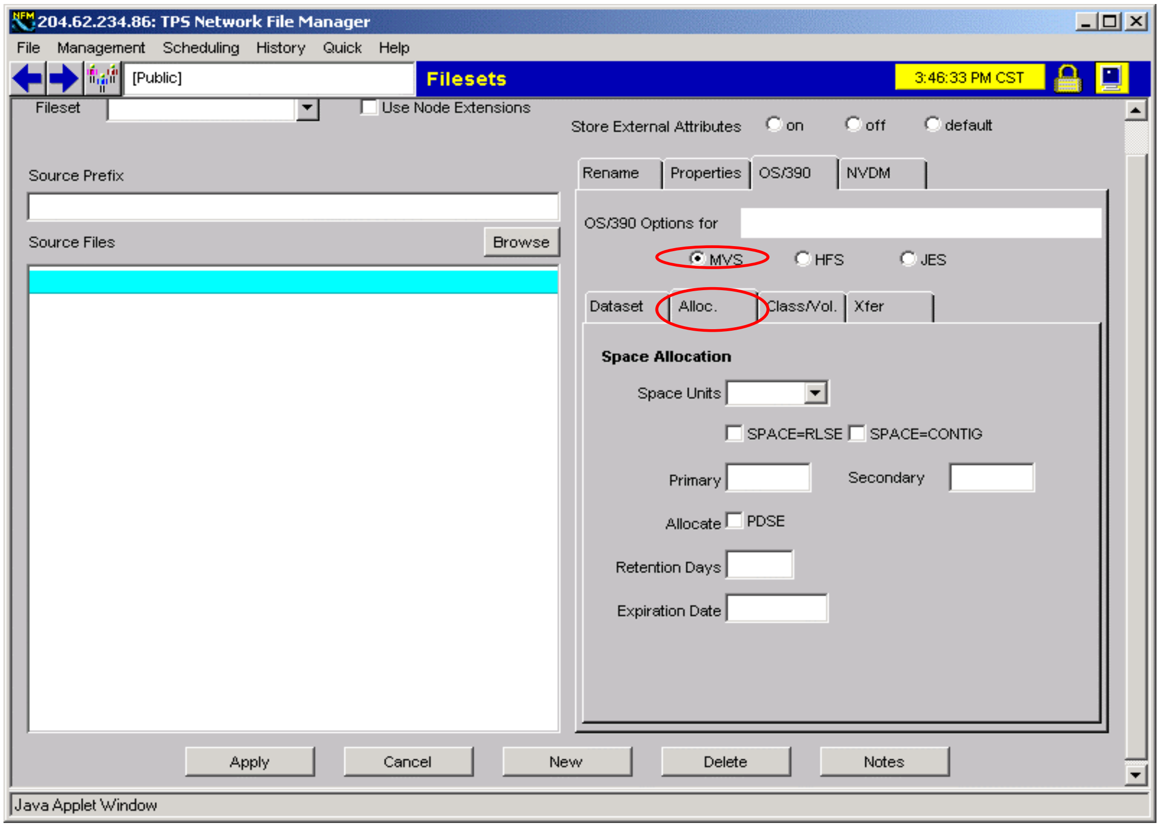
Password

REFDD=

** When files are transferred to and from a mainframe, the Source Prefix and Source Files fields always apply to the non-mainframe node, regardless of the direction of the transfer. **

Field name	Field Description
OS/390 Options	Filename selected under Source Files
DSNAME=	Host dataset name for the file. This should be in the standard form "`a.b.c.d`" for a dataset and "`a.b.c.d(member)`" for a partition dataset member.
DCB=	Use the DCB named here to provide default attributes when creating a dataset.
This is a DD name	Indicates that the name provided in the previous field (DCB), is actually a DD name that provides default attributes for target dataset creation.
RECFM=	Record format options: (blank) Use the default type. **`U`** Undefined. **`F`** Fixed length, unblocked. **`FB`** Fixed-length, blocked. **`FS`** Fixed-length, unblocked, standard. **`FBS`** Fixed-length, blocked, standard. **`FA`** Fixed-length, ASA print-control characters. **`FBA`** Fixed-length, blocked, ASA print-control characters. **`FSA`** Fixed-length, unblocked, standard, ASA print-control characters. **`FBSA`** Fixed-length, blocked, standard, ASA print-control characters. **`V`** Variable-length, unblocked. **`VB`** Variable-length, blocked. **`VS`** Variable-length, unblocked, spanned. **`VBS`** Variable-length, blocked, spanned. **`VA`** Variable-length, ASA print control characters, unblocked. **`VBA`** Variable-length, blocked, ASA print control characters. **`VSA`** Variable-length, spanned, ASA print control characters. **`VBSA`** Variable-length, blocked, spanned, ASA print control characters.

Field name	Field Description
LRECL=	Logical record length
BLKSIZE=	Block size
LIKE=	Use the cataloged data set named here to provide default attributes when creating a dataset.
Password	Password for a password-protected dataset
REFDD=	Use the specified ddname to provide default attributes when creating a dataset.



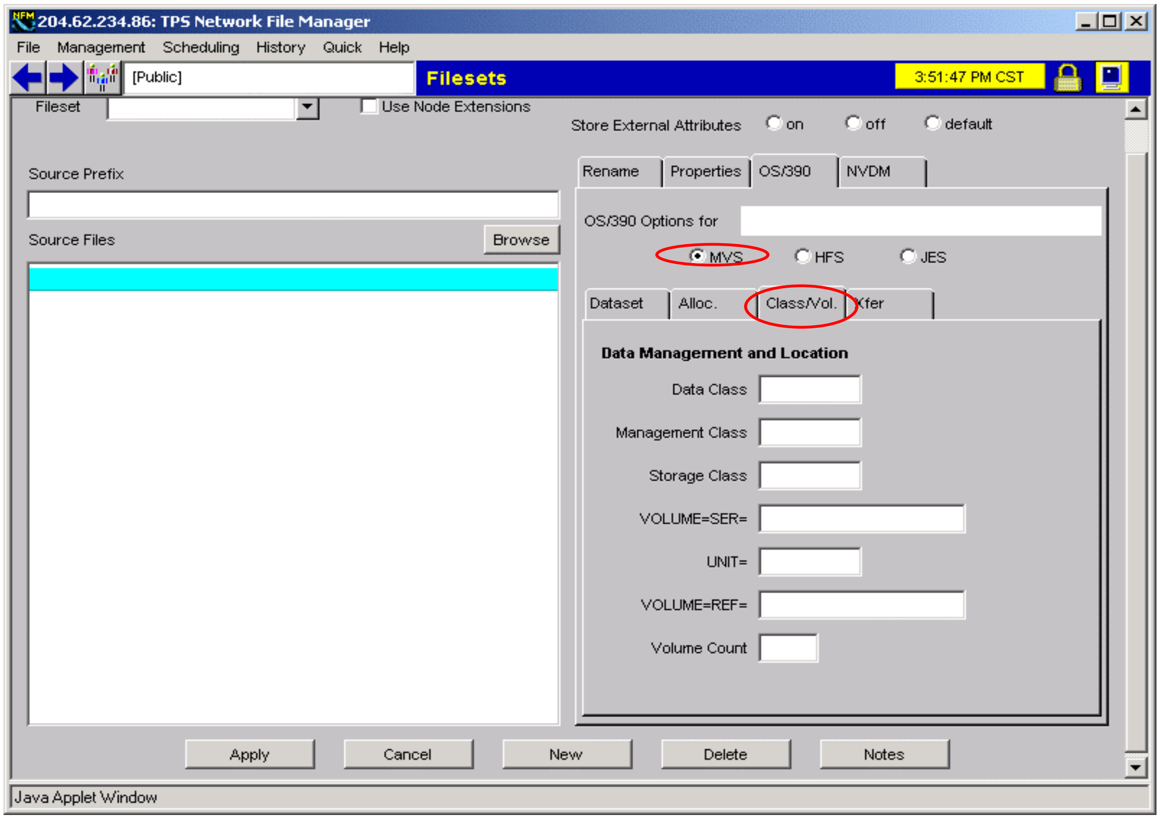
[Public]

3:46:33 PM CST

Store External Attributes ○ on ○ off ○ default

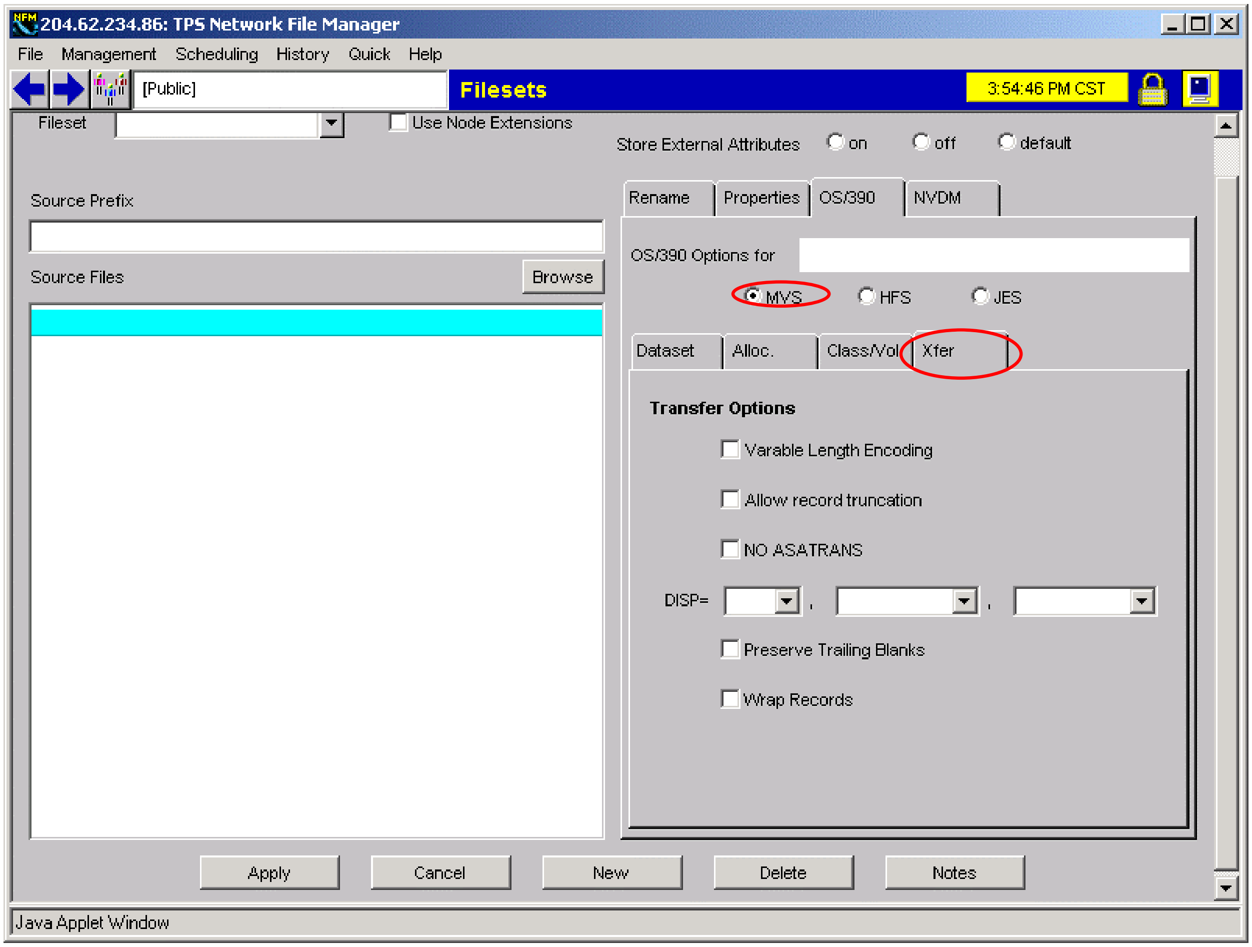
OS/390 Options for ● MVS ○ HFS ○ JES

Field name	Field Description
Space Units	Space allocation unit format.
SPACE=RLSE	Causes unused space to be released after a dataset is created and the necessary space to hold the transferred file is allocated and used.
SPACE=CONTIG	Indicates that the space requested for a target dataset be allocated contiguously.
Primary	Primary space allocation unit.
Secondary	Secondary space allocation unit.
Allocate PDSE	Allocate a PDSE (extended) rather than a PDS.
Retension Days	Number of days before data set expires.
Expiration Date	The date the data set expires.



OS/390 Options for ● MVS ○ HFS ○ JES

Field name	Field Description
Data Class	Data class for the new data sets.
Management Class	Management class for new data sets.
Storage Class	Storage class for new data sets.
VOLUME=SER=	Volume serial number.
UNIT=	Device type if applicable.
VOLUME=REF	Volume ID for dataset.
Volume Count	Number of volumes this data set may span.



204.62.234.86: TPS Network File Manager

[Public]

Filesets

3:54:46 PM CST

Fileset

□ Use Node Extensions

Store External Attributes ○ on ○ off ○ default

Source Prefix

Source Files

OS/390 Options for

● MVS ○ HFS ○ JES

Transfer Options

□ Varable Length Encoding

□ Allow record truncation

□ NO ASATRANS

DISP= , ,

□ Preserve Trailing Blanks

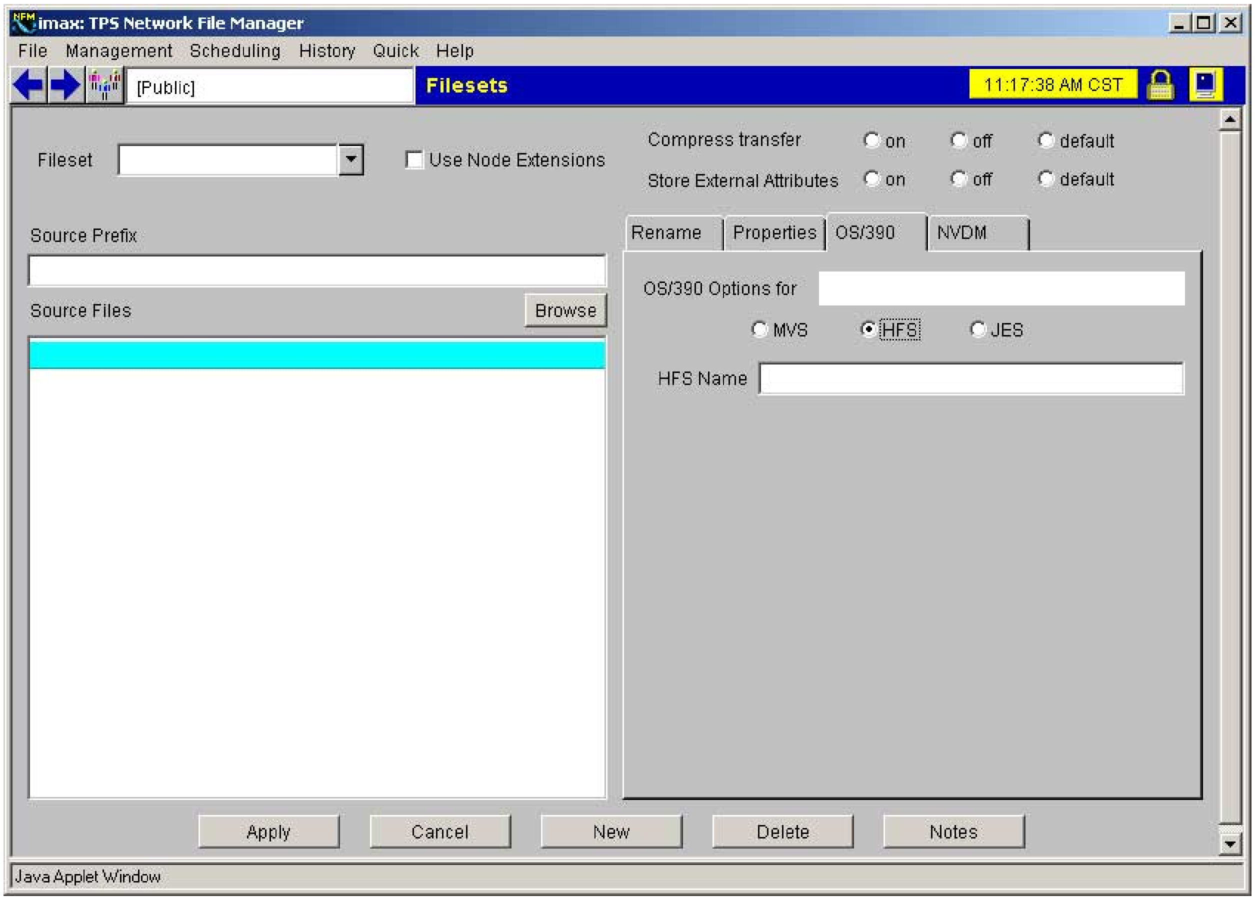
□ Wrap Records

Field name	Field Description
Variable Length Encoding	Used to enable support for variable length encoding within files, in order to preserve variable record length information in binary files. This option will only affect files that have a File Type of Binary and include a Record Format of Variable-length. Such files have no intrinsic method of denoting the beginning and end of each record when being transferred from a non-OS/390 flat file to or from an OS/390 variable record length file. For transfers to the NFM/OS390 client, this option states that the non-OS/390 source file contains a specific data format that embeds the variable length record information. Each file begins with a 2 byte length field that contains the length of the first variable length record, followed by the contents of that record. This is followed by another 2 byte length field for record two, and so on. These embedded VLE descriptors can be provided directly by a customer application, and is also provided as an option of the nfmparse program in preparing files for transfer. If the VLE option is turned off, the OS/390 target file will have variable record boundaries that are meaningless and undefined. For transfer from the NFM/OS390 client, this option states that the non-OS/390 target file will be created with VLE descriptors embedded in the data of the target file. If the VLE option is turned off, the non-OS/390 target file will only contain the binary contents of the original file with all record boundaries stripped off. NOTE: The act of using the VLE option to transfer a binary variable length file from OS/390 to a non-OS/390 platform (followed by transferring the file back to another OS/390 platform) will perfectly preserve the original OS/390 file layout while recreating all the proper variable length record boundaries.
Allow record truncation	Used to modify the handling of text data being written to an OS/390 file that uses a fixed record format. For NFM/OS390 text data transfers, the “newline” characters in the non-OS/390 source file define each record boundary. This makes these files intrinsically variable length in nature. When writing to a fixed length text file, text lines that are shorter than the fixed record size will be padded with spaces on the end of the record until it reaches the fixed record size. By default, if a text line is

Field name	Field Description
	longer than the fixed record size, a file transfer error occurs; aborting the transfer. When this option is specified, this condition is treated as a warning that appears in the audit record, and the transfer is allowed to continue.
No ASATRANS	Specifies that ASA control characters should not be translated to print control characters during file transfer.
DISP=	Specifies the status of a data set.
Preserve Trailing Blanks	Do not automatically truncate trailing blanks when copying a text file from OS/390
Wrap Records	Automatically wrap data that exceeds the record size into the next record when copying a text file to OS/390. By default, data that exceeds the record size is truncated.

OS/390 Tab | HFS Fields

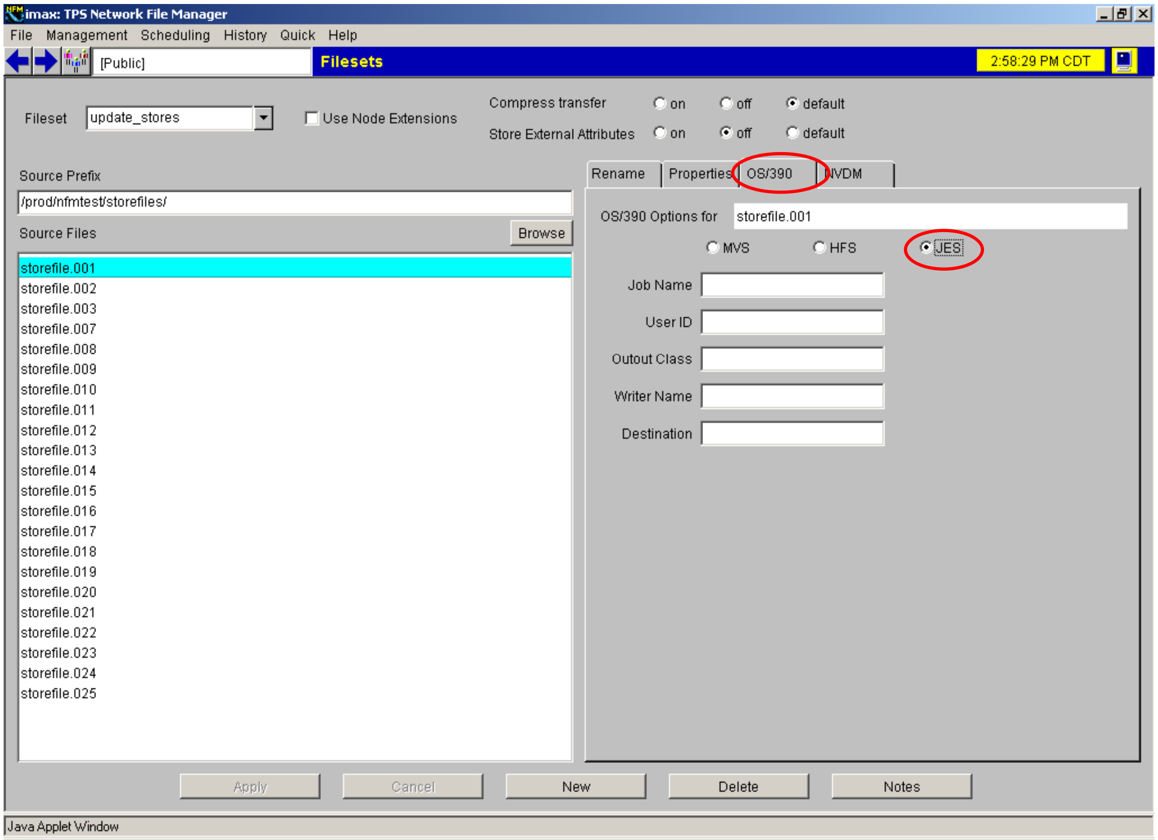
This panel contains attributes for files copied to and from a HFS (hierarchical file system) on the OS/390 platform.



Field Name	Field Description
HFS Name	The full HFS path and file name

OS/390 Tab | JES Fields

This panel contains attributes used to select data for transfer directly from the JES output queue to standard files on a non-OS/390 system. Each file in the output queue that matches all of the fields that are specified here will be selected.



Fileset update_stores

□ Use Node Extensions

Compress transfer ○ on ○ off ● default

Store External Attributes ○ on ● off ○ default

Source Prefix /prod/nfmtest/storefiles/

Source Files
storefile.001
storefile.002
storefile.003
storefile.007
storefile.008
storefile.009
storefile.010
storefile.011
storefile.012
storefile.013
storefile.014
storefile.015
storefile.016
storefile.017
storefile.018
storefile.019
storefile.020
storefile.021
storefile.022
storefile.023
storefile.024
storefile.025

OS/390 Options for storefile.001

○ MVS ○ HFS ● JES

Job Name

User ID

Outout Class

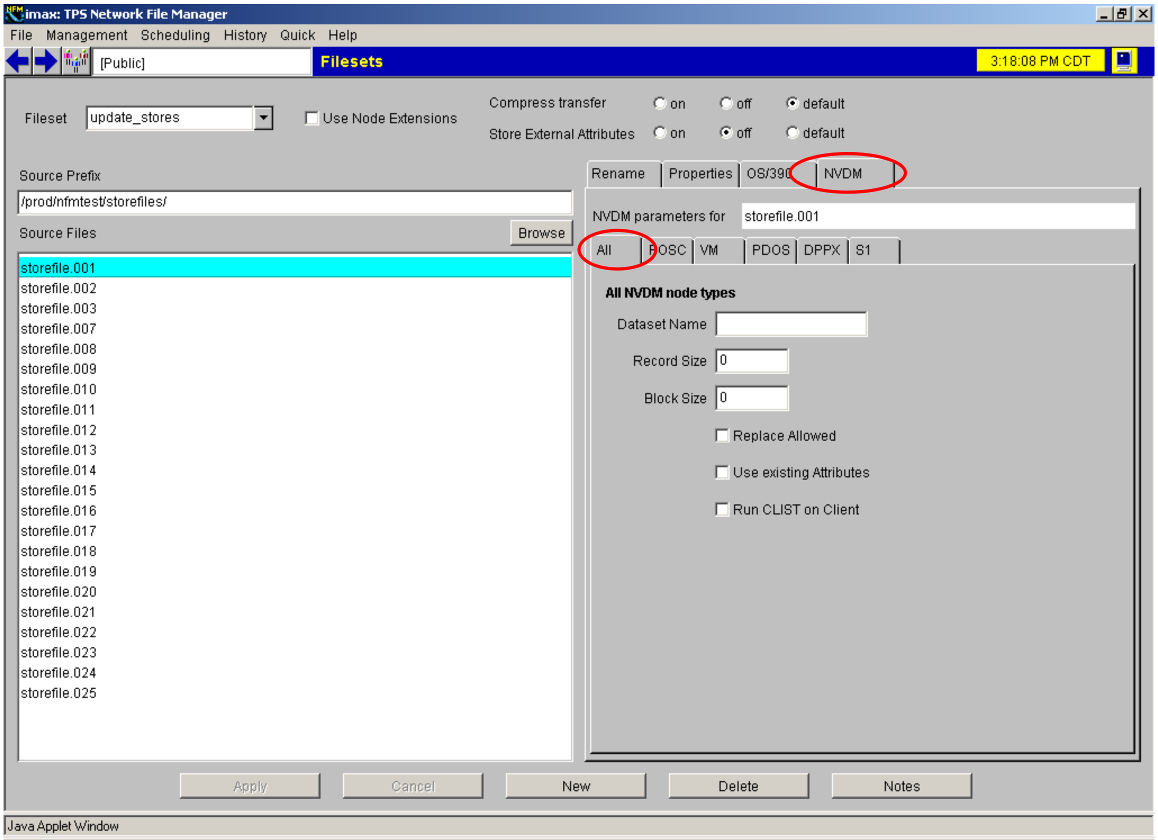
Writer Name

Destination

Field Name	Field Description
Job Name	Job name
User ID	User ID
Output Class	<system specific>
Writer Name	<system specific>
Destination	<system specific>

NVDM Tab

NVDM tab contains per file definitions for use with NVDM type nodes.



Fileset update_stores ☐ Use Node Extensions

Compress transfer ○ on ○ off ● default

Store External Attributes ○ on ● off ○ default

Source Prefix

/prod/nfmtest/storefiles/

Source Files

storefile.001
storefile.002
storefile.003
storefile.007
storefile.008
storefile.009
storefile.010
storefile.011
storefile.012
storefile.013
storefile.014
storefile.015
storefile.016
storefile.017
storefile.018
storefile.019
storefile.020
storefile.021
storefile.022
storefile.023
storefile.024
storefile.025

NVDM parameters for storefile.001

All NVDM node types

Dataset Name

Record Size 0

Block Size 0

☐ Replace Allowed

☐ Use existing Attributes

☐ Run CLIST on Client

NVDM, with the appropriate SNA Primary software running on the NFM Server, files may be transferred to/from NVDM catcher enabled nodes. Requirements and parameters for these files need their own configuration so the NVDM box allows defining these characteristics on a per file basis. (See Appendix C: Setting up NVDM node

Plans

Plans detail a set of tasks to be performed involving NFM nodes. This may include file transfers, program execution, and various other functions as well. Plans enable the user to specify conditional logic, which allows parts of the plan to be run or not run based on the success or failure of other parts. A plan is comprised of the following components, a header, one or more phases, one or more functions:

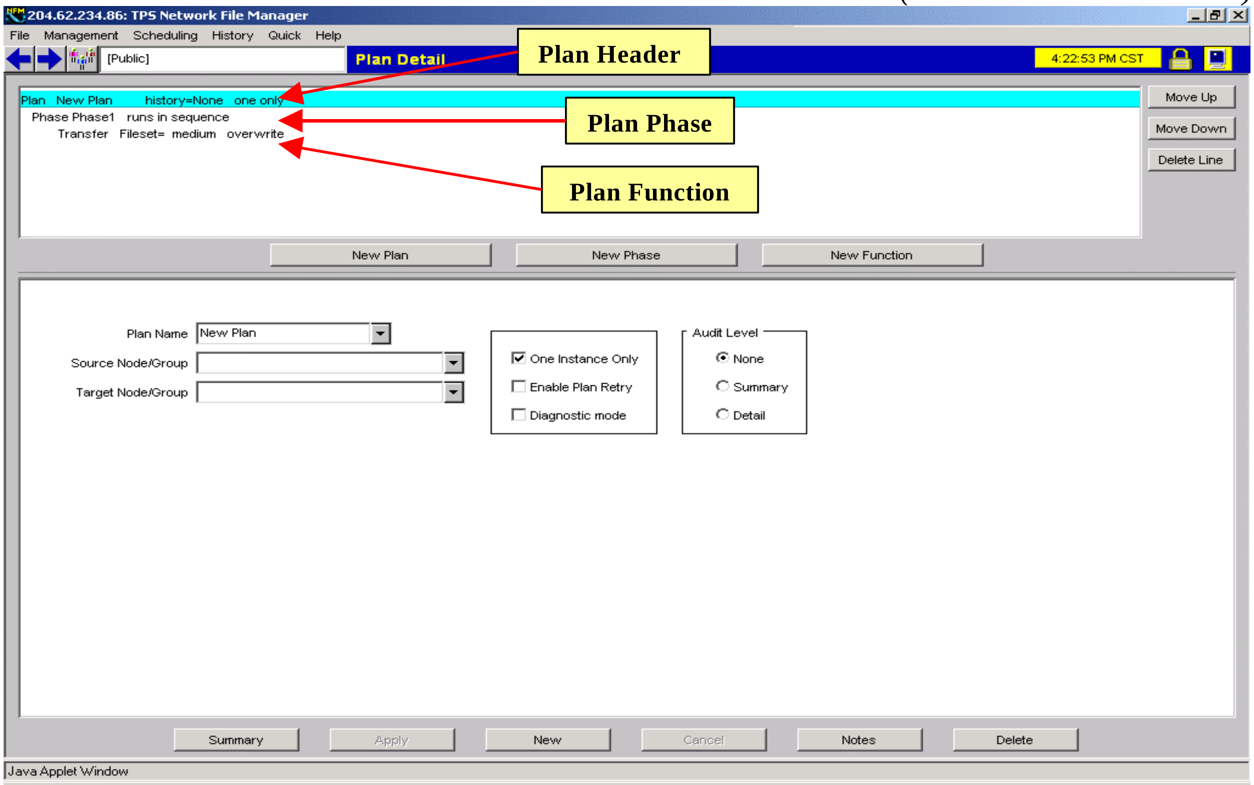
Plan Header The header contains the plan name, the default source and target node / node groups and a flag specifying the amount of history to record.

Plan Phase A phase will determine when and if a group of functions will run. Phases may be set to run consecutively (by default) or concurrently. Phases may also be set to run conditionally upon the previous phase or collective phases return codes(s).

Plan Function Functions specify individual steps to perform in a plan.

Management | Plan ▸ | Plan Detail

(to access the Filesets screen)



```
Plan   New Plan        history=None   one only
   Phase Phase1    runs in sequence
       Transfer   Fileset=  medium   overwrite
```

Plan Name: New Plan
Source Node/Group:
Target Node/Group:

☑ One Instance Only
□ Enable Plan Retry
□ Diagnostic mode

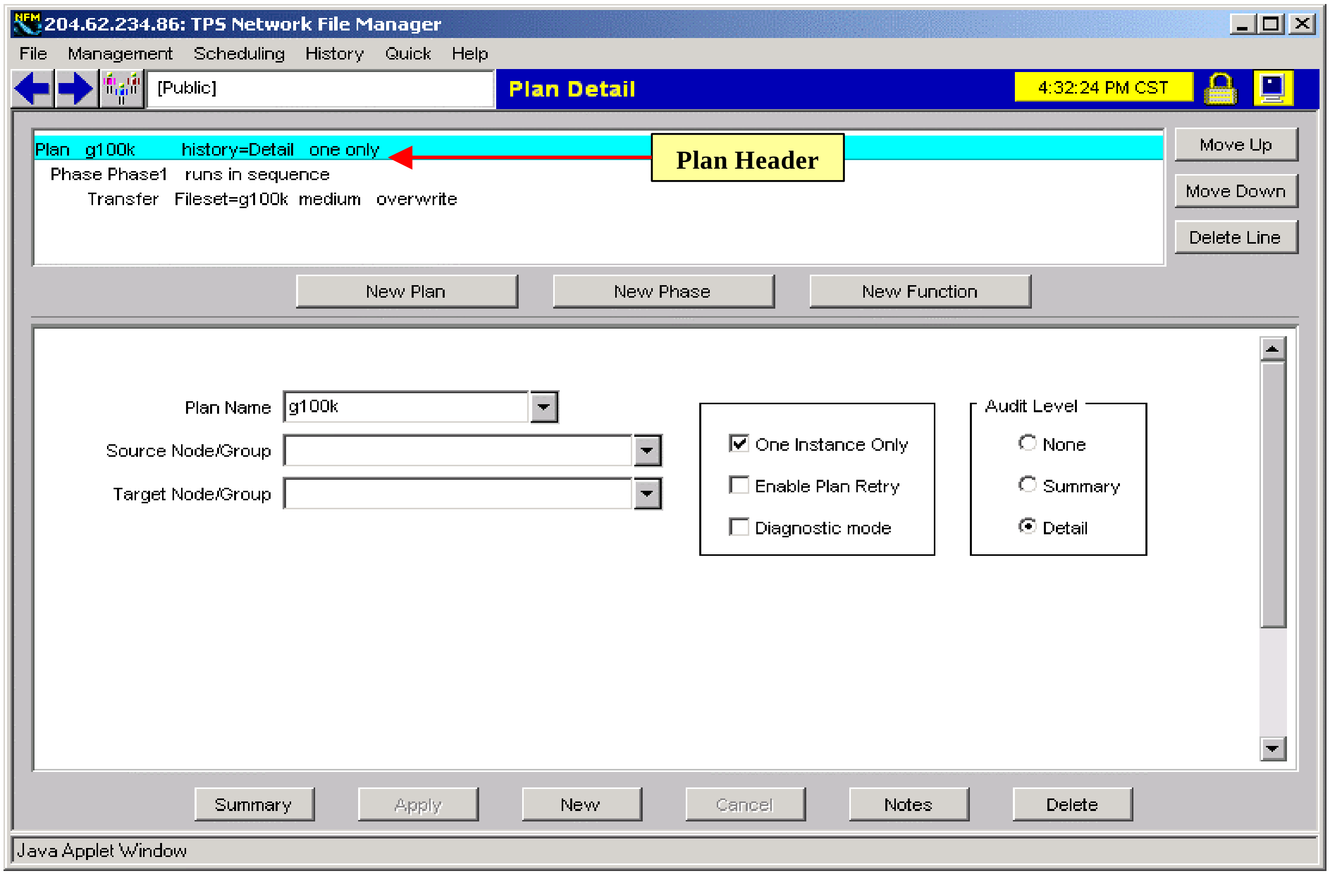
Audit Level
● None
○ Summary
○ Detail

(This example contains 1 Plan Header, 2 Plan Phases, and 2 Plan Functions)

Plan Components

Plan Header

The header contains the plan name, the default source and target node / node groups and a flag specifying the amount of history to record. A plan can only contain one Plan Header.



204.62.234.86: TPS Network File Manager

[Public]

Plan Detail

4:32:24 PM CST

```
Plan  g100k      history=Detail  one only
  Phase Phase1   runs in sequence
      Transfer   Fileset=g100k  medium   overwrite
```

Plan Header

Plan Name: g100k

Source Node/Group:

Target Node/Group:

☑ One Instance Only

☐ Enable Plan Retry

☐ Diagnostic mode

Audit Level

○ None

○ Summary

● Detail

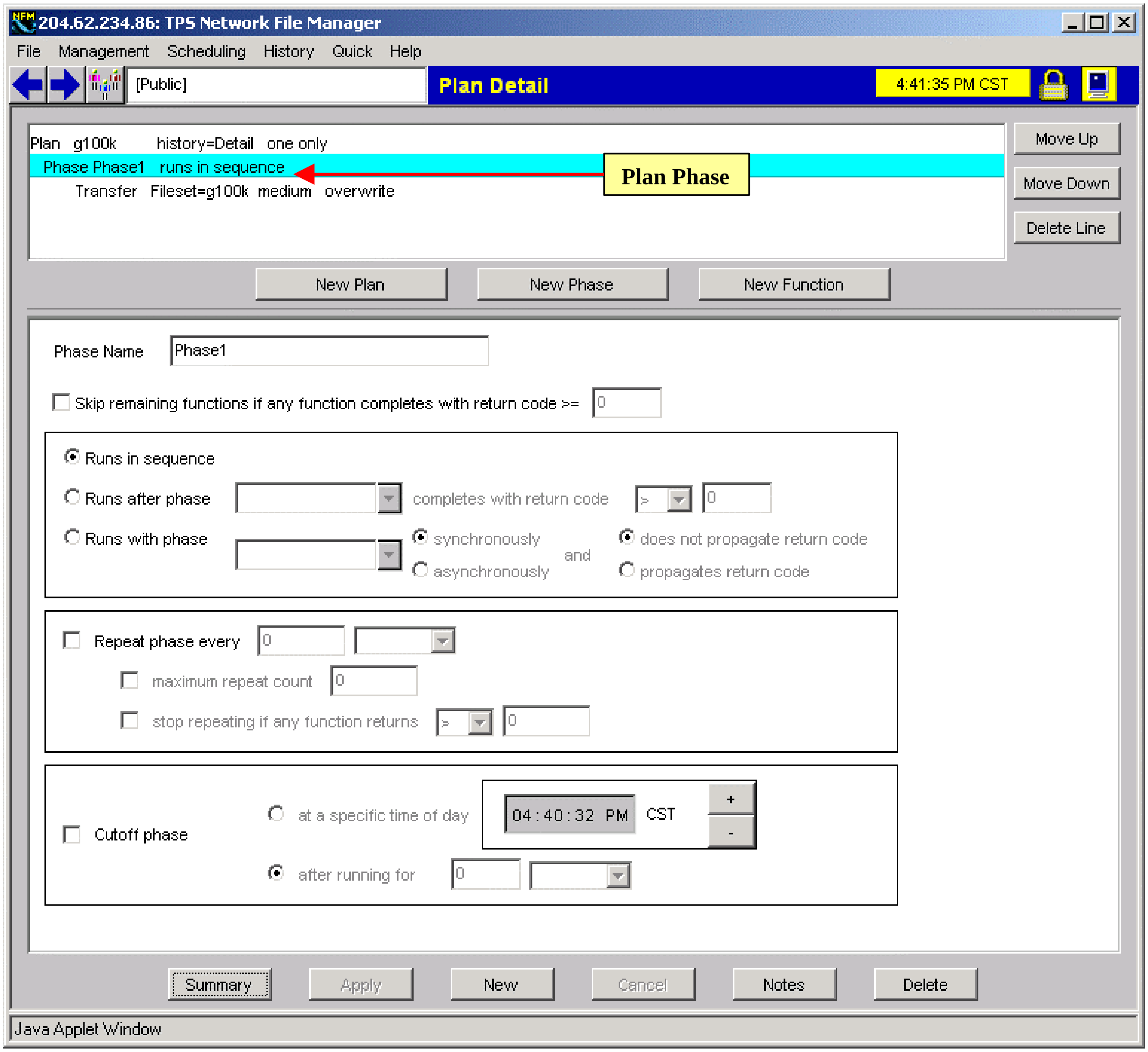
Java Applet Window

Field Name	Field Description / Selection Options
Plan Name	A user defined name to identify the plan. (Required)
Source Node/Group	This node or node group represents the default source node that will be used in every function when none is specified individually in the functions definition (see below). The source node/group is primarily required for file transfer type functions. It does have certain implications in other functions that will be specified in the function descriptions below. (May be left blank or can be overwritten in the Plan Schedule screen).

Field Name	Field Description / Selection Options
Target Node/Group	This node or node group represents the default target node that will be used in every function when none is specified individually in the functions definition (see below). A target node/group is specified along with a source node/group for file transfers. Most other functions work only with the target node. (May be left blank or can be overwritten in the Plan Schedule screen).
One Instance Only	This prevents the system from running more than one instance of this particular plan at any given time. This would keep an operator from accidentally submitting a plan more than once.
Enable Plan Retry	Enables the plan retry feature for any failed instances of this plan. A plan can be retried from the Plan Activity screen. This is very beneficial when working with a node group and one or more nodes fail. Plan retry allows only the failed portions of the plan to be rerun.
Diagnostic Mode	This option turns on diagnostic mode for this plan. This will cause trace logs to be created for all activity generated by this plan to aid in diagnosing problems. This would be similar to turning on diagnostics for each node involved in this plan but would only log node activity specifically for this plan.
Audit Level	Specifies the level of information recorded in the audit trail during an instance of this plan. Regardless of this setting, a plan start and plan stop informational message along with all error messages are recorded. This field may be overridden during the submission of this plan. One of the following entries should be checked: **None** No additional information is recorded, other than errors. **Summary** Summary information is also recorded including the starting and ending of each phase and the accessing of each individual node. **Detail** This records Audit trail records for every file operation and program/script execution. When executing a script or program that outputs to standard out, if Detail is selected, the output is captured and stored in the audit record. This is an important point, and you should be aware that large amounts of output can cause the audits file to grow as well as take longer for the plan to finish. (An excellent tool when troubleshooting problems)

Plan Phase

Plan Phases provides a means for grouping a series of related functions together. This allows the user to specify when and if the phase runs based on the relative success or failure of other phases. All functions within a phase will run consecutively. In order for two functions to run simultaneously they must be put into separate phases that are set to run concurrently.



```
Plan   g100k        history=Detail   one only
   Phase Phase1    runs in sequence
         Transfer   Fileset=g100k  medium   overwrite
```

Phase Name: Phase1

□ Skip remaining functions if any function completes with return code >= 0

● Runs in sequence

○ Runs after phase ____ completes with return code > 0

○ Runs with phase ____ ● synchronously ○ asynchronously and ● does not propagate return code ○ propagates return code

□ Repeat phase every 0

□ maximum repeat count 0

□ stop repeating if any function returns > 0

□ Cutoff phase ○ at a specific time of day 04:40:32 PM CST ● after running for 0

Field Name	Field Description / Selection Options
Phase Name	This allows the user to assign a descriptive name to the phase. This may be used when another phase needs to reference this phase ("runs after" or "runs with" fields). This name will appear in audit records generated by the plan.
Breaks if any function completes with return code >=	The remaining functions within this phase to be skipped if the return code from any of the functions is greater than or equal to the selected value.
Runs in sequence	Select this entry if you want the phase to run following its proceeding entry listed above (This is the default).
Runs after phase completes with return code	Select this entry if you want the phase to run conditionally after another phase. This phase will only run if the phase specified finishes with a return code that satisfies the expression and value of the two remaining boxes (>, <, >=, <= may be chosen along with a number.
Runs with phase	Select this entry if you want the phase to run at the same time as another phase. Any number of phases may be grouped together in this manner, with all of them starting at the same time. The additional check boxes should be specified in conjunction with this option: **Synchronously** The phase will not be considered finished until the run with phase is also finished. This would keep a phase that is to run after this one, idle until both phases are finished. If this option is checked, the choice to propagate the return codes can be checked as well. This forces the return code on both phases to be set to the higher of the two. **Asynchronously** The phase will run independently of the runs with phase. Any phases set to run after this phase will start as soon as this one is finished.
The next set of fields allows for the configuration of a repeating phase. A repeating phase may be used to create a plan that performs a polling type of operation. A repeating phase is usually preferable to a repeating plan for this type of activity because of less overhead and easier manageability.	

Field Name	Field Description / Selection Options
Repeat phase every # interval	Specify the count and interval for the repeating phase (Example, every 30 seconds). If the phase does not finish the activity specified within the time allotted, the beginning of the next iteration will be delayed.
Maximum repeat count	Specify a maximum number of times for the phase to repeat. This is one way to stop a phase.
Stop repeating if any function return	Specify an expression and value to terminate a phase. This is another way to stop a phase.
Cutoff phase	Use the checkboxes and fields to specify a termination time as either a fixed time of day, or a relative amount of time from when the phase started.

Note about polling plans. As described previously a repeating phase has advantages over a repeating plan, however a combination of the two may offer the best overall solution. Consider the following example:

A plan is required that copies the contents of a directory on a node to a target location every five minutes. This needs to be ongoing indefinitely. The following steps are taken:

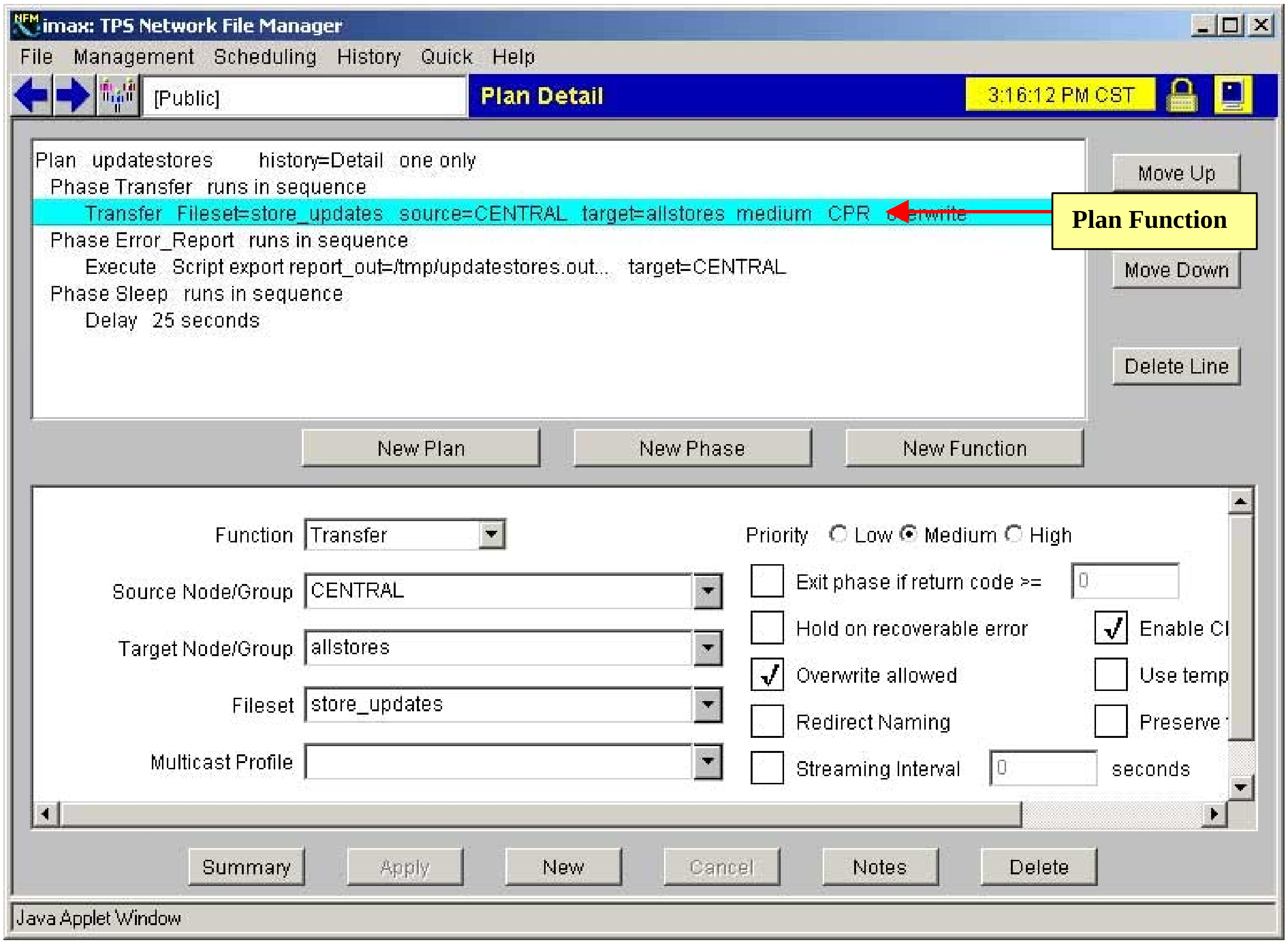
The copying phase is set to repeat every 5 minutes.
The phase is set to terminate at 11:59 pm.
The plan is submitted to run at midnight and repeat every 1-day.

This setup is ideal because it creates a single plan instance for each 24-hour period. This keeps the Plan Activity screen from getting overloaded with plan ID’s. It also keeps the audits grouped nicely, allowing a user to view activity for a particular day by simply clicking on that day’s plan ID in the Plan Activity screen.

Plan Function

Functions specify individual steps to perform in a plan. These can include defining a file transfer, an execute, running another plan, delete on target / source, etc.. Highlights of Plan Function include:

- Exit on return code allows the phase to be exited if a return code is over a specified value.
- Hold on recoverable error allows the function to be suspended (without exiting the plan). This could allow a support person to correct the problem and then release the function.
- Enable checkpoint restart allows for an interrupted transfer of a large file to pickup where it left off.
- Priority for the transfer may be specified with regard to other transfers that may be occurring within NFM.



```
Plan  updatestores       history=Detail  one only
  Phase Transfer  runs in sequence
     Transfer  Fileset=store_updates  source=CENTRAL  target=allstores  medium  CPR  overwrite
  Phase Error_Report  runs in sequence
     Execute  Script export report_out=/tmp/updatestores.out...   target=CENTRAL
  Phase Sleep  runs in sequence
     Delay  25 seconds
```

Function: Transfer

Source Node/Group: CENTRAL

Target Node/Group: allstores

Fileset: store_updates

Multicast Profile:

Priority ○ Low ● Medium ○ High

□ Exit phase if return code >= 0

□ Hold on recoverable error

☑ Enable Cl

☑ Overwrite allowed

□ Use temp

□ Redirect Naming

□ Preserve

□ Streaming Interval 0 seconds

Field Name	Field Description / Selection Options
Function	Defines the type of function. Depending upon the function chosen, some of the remaining fields may change or have no effect. The remaining descriptions indicate what meaning they have with the different function types. Choices include: **Transfer** Copies a fileset from a source node or node group to a target node or node group. **Delete on Source** Deletes a fileset on the specified source node or node group. The Source Prefix and Source Files fields in the filesets record would designate which filenames to delete. **Delete on Target** Deletes a fileset on the specified target node or node group. The Rename Prefix and Target Files fields in the filesets record would designate which filenames to delete, unless they were empty in which case the Source fields would be used. **Execute** Executes a command on a specified target node or node group. **Rename** Renames the entries of a fileset on a target node or node group. The source and rename fields of the fileset must both be specified for this option.. **Run Plan** Submits a plan for execution. The source and target nodes specified become the default source and target node just as if they had been typed in manually from the Plan Scheduler. This function will not complete until the submitted plan completes. The return code from the plan instance will become this functions return code. **Custom FTP** This advanced option allows the use of a customized FTP transfer whereby the user can add additional commands to the default ones used for an FTP transfer. The default commands used for FTP exist in template files (these are in an “attachments” subdirectory in NFM). The default files `ftppush` and `ftppull` may be copied to other names and placed in the subdirectory “attachments/custom” where they may be modified.
	Once this is done these files will be available from the Template pop-up list field in the plan function. It is the users’ responsibility to use

Field Name	Field Description / Selection Options
	the correct base file based on the direction of the transfer. The file `ftppush` is used for sending files from an NFM client to an FTP node, the file `ftppull` copies in the other direction. Never modify these original files. **Delay** This simply allows for a wait to occur before continuing on to the next function or phase. The time can be specified in seconds or as an absolute time of day. **Release** This function releases (stops) a previously started streaming file transfer on a given node or node group for a given fileset.
Source Node/Group (Supported Function Types: *Transfer, Execute, Delete on Target, Delete on Source, Run Plan, Custom FTP, Release*)	This node or node group specifies the source node(s) for a file transfer and designates where to delete files for a Delete on Source. (May be left blank or can be overwritten in the Plan Schedule screen).
Target Node/Group (Supported Function Types: *Transfer, Execute, Delete on Target, Delete on Source, Rename, Run Plan, Custom FTP*)	This node or node group specifies the target node(s) for most functions. (May be left blank or can be overwritten in the Plan Schedule screen).
Fileset (Supported Function Types: *Transfer, Delete on Target, Delete on Source, Rename, Custom FTP, Release*)	Allows selection of an existing fileset definition
Command (Supported Function Types: *Execute*)	Allows for a single line command to be typed in that is to be run on the target `node/nodegroup`.

Field Name	Field Description / Selection Options
Multicast Profile (Supported Function Types: *Transfer*)	Allows the selection of an existing multicast profile name. When left blank, this specifies that the transfer will not use IP multicast. (Multicasting is only supported if your network equipment allows it)
Priority (Supported Function Types: *Transfer*)	A priority for a file transfer with regard to other file transfers that are taking place within NFM at the same time. This only applies to socket type transfers. This field is only used if the prioritization option is turned on in the System Settings.
Exit if return code >= (Supported Function Types: *Transfer, Execute, Delete on Target, Delete on Source, Delete on Source, Rename, Run Plan, Custom FTP, Release*)	The remaining functions within this phase are to be skipped if the return code from this function is greater than or equal to the selected value. This field overrides the default termination value specified in the phase for the current function.
Checkpoint restart (Supported Function Types: *Transfer, Custom FTP*)	Specified to allow a retransmission of a file to transfer only the portion that did not copy. (** Checkpoint restart is only supported between NFM Client nodes **)
Overwrite allowed (Supported Function Types: *Transfer, Rename, Custom FTP*)	Allow transfers or renames to automatically overwrite any existing target file; otherwise, an error will occur if the target file already exists.
Use temporary names (Supported Function Types: *Transfer, Custom FTP*)	Instructs NFM to use temporary names while transferring files. This will cause the files to be copied to their corresponding destination names followed by an underscore '_.' After the copy is successful, each file will be renamed to its proper destination name. This option allows other software that is polling for the presence of a specific file, to be assured it has a complete copy of the file.
Redirect Naming (Supported Function Types: *Transfer, Delete on Target, Delete on Source, Custom FTP*)	Reverses the meaning of the source and target fields in the fileset definition for this specific function. This allows specifying a reverse copy (source prefix and files become target, and, rename prefix and files become source). This allows using an existing fileset definition to copy files in the reverse direction for what is implied by the fileset definition.
Preserve Timestamps (Supported Function Types: *Transfer*)	Sets the timestamp on a target file equal to that of the source file during a transfer.

Field Name	Field Description / Selection Options
Streaming Interval (Supported Function Types: *Transfer, Custom FTP*)	This transfer is a streaming type transfer. If so, also specify a non-zero number in the following field to designate the streaming interval in seconds. A streaming file transfer allows a file to be transferred as another application or command is writing to it. In order to use streaming type transfers the file must have data appended to it and any application that writes to the file must not restrict NFM from opening the file simultaneously. (** Streaming can only occur between NFM Client nodes **)
Plan Name (Supported Function Types: *Run Plan*)	Specifies the plan you want to run.
Asynchronous (Supported Function Types: *Execute*)	Do not wait for the program to terminate
Use Shell (Supported Function Types: *Execute*)	NFM will attempt to run the command using the native operating system shell or command line interface. This would be necessary for doing things like redirecting output. Also, by specifying this option, the Command field is changed to multiple line text box, called Script that allows for a multiple line command to be entered. This multiple line command will now be run as if it were a local batch file on the target nodes, which eliminates the need for separately sending down a batch file and then executing it.
Timeout (Supported Function Types: *Execute*)	Allows a time-out value (in seconds) to be specified. If the command is not finished executing within this amount of time, the command is killed and an error is reported.
Error Level (Supported Function Types: *Execute*)	Normally, when an execute function finishes with any value other than zero, the function is flagged as an error, with an error type audit message. This field allows setting a threshold to determine when this occurs. If the actual exit code is less than the value specified in this field, the function will not be considered in error. There will not be an error audit message and the function will not be retried during a plan retry operation. The function will still finish with the exit code, which will still filter up to affect the plans return code.
Append to Target (Supported Function Types: *Transfer*)	Causes the source file(s) to be appended to any existing target file(s) rather than simply copied over them. It is recommended that 'Use temporary names' be specified in conjunction with this option (see description above).
Clear contents first (Supported Function Types:	If an append operation (previous option), is being done. This causes the target file to be cleared prior to the append being done. At first glance this would appear to negate an append operation

Field Name	Field Description / Selection Options
Transfer)	At first glance this would appear to negate an append operation, and indeed would do so in a one-to-one transfer without wildcards being used. However, an append operation may also be used to combine files from multiple sources, in the following two scenarios: 1) A many-to-one transfer using a fixed target file name (no node extensions & no imbedded environment variables). 2) A transfer involving wildcard names that specifies a fixed target file. Either of these may combine multiple files and optionally append them to the end of existing data or not based on this option.
Delete source file after completion (Supported Function Types: *Transfer*)	This option causes a delete on source operation to occur automatically if the file(s) are successfully transferred. If a delete is desired, this is preferable to specifying a separate function to accomplish this task for two reasons: 1) It guarantees that a delete will not occur if the file(s) are not transferred. This would otherwise require the user to carefully select the correct termination logic to avoid accidentally deleting undelivered files. 2) If wildcards are being used, it avoids the potential timing problem of deleting a file that appeared after the transfer was finished but before the delete was done.

NFM GUI

EXTENDED CONFIGURATION

This section covers additional configuration needed for setup depending on your use of these features. This covers the following GUI configurations / screens:

Node Access Points (NAP)
Multicast Profiles
System Settings
Quick Functions
Servers
Watch Files

Node Access Points (NAPs)

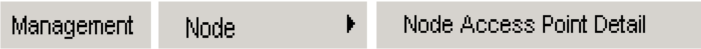
A Node Access Point (or NAP) creates a reference point to a node that may be used in place of referring to a node directly in most places where a node name is required. This serves several purposes:

1. Creates node visibility outside of its usergroup. By creating a NAP in one usergroup that references a node in another, users may now be given access to a node without having to add that node's usergroup to the user's access list.

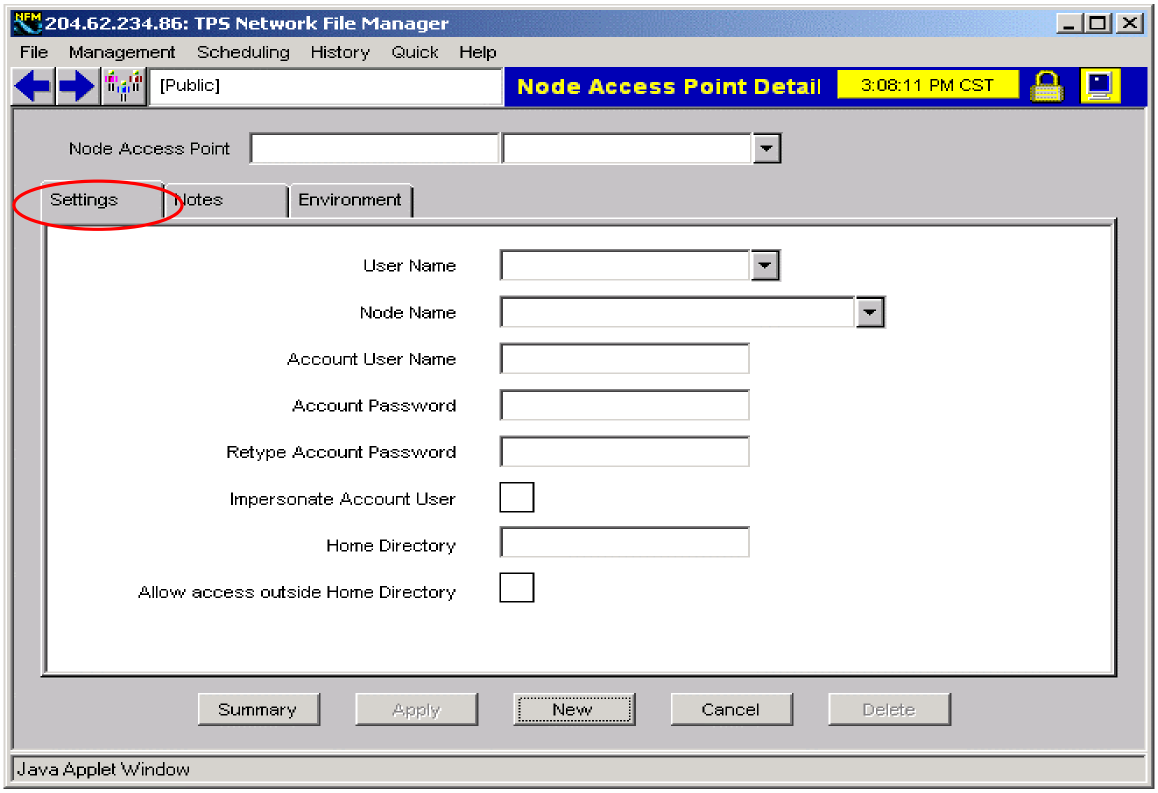
2. Restrict access on a node to a particular directory tree (or set of qualifiers for OS/390).

3. Provide different levels of access to the same node by different users or user groups. This is possible because any number of NAPs may point to the same node. Additionally, a NAP may be defined for common access among members of a usergroup, or a NAP may be created that individually sets the access for a particular user.

4. Allow for an account user name and password to be configured. This will override the user name and password supplied for an FTP type node, allowing for multiple accounts to be accessed without having to create a different node for each one. This account name and password may also be used for user exit authentication purposes.



(to access the Node Access Point screen)



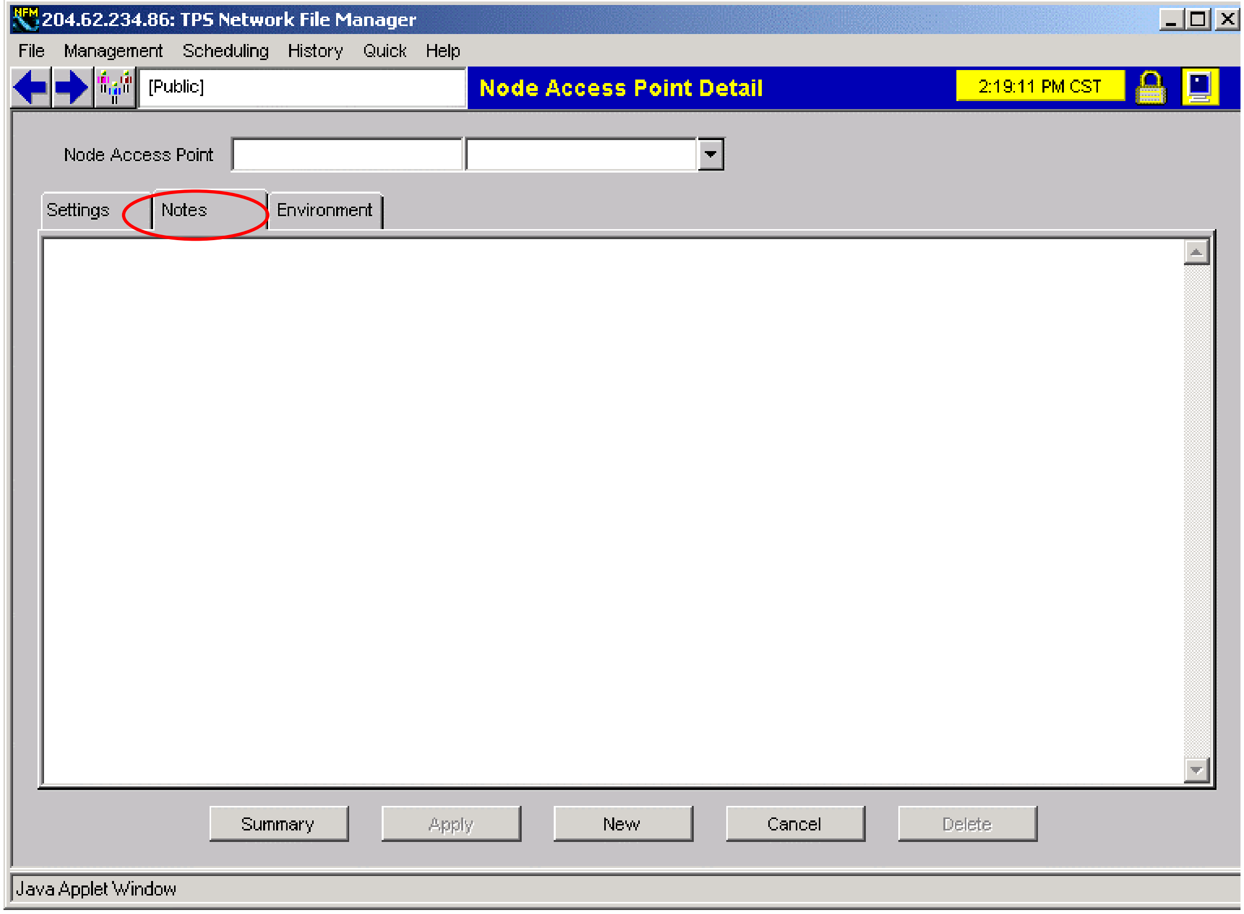
Field Name	Field Description
Node Access Point Name	User-selected name for the given NAP record. Type in a new entry, or select an existing entry (along with user name if desired). This name combines with the next field `User Name` (which may be blank) to create a unique entry. The name chosen may be the same as the node name to which this NAP refers. Although doing this might seem confusing, it allows a NAP to be transparently introduced to an existing NFM configuration without having to change all of the plans that reference this node to use the NAP instead. This is described in more detail below.

Field Name	Field Description
User Name	If this field is set to an existing user name, then this NAP will only be used when the NAP name (above) is chosen or referenced by this particular user. (For a node reference within a plan, this would be the user who submitted the plan). If this field is left blank, then the NAP is considered generic and will be used by default if a user specific NAP by the same name is not found (described in more detail below).
Account User Name **Account Password**	The account user name and password fields are optional and provide one of two things: 1. The name and password fields may be used to satisfy a login request for an FTP type node. If they are specified here, they override any account name and password fields that may otherwise be specified in the node or model record. This allows for different accounts at the same site to be used with a single node record. 2. The name and password fields may be supplied to a user exit routine (if one exist) for the NFM client type node being accessed.
Home Directory	A home directory may be specified here. A home directory provides either an enforced starting directory for access or a default location, depending upon the setting of the next field (Allow access outside of Home Directory) described below. Note, the use of a home directory here does not circumvent the restrictions of the Paths folder settings in the corresponding node or model.
Allow access outside Home Directory	If a home directory is specified, this check box indicates whether or not access is allowed outside of it. If this box is unchecked, any reference to a file name on this node is automatically appended to the home directory. The only exception to this, a file name that exactly matches the prefix (up to the length of the prefix) is not appended, but referred to directly. If this box is checked, a file name that is not fully qualified (does not start with forward slash or drive letter) is automatically appended to the home directory. A fully qualified name is referred to directly (home directory is ignored). This only applies to non OS/390 nodes (an OS/390

Field Name	Field Description
	data set name is always appended to the home directory).

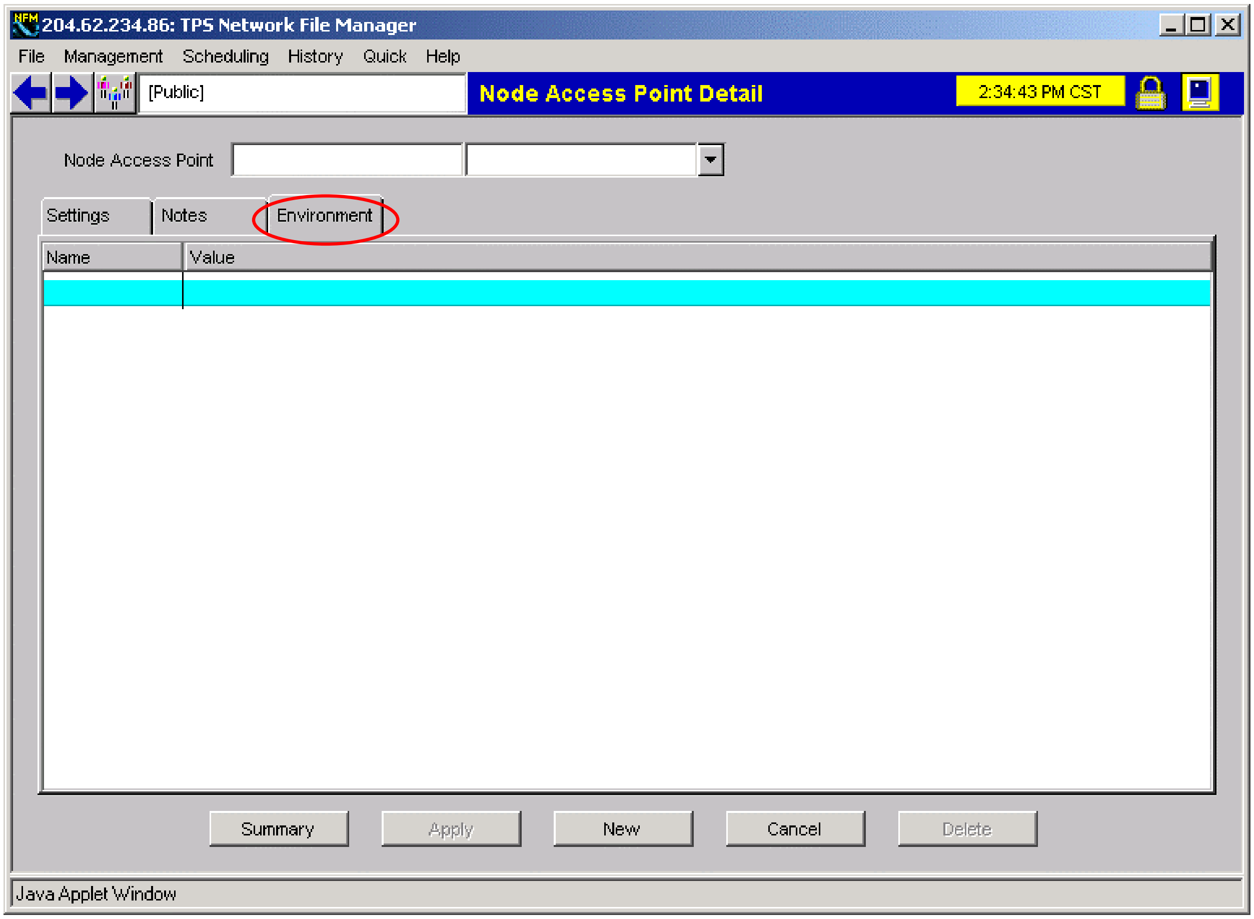
Notes Tab

The Notes tab provides a text window for storing text specific to this NAP record.



Environment Tab

The Environment tab allows users to create custom environment.



A NAP name is generally interchangeable with a node name. It may, for example, be the target of a plan transfer function, or a member of a node group, etc. Besides making it easier to introduce its usage into an existing environment, this allows for a simple customization of access based on the user or user groups involved.

When a node name is referenced, say during a transfer, the following lookup is actually done until a match is found:

- The system searches for a NAP record whose name matches the node name requested and whose user name matches the user performing the operation.
- If not found, the system searches for a NAP record whose name matches the node name requested and whose user name is blank.
- If neither of these are found, a simple search for an actual node record with a matching name is done.

This would allow an administrator for example, to restrict one user to accessing only a particular directory on a node by introducing a NAP record with the node and user's name. This would have no affect on other users accessing the same node.

In a similar manner a NAP added to a particular usergroup that referenced a node in a different usergroup, would have no affect on any users outside the usergroup who were accessing the same node.

Multicast Profiles

When a fileset is transferred from a single source client to a large group of target clients, the IP multicast protocol can provide dramatically increased performance. To realize this performance, several communications parameters must be carefully tuned for the network topology, traffic and end-node capabilities in use during the transfer.

IP multicast is similar to broadcast in that the source node transmits a single packet that is then received by multiple target nodes. IP multicast differs from broadcast in two important ways:

1. broadcast packets cannot be “forwarded” across a router. IP multicast packets can be forwarded across multiple routers.
2. broadcast packets can only be received by nodes on the same subnet, even if they are on the same physical LAN. Nodes on any subnet can receive IP multicast packets even if they are not on the same physical LAN.

IP multicast may be used effectively when:

1. The same fileset is to be sent to multiple targets.
2. The source node is an NFM client.
3. The target nodes are NFM clients.
4. Both source and target clients have enough temporary space to store a compressed version of the fileset in the clients’ `/tmp` directories.
5. The network is configured to properly forward IP multicast packets.
6. The network has sufficient available bandwidth to effectively carry IP multicast traffic.

A multicast profile is a collection of communication parameters that allow precise tuning to match the network. A single multicast profile may be specified for multiple functions in multiple plans. This is similar to the reuse of models in the definition of nodes.

Unlike the TCP/IP protocol used for one-to-one transfers, the IP multicast protocol is not inherently reliable. IP multicast packets may be dropped by the network or by the target client’s IP stack. The base IP multicast protocol makes no provision for ensuring that the data arrives at its destination or is in the proper sequence.

NFM clients implement an algorithm for reliable file transfer between NFM clients using IP multicast. The basic transmission algorithm consists of prepare, announce, transmit, confirm and apply steps. The transmit and confirm steps are automatically repeated as necessary.

Only the transmit step uses IP multicast. During the announce and confirmation steps, the source client opens a TCP/IP connection to each target node. The source client can open multiple simultaneous TCP/IP connections. The Announcement Channels and Confirmation Channels fields of the Multicast Profile control the number of simultaneous connections.

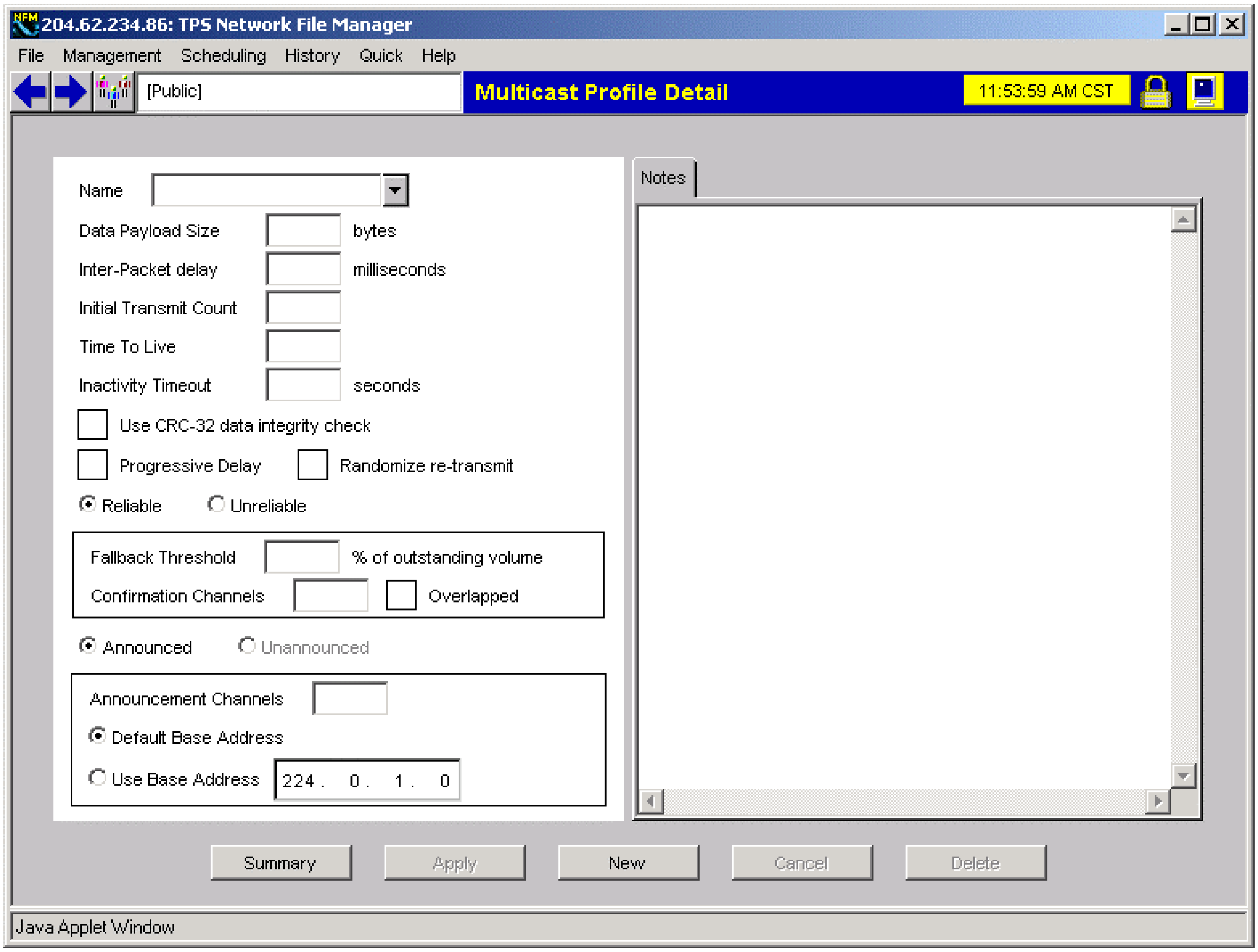
To reduce the computational load on the source client and facilitate partial retransmission of data, a fileset is prepared on the source client before it is sent via IP multicast. This preparation of a fileset for multicast transfer includes compression, encryption and storage of the fileset on the source client. This occurs only once on the source client at the start of the IP multicast transfer. The apply of a fileset includes decryption, decompression, storage of the files and removal of the temporary file on the target clients. This occurs once on each target client after all of the fileset data has been received.

In some cases, an IP transfer to a particular client may not be possible. If an IP transfer function fails, the NFM server will retry the transfer using the default (non-multicast) transfer method.

The suggested values below are intended only to provide a starting point. Some experimentation will be necessary to achieve optimal results on a particular network under a particular load.

Management | Multicast Profile ▸ | Multicast Profile Detail

(to access Multicast Profile screen)



Name

Data Payload Size bytes

Inter-Packet delay milliseconds

Initial Transmit Count

Time To Live

Inactivity Timeout seconds

□ Use CRC-32 data integrity check

□ Progressive Delay □ Randomize re-transmit

● Reliable ○ Unreliable

Fallback Threshold % of outstanding volume

Confirmation Channels □ Overlapped

● Announced ○ Unannounced

Announcement Channels

● Default Base Address

○ Use Base Address 224 . 0 . 1 . 0

Field Name	Field Description
Name	This is the user-selected name that identifies this multicast profile.
Data Payload Size	This is the number of data bytes that will be transmitted in a single IP multicast packet. An additional four to eight bytes of overhead will also be sent in each IP multicast packet. Values below 128 may yield poor performance due to the larger number of packets that must be transmitted and confirmed. Values larger than 1024 may cause many packets to be dropped by the network or the target client's IP stack if non-multicast traffic is present. Values in the range of 256 to 1024 bytes will often give good results. Suggested value: 1024
Inter-Packet delay	This is the number of milliseconds that the source client will delay between transmitting data packets. This delay is important to allow the target clients time to read and store the data. This delay also allows for TCP/IP traffic to be processed during the delay. Longer delays will cause slower overall transmission speeds. Delays that are too short may cause many packets to be dropped resulting in more retransmission of those packets. Suggested value: 20
Initial Transmit Count	This is the number of times that the fileset will be transmitted before beginning the first confirmation step. On lightly loaded, reliable networks, a value of 1 may be sufficient. For networks with higher traffic or less reliability, or when there are very many target clients, a value of 2 or more may provide better overall performance. Suggested value: 1
Time to Live	This value corresponds to the number of routers that the IP multicast packets must cross to reach their farthest destination. The allowable range is from 1 to 255. Suggested value: 10
Inactivity Timeout	This is the number of seconds that the target client will wait to receive multicast data. If no multicast data is received within this time, the target client will stop attempting to receive multicast data. Suggested value: 600

Field Name	Field Description
Use CRC-32 Data Integrity Check	This box should be checked to enable CRC-32 checking of each data packet. This protects against data corruption during transmission. On highly reliable networks, this may not be needed. Suggested value: yes
Progressive Delay	Checking this box will cause the source client to delay progressively longer each time a block is retransmitted. The progression is 25 percent additional delay for each retransmission up to a maximum of five times the initial delay. This allows setting of a lower initial Inter-Packet delay while dynamically accommodating slower clients or unpredictable network traffic. Suggested value: yes
Randomize Transmit	Checking this box will cause retransmitted packets to be sent in a pseudo random order. This often relieves cyclical problems such as dropping every n'th packet. Most cyclical problems result from network or target client processing limitations. Suggested value: yes
Reliable / Unreliable	Choosing reliable guarantees that either the fileset will be received correctly or any failure will be reported. Choosing unreliable will not guarantee that the fileset will be received and failures at the target clients may not be reported. Suggested value: Reliable
Fallback Threshold	It may be ineffective to continue multicasting packets that are consistently dropped before being saved at the target client. In this case, the source clients can fallback to non-multicast (TCP/IP) for the remaining data blocks. During confirmation, the source client evaluates the number of outstanding data blocks (packets) that the target client has received since the last confirmation. If the percentage is less than this value, the source client will begin fallback data transmission for the outstanding data. Suggested value: 20
Confirmation Channels	The source client can perform multiple simultaneous confirmations with multiple target clients. Each of these confirmations uses a "channel" consisting of one system thread and one socket. Source client resources are limited.

Field Name	Field Description
	In particular, the source client may be limited in the number of simultaneous open sockets or the number of available threads. If either of these numbers is less than the number of target clients, the source client should be restricted to less than the number available. Suggested value: 2
Overlapped	The confirmations may be performed while simultaneously multicasting data. When the number of target clients is much larger than the number of confirmation channels, this option may increase overall performance on some networks. Note that some multicast data will be dropped because of the higher priority TCP/IP traffic used by the confirmation channels. Suggested value: no
Announced	This is required for reliable transfers. Suggested value: announced
Announcement Channels	Similar to confirmation channels, announcement channels are used by the source client to set up the multicast transfer with the target clients. Suggested value: 2
Default Base Address / Use Base Address	The NFM server will select a multicast address for the transmission. The address will be selected from the multicast pool (224.0.1.1 through 239.255.255.254). The next available address (not already in use by the NFM server) from the specified base address will be used. The default base address is 224.0.1.1. A specific base address may be specified to avoid conflicts with other IP multicast applications or to facilitate network diagnostics. Suggested value: default base address
Default Multicast Address / Use Multicast Address	The NFM server will assign a multicast address for the transmission. The address will be either the default (224.0.1.1) or the value specified. If this address is in use by another NFM multicast transmission, the transfer will fail.
Raw Data Only	When selected, this option will cause the source client to send only the content of the data files without any information about file names or permissions

System Settings

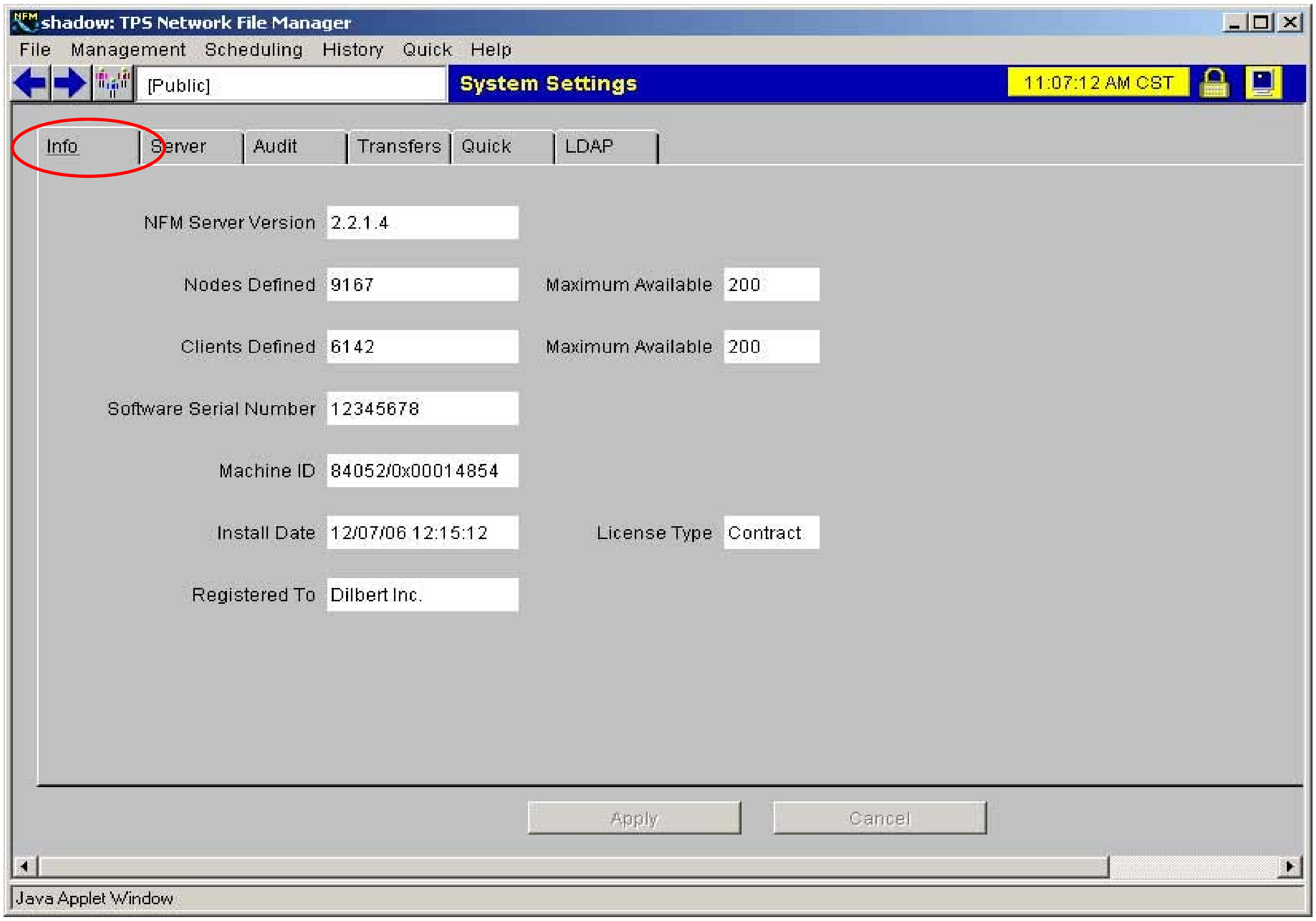
System Settings displays and maintains overall system information as well as settings that are used to customize NFM for a particular customer's environment.

Management System Settings

(to access System Settings screen)

Info tab

The Info tab contains version and license information.



shadow: TPS Network File Manager
File Management Scheduling History Quick Help
[Public]
System Settings
11:07:12 AM CST
Info Server Audit Transfers Quick LDAP
Apply
Cancel
Java Applet Window

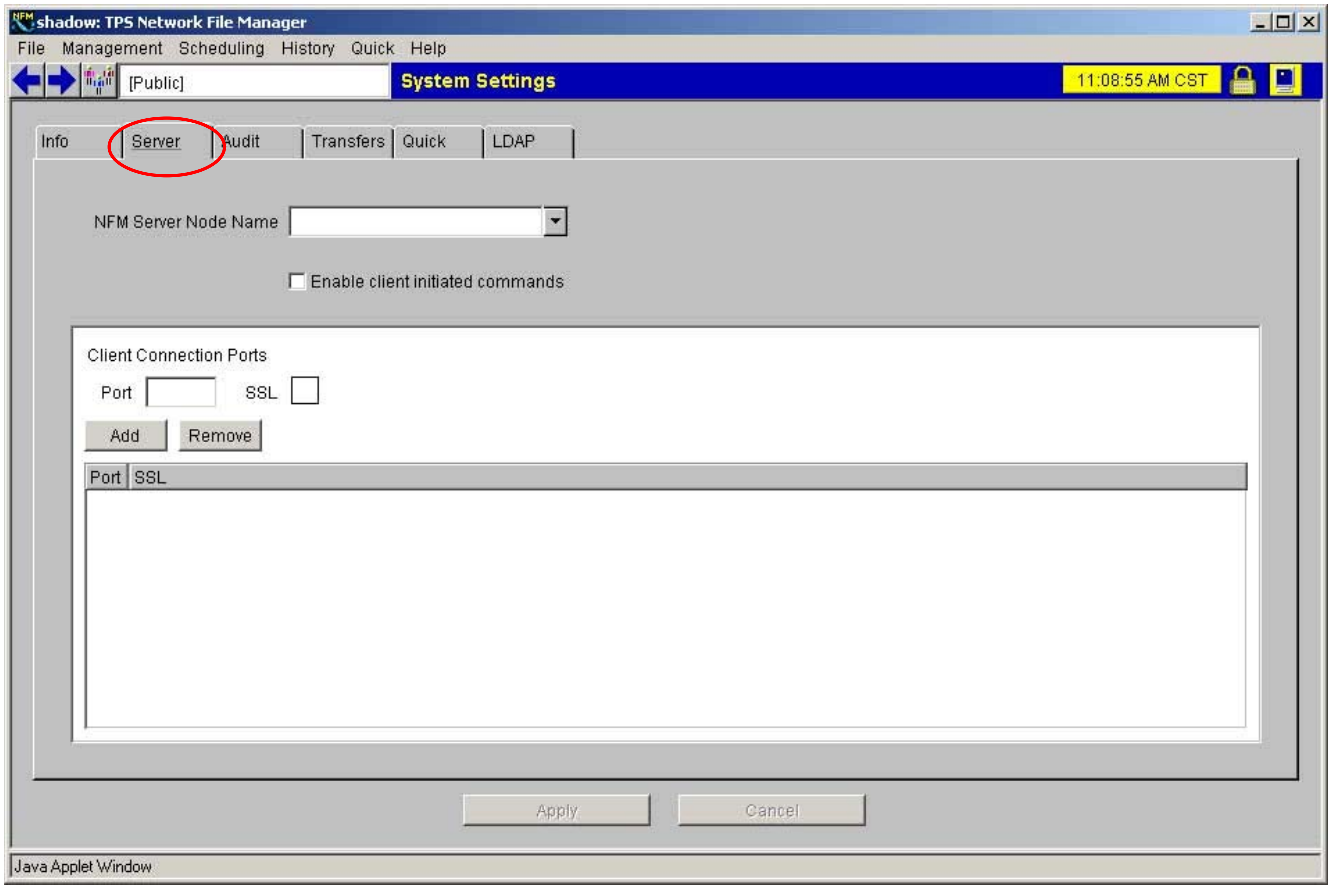
NFM Server Version	2.2.1.4		
Nodes Defined	9167	Maximum Available	200
Clients Defined	6142	Maximum Available	200
Software Serial Number	12345678		
Machine ID	84052/0x00014854		
Install Date	12/07/06 12:15:12	License Type	Contract
Registered To	Dilbert Inc.		

Field Name	Field Description
NFM Server Version	Current version number of the NFM Server
Nodes Defined / Maximum Available	Total number of nodes defined and maximum available

Field Name	Field Description
Clients Defined / Maximum Available	Total number of NFM Client nodes and maximum available
Software Serial Number	Serial Number assigned by TPS
Machine ID	Server's hardware /machine ID

Server tab

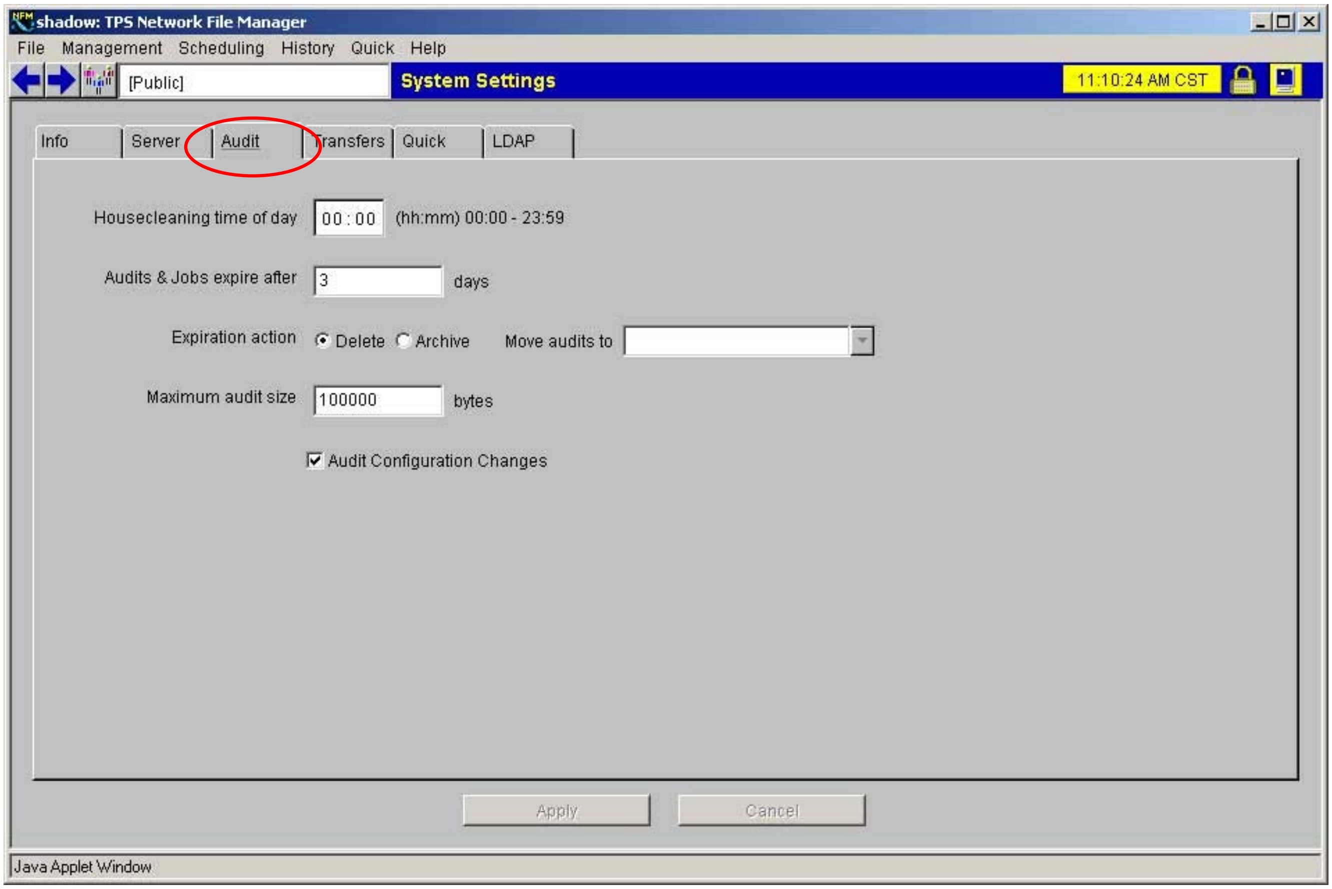
The Server tab contains overall configuration items for the server



Field Name	Field Description
NFM Server Node Name	Node record that represents the NFM Server
Enable client initiated commands	Enables the remote server interface. (What is the remote server interface?)

Audit tab

The Audit tab contains options that are specific to the audit keeping feature of NFM.



Housecleaning time of day 00 : 00 (hh:mm) 00:00 - 23:59

Audits & Jobs expire after 3 days

Expiration action ● Delete ○ Archive Move audits to

Maximum audit size 100000 bytes

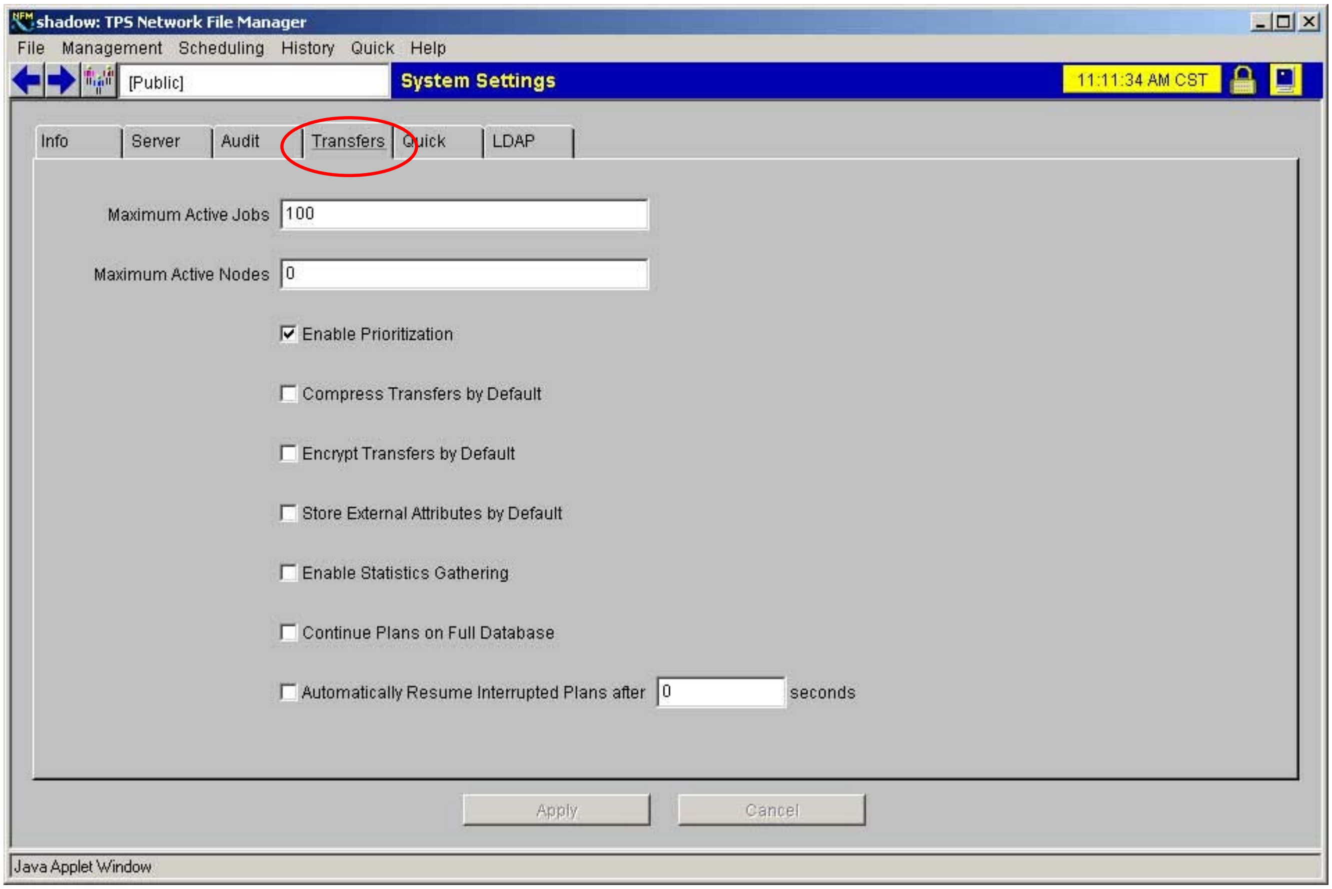
☑ Audit Configuration Changes

Field Name	Field Description
Housecleaning time of day	Time of day to perform activities such as removing old audit records and finished plan instances.
Audits & Jobs expire after # days	How long old audit records and completed plan instances should be kept
Expiration action	What to do with audits after they have expired. Delete removes them from the system while archive moves them to the archive location specified in the 'Move audits to' field.

Field Name	Field Description
Move audits to	This designates the default archive location. An archive location is a named entry that represents a physical directory in the NFM server file system. A predefined entry "ARCHIVE" is available. Additional locations may be created by the user using the NFM command line interface (reference the sub-command `putasource` in Appendix D: NFM Commands & Parameters).
Maximum audit size	The maximum size, in bytes, of a single audit message. This limits the size of audts that get created from the output of plan execution type functions. It will cause the output to be truncated.
Audit Configuration Changes	Create audit messages when an operator modifies any part of the NFM's configuration

Transfers tab

The Transfers tab contains options that are specific to file transfers.



shadow: TPS Network File Manager
File Management Scheduling History Quick Help
[Public]
System Settings
11:11:34 AM CST
Info Server Audit Transfers Quick LDAP
Apply Cancel
Java Applet Window

Maximum Active Jobs: 100

Maximum Active Nodes: 0

☑ Enable Prioritization

□ Compress Transfers by Default

□ Encrypt Transfers by Default

□ Store External Attributes by Default

□ Enable Statistics Gathering

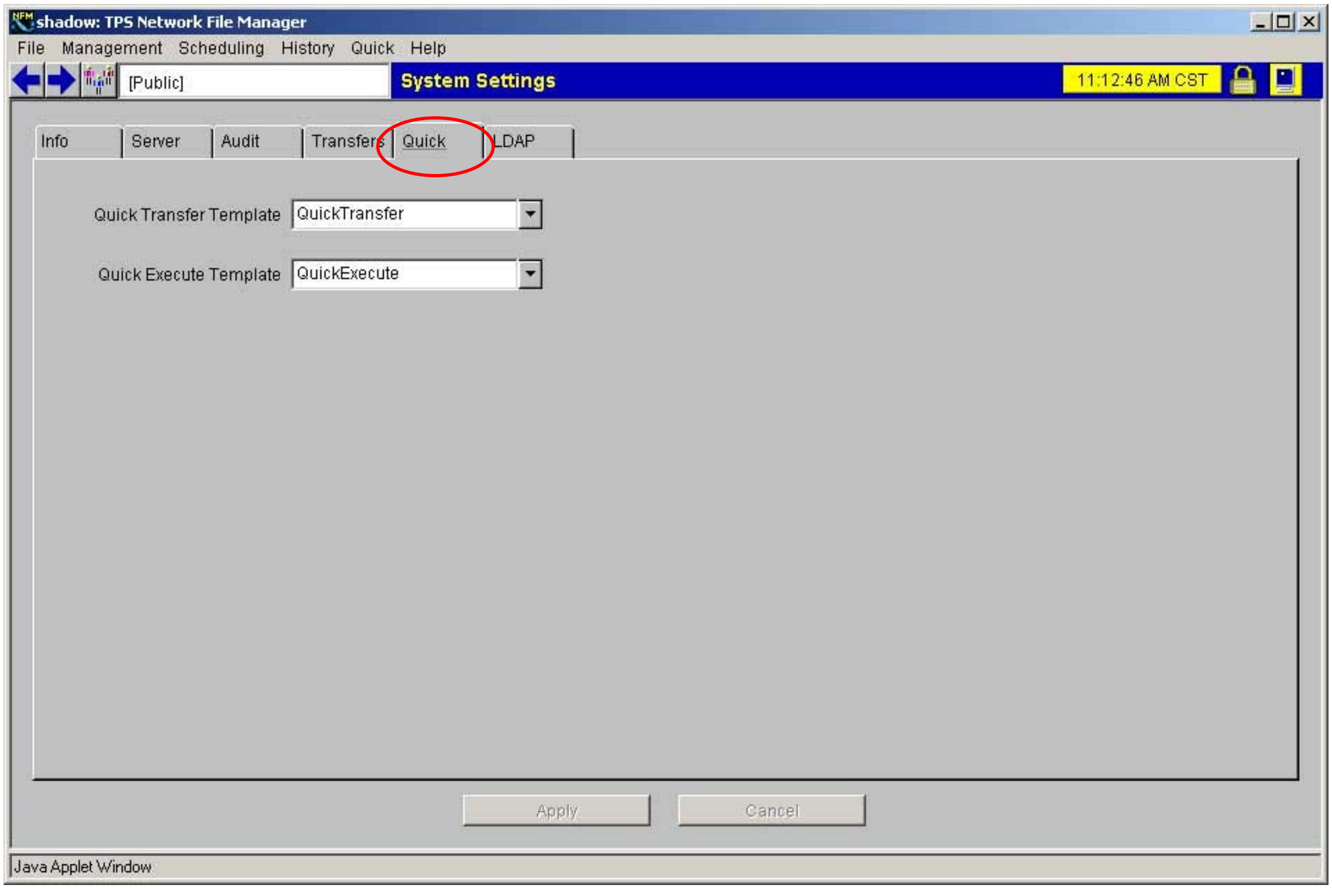
□ Continue Plans on Full Database

□ Automatically Resume Interrupted Plans after 0 seconds

Field Name	Field Description
Maximum Active Jobs	The maximum number of plans that can be running simultaneously.
Maximum Active Nodes	Maximum number of nodes that can be actively involved in file transfers at any given moment. This is independent of the 'Maximum Active Nodes' setting available on a per node group basis.
Enable Prioritization	Gives the option for individual transfers to have priority over others.
Compress Transfers by Default	Specifies system wide use of compression during file transfers. (can be overridden in the Fileset)
Encrypt Transfers by Default	Specifies system wide use of encryption during file transfers. (can be overridden in the Node and Model)
Store External Attributes by Default	Specifies system wide use of store external attributes during file transfers. (can be overridden in the Fileset)
Enable Statistics Gathering	Enables the gathering of statistics during plan activity for use by the performance monitoring tools.
Continue Plans on Full Database	Allows plans to continue processing even if the NFM database runs out of space and can no longer store additional data. By default, if the system runs out of space, all plan activity will go on hold. Once space becomes available, plan activity will resume as if nothing happened.
Automatically Resume Interrupted Plans after # seconds	A plan instance becomes "Interrupted" if the NFM server computer is rebooted or if a backup NFM server takes over for a crashed system. The interrupted state of a plan instance is very similar to the cancelled state. It may be resumed manually from the Plan Activity screen, or by checking this option, all interrupted plan may be automatically resumed, when the NFM server starts/restarts. The delay in seconds may be specified to allow other action (such as human intervention) to occur after a reboot, but prior to resuming failed plans

Quick tab

Quick tab contains options that are specific to the Quick menu functions.



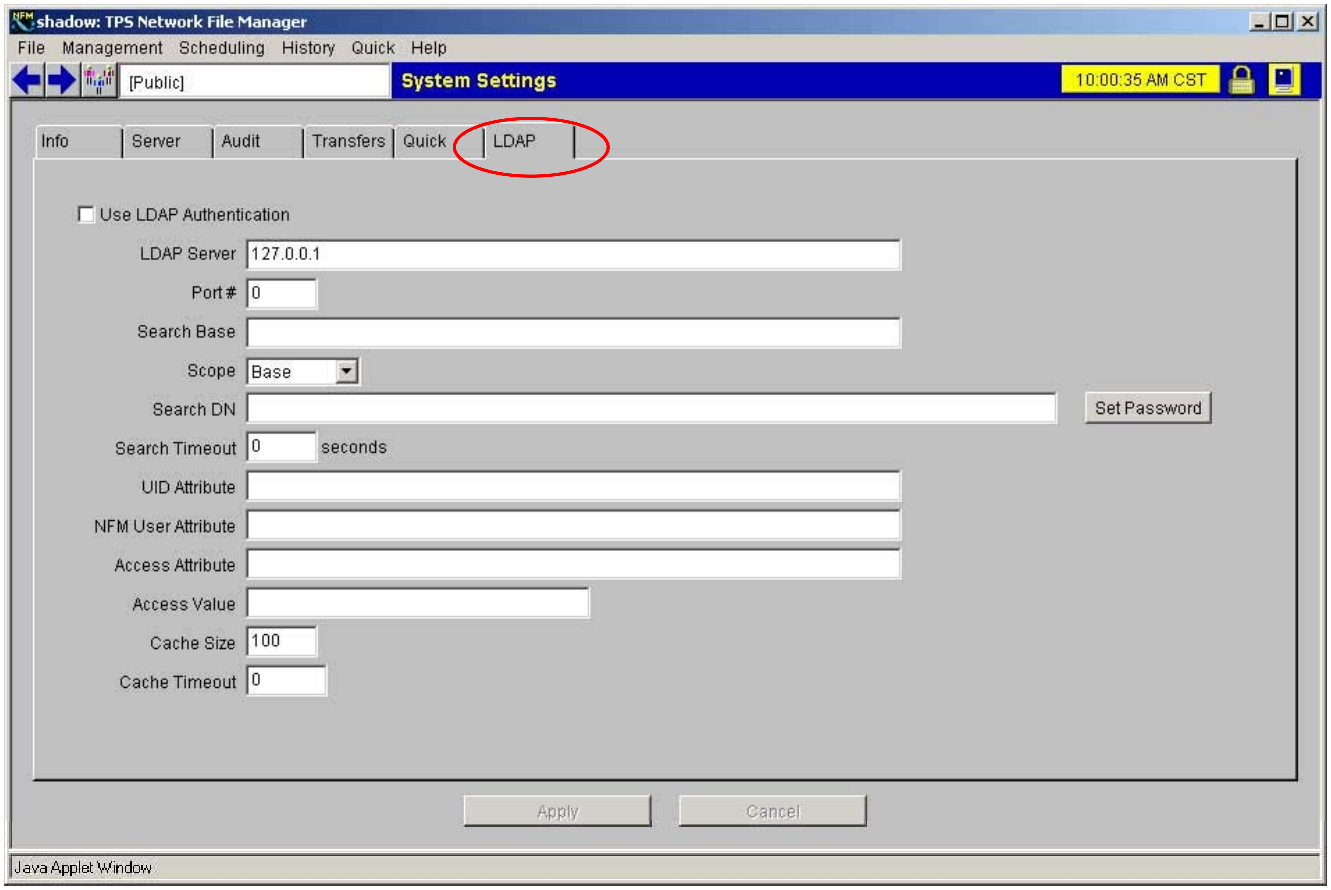
Quick Transfer Template: QuickTransfer

Quick Execute Template: QuickExecute

Field Name	Field Description
Quick Transfer Template	Used to configure the NFM plan that represents a Quick Transfer.
Quick Execute Template	Used to configure the NFM plan that represents a Quick Execute.

LDAP tab

NFM can be configured to perform user authentication by an LDAP lookup when logging on to, and using the NFM user interface (both the GUI and the command line interface). The fields configured on this tab supply the information needed by NFM to verify the user ID and password with the LDAP server.



☐ Use LDAP Authentication	
LDAP Server	127.0.0.1
Port #	0
Search Base	
Scope	Base
Search DN	
Search Timeout	0 seconds
UID Attribute	
NFM User Attribute	
Access Attribute	
Access Value	
Cache Size	100
Cache Timeout	0

Field Name	Field Description
Use LDAP Authentication	Indicates whether or not LDAP authentication is enabled. For security reasons, this field may not be changed from this screen. It can be enabled or disabled only by using the command line interface, and should only be enabled after each of the other fields on this screen have been properly configured.
LDAP Server	This is the DNS name or IP address of the LDAP server.
Port #	This is the port number served by the LDAP server if other than the default. (`0` = use default).

Field Name	Field Description
Search Base	This is the base entry where NFM will begin its LDAP search.
Scope	This is the scope of the LDAP search below the LDAP Base. Select from the provided menu: (Base | One Level | Subtree).
Search DN	This is the distinguished name that NFM will use to bind to the LDAP server. This DN must have "search" permission.
Search Timeout	This is the maximum amount of time in seconds that the LDAP server has to respond to an LDAP search.
UID Attribute	This is the name of the LDAP attribute that will contain the user ID supplied for the NFM login. If left empty the default value `uid` will be used.
NFM User Attribute	This is the name of an LDAP attribute that contains an existing NFM user ID that will be used to inherit user settings (permissions, preferences, etc.) from. If this field is specified, each user that logs in will be expected to have the field of this name set to the NFM user that they will inherit settings from. In this case it will not be necessary to have an actual NFM user defined with the login user's name. (Note that multiple LDAP users may inherit settings from the same NFM user). If this field is left blank then each user that logs in must have an actual NFM user defined for them, whose name matches their LDAP login ID.
Access Attribute	This is the name of an optional LDAP attribute that will be checked for permission to access the NFM application. If left empty this option is not used.
Access Value	This is the value that must be present in the LDAP access Attribute to allow access to the NFM application. If left empty this option is not used.
Cache Size	This value defines the number of cached LDAP authentication requests that may exist at any given moment. The NFM LDAP cache exists to improve the performance of the NFM user sessions by not forcing each and every command to be individually authenticated through the LDAP server. Ideally this value should be greater than or equal to the number of users that may access the NFM server.

Field Name	Field Description
Cache Timeout	(See description of the cache above). This defines how long an authentication request may remain cached prior to forcing it to be re-authenticated through the LDAP server. A long timeout will improve performance by minimizing lookups, however, if a user is denied access to NFM via an LDAP configuration change, it will take this long (the timeout value) before the change takes effect.

** If LDAP authentication is selected for use with NFM, it is not actually necessary to create an NFM user record for each potential user. Instead, LDAP users may inherit the settings of an existing NFM user record (see “NFM User Attribute” below).

Quick Functions

NFM quick functions enable users to perform simple operations on demand (non-scheduled). The following types of operations are available:

Quick Transfer User can perform a simple file transfer (one file) from a source node to a target node.

Quick Execute User can execute a non-interactive command on a target node (similar to the execute function within a plan).

Quick Plan User can run a pre-defined plan.

Quick Monitor User can display the actual plan information and status.

The quick functions, in addition to offering a simplified way for users to perform certain tasks, also allows for a class of users to be created that can benefit from the services of NFM while being completely restricted as to which resources they may access. Users can be restricted to any combination of the above types of functions as well as being restricted to certain nodes, and even certain directories of certain nodes, based on their user ID.

Users performing the quick transfer and execute type functions, can enter the appropriate parameters for the operation, selecting nodes from pop-up lists and file descriptions from browse menus, or typing both in manually. Once these parameters are set, the entire operation may be saved under a user-selected name, to simplify performing the same operation in the future by being able to simply select the name from a menu.

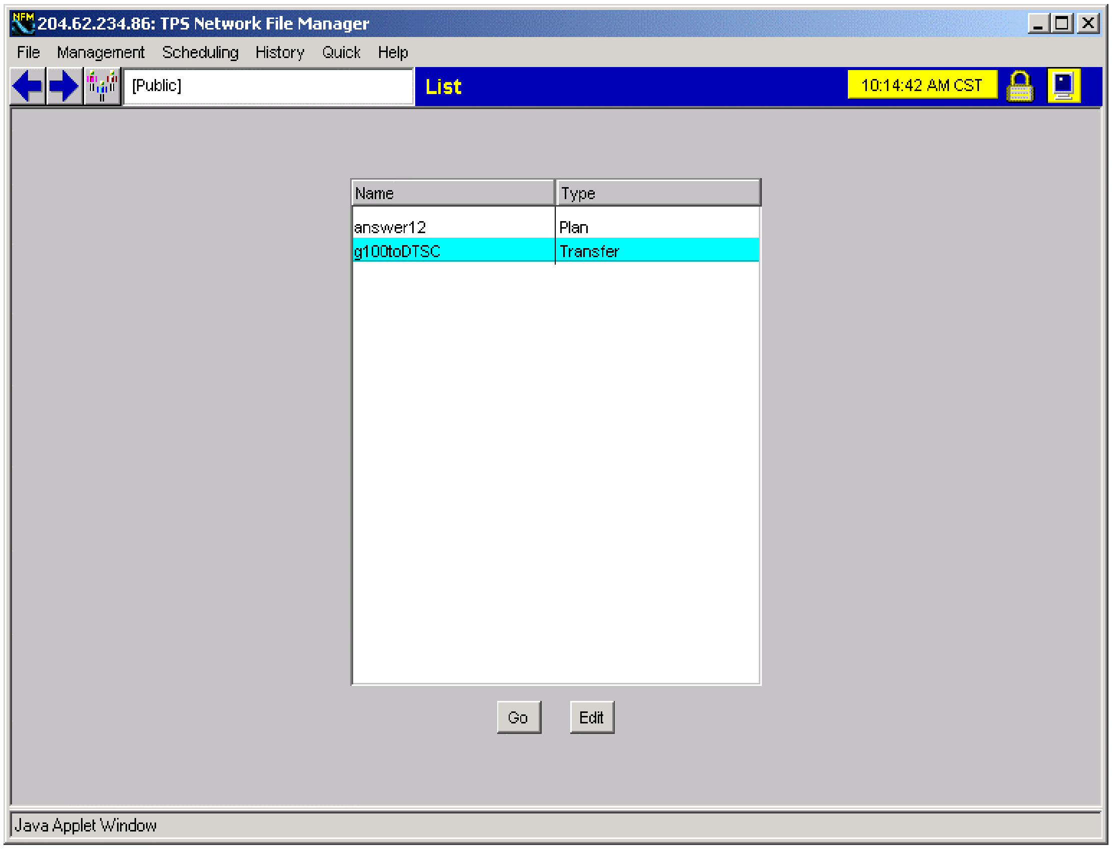
The next sections provide detailed instructions for operating the different quick function menus. Following those is a section on configuration that is performed by administrators to make and assign different quick operations to different users.

Quick List

Quick list will list all the previously saves quick entries available to a given user. This includes Quick Transfers, Quick Executes and Quick Plans. Quick Plans cannot be directly created by end-users, but must instead be pre-configured by an administrator.

Quick List

(to access Quick List screen)



Name	Type
answer12	Plan
g100toDTSC	Transfer

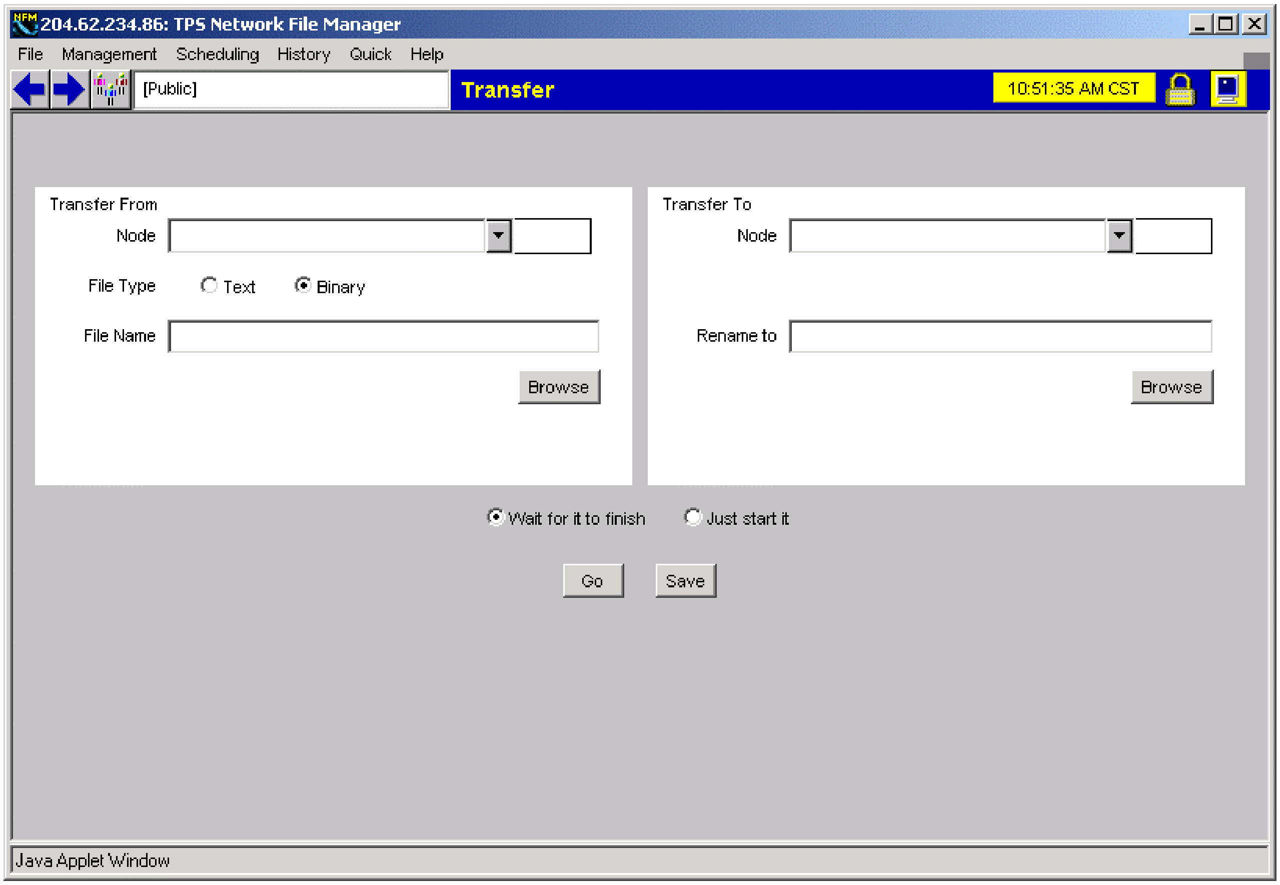
Button Name	Field Description
Go	Begin executing the quick function directly. This will take you to the appropriate screen based on the Quick function type and begin processnig directly.
Edit	Takes you to the appropriate screen based on the Quick function type.

Quick Transfer

Quick Transfer will allow you to immediately perform a single file transfer from a source or target node without the need to configure a plan.

Quick Transfer

(to access Quick Transfer screen)



[Public]

Transfer From
Node
File Type ○ Text ● Binary
File Name

Transfer To
Node
Rename to

● Wait for it to finish ○ Just start it

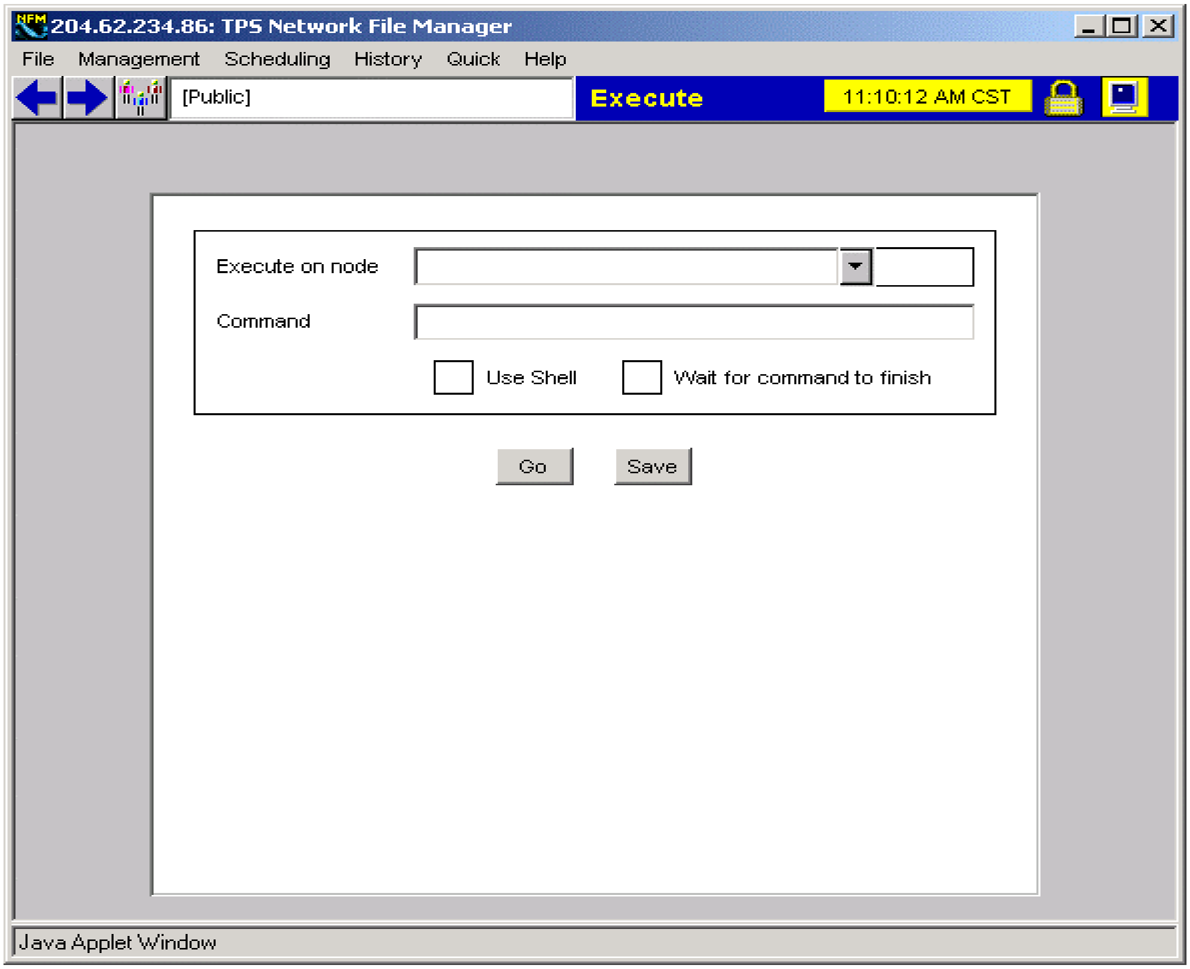
Field Name / Button Name	Field Description
Transfer From Node	Select the desired source node name
Transfer From File	Type in (or browse) the full file name that represents the source file of the transfer. This file cannot be a directory name and it cannot contain wildcards.

Field Name / Button Name	Field Description
File Type	Check the Binary or Text box to describe the mode to transfer this file. Text mode is necessary when transferring a text file between nodes that store text files differently from one another; such as UNIX to Windows
Host File Parameters	This button will appear, if the source node is a Host or mainframe type node. Click on this button to bring up an additional window that allows defining any necessary parameters for referencing a Host type dataset. For the source node, usually just the Dataset Name field is necessary.
Transfer To Node	Select the desired target node name.
Transfer To File	Type in (or browse) the full file name that represents the target file of the transfer. You may select a target directory without specifying the file base name itself. This will cause the base name of the source file to be created in the target directory. If you type in this directory, make certain you include the trailing slash (/) to indicate it is a directory; otherwise, NFM will assume it a target file name and may try to create a file by that name (if you are browsing, the trailing slash is added automatically.
Host File Parameters	This button will appear, if the target node is a Host or mainframe type node. Click on this button to bring up an additional window that allows defining any necessary parameters for creating a Host type dataset.
Wait Option	Select the appropriate button, Wait for it to finish or Just start it to indicate the desired option.
Go	Begin file transfer.
Save	Save the parameters for use at a later time.

Quick Execute

Quick Execute will allow you to immediately perform a single execute of a program on a target node without the need to configure a plan.

(to access Quick Execute screen)



Execute on node

Command

□ Use Shell □ Wait for command to finish

Field Name / Button Name	Field Description
Execute on Node	Select the desired target node name.
Command	Type in the command to run.

Field Name / Button Name	Field Description
Use Shell	This option directs NFM to run the command using the native operating system shell or command line interface. This is necessary for doing things like redirecting output or causing environment variable substitution.
Wait for command to finish	This option will cause the screen to lock until the command is finished running.
	Begin executing the command.
	Save the parameters for use at a later time.

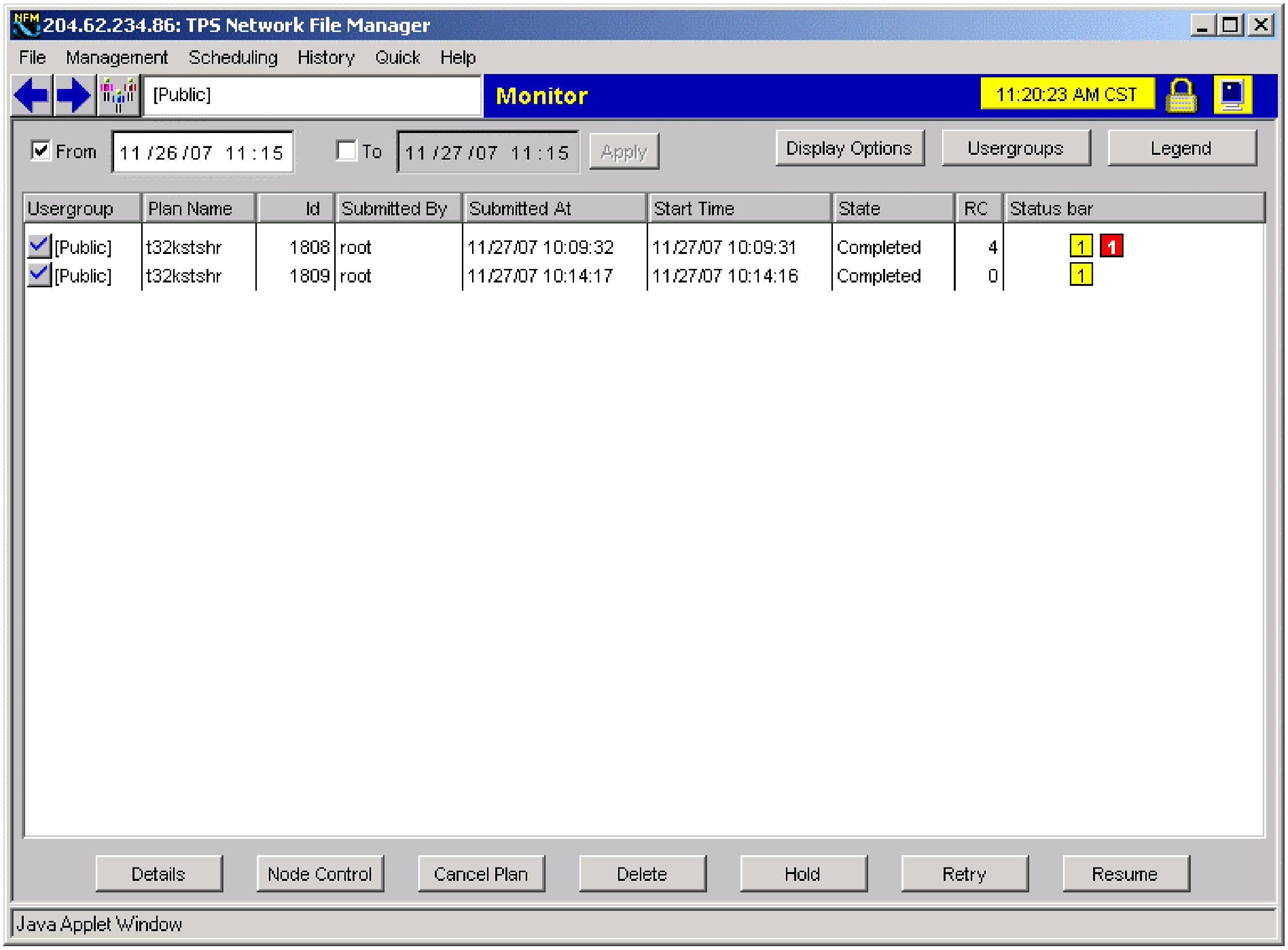
NOTE: Any output that may be generated by the command is not automatically displayed. It can however, be viewed by going to the Quick Monitor screen, clicking on the appropriate instance, and then clicking on the audit message that contains the output. The Quick Monitor screen is described later in this section.

Quick Monitor

Quick Monitor is identical in functionality to the Plan Activity screen. This will display the actual plan information and status that resulted from any of the quick functions performed. Unlike the main Plan activity screen, this list will only include plans that were submitted by the current user. Reference the Plan Activity and Plan Monitor sections for a full description of the information and available options for this screen.

Quick Monitor

(to access Quick Monitor screen)



[Public]

11:20:23 AM CST

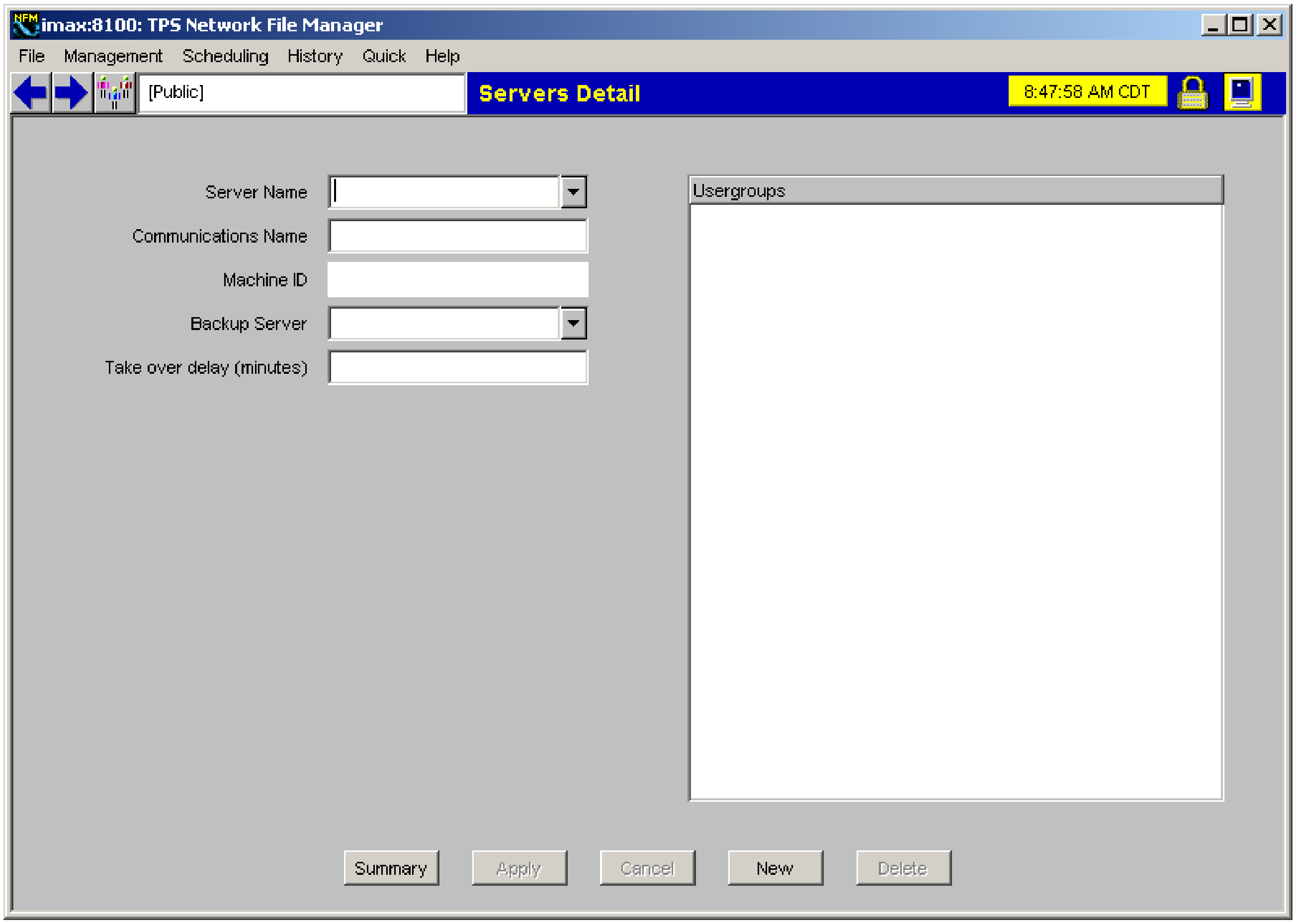
☑ From 11 /26 /07 11 :15 ☐ To 11 /27 /07 11 :15

Usergroup	Plan Name	Id	Submitted By	Submitted At	Start Time	State	RC	Status bar
☑ [Public]	t32kstshr	1808	root	11/27/07 10:09:32	11/27/07 10:09:31	Completed	4	🟨1 🟥1
☑ [Public]	t32kstshr	1809	root	11/27/07 10:14:17	11/27/07 10:14:16	Completed	0	🟨1

Servers

NFM server configuration records are required when running in a multi-server environment, where the servers will be running collectively to form a virtual server running across multiple computers. This is described in detail in section “NFM multiple server support,” “Advanced Options.” Please read that section thoroughly before making any configuration changes here.

When running a single NFM server, or when using multiple NFM servers that are operating independently, it is not necessary to change the configuration using this section.

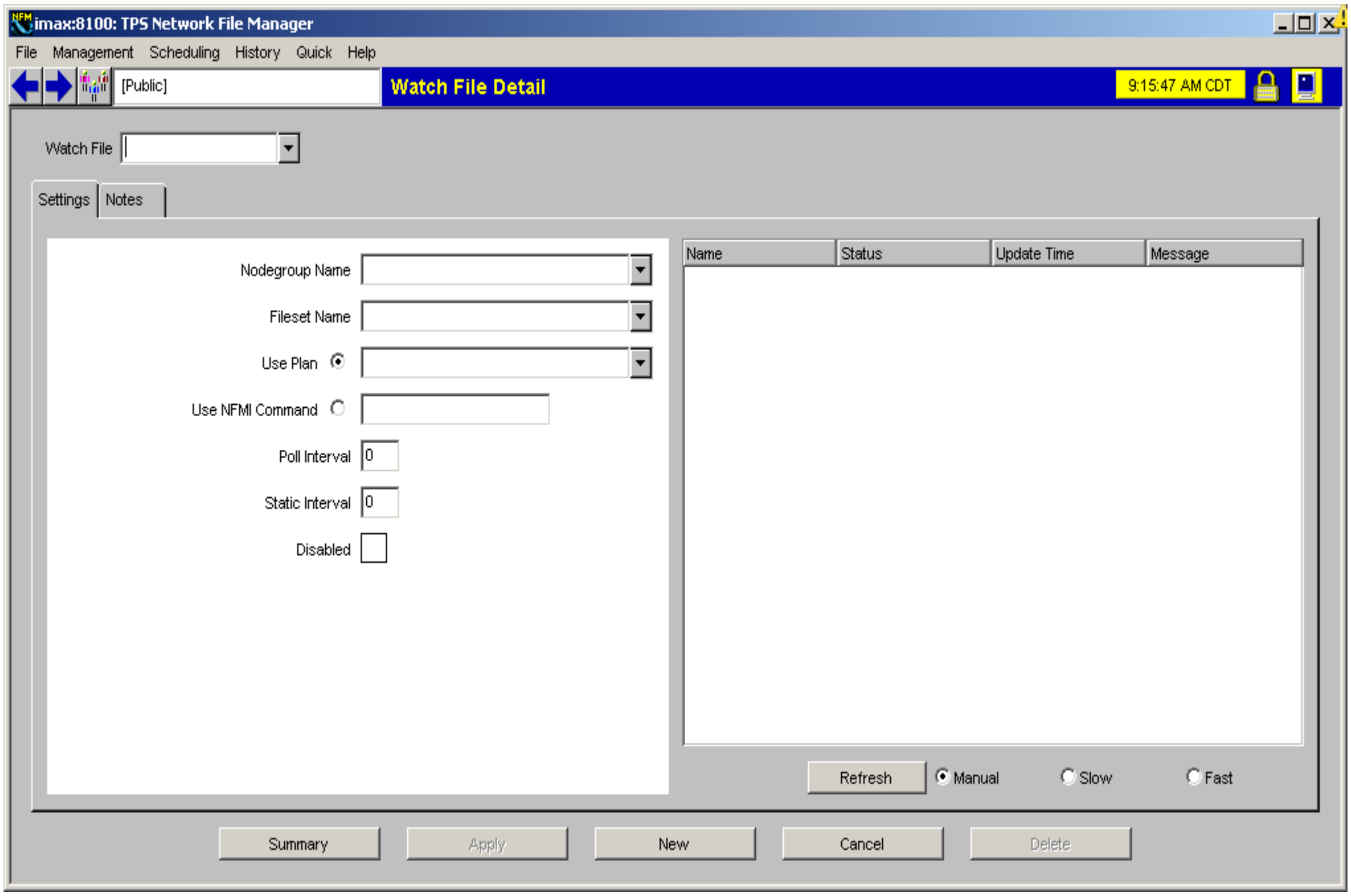


Field Name / Button Name	Field Description
Server Name	This is a user-selected name for an NFM server. Type in a new entry, or select an existing entry from the drop down list. A default entry with the name "SERVER01" should already be present. This entry that represents the current primary server is created automatically at install time.
Communications Name	This field should be set to the IP address or DNS name of the corresponding server. This field is required for other servers to be able to communicate with this server.
Machine ID	This display only field should reflect the machine ID of the corresponding server. This field should be set automatically once the NFM server on the computer has started and detected a proper configuration.
Backup Server	This field designates a backup server to this server definition. In the event that the assigned server fails to start any scheduled plans within a specified time limit, the backup server will automatically do so instead (running the plans on itself). For more information on Backup Servers, please see section "Backup Servers," "Advanced Functions."
Takeover delay (minutes)	This field specifies how long the backup server waits, on a per plan instance basis, before assuming the responsibility of running a plan that the designated server failed to start. The backup server will attempt to factor in the difference, if any, between the clocks on the two computers. Example: - Server A is set to 8:00 p.m. - Server B is set to 8:03 p.m. - Server B is a backup to A with a takeover delay of 5 minutes. - Plans update stores instance 104 is scheduled to run at 8:00 p.m. If Server A never starts the plan, Server B would start the plan at 8:08 p.m. its time (or 8:05 p.m. Server A's time). If Server B cannot fetch the current time from Server A, it will start the plan at its time of 8:05 p.m. (or 8:02 p.m. Server A's time). It is therefore a good idea to keep the clocks in sync, and also, to set the takeover delay long enough to be greater than any skew between the clocks.
Usergroups	This display panel shows all of the usergroups that currently have the field Home Server set to this server record name. (See section "Usergroups")

Watch Files

The NFM watch file feature combines a file discovery mechanism with automatic plan invocation to provide the ability to transfer files automatically when they are placed in a predetermined location on the source node. The transfers themselves rely on the conventional running of a plan. This allows the transfer process to be customized to as desired by the customer.

This feature eliminates the need for the customer to repeatedly run plans to "poll" for data which can create large amounts of plan activity (audits, etc.) when there may be nothing going on. It also provides the ability to transfer files very quickly once they are available.



[Public]

Watch File

Nodegroup Name

Fileset Name

Use Plan ●

Use NFMI Command ○

Poll Interval 0

Static Interval 0

Disabled □

● Manual ○ Slow ○ Fast

Configuration

The following steps are required to implement this new feature:

1. Create the fileset containing the file names or wildcard expressions to "watch" for on the source node or nodes. Any files matching any entries within the fileset will trigger the watch file action. Since the transfer will attempt to copy all files specified in the fileset, a wildcard expression is generally used. Refer to "Configuration," section "FILESETS" for specific information related to the creation of a fileset.

2. Create a node group that contains all of the desired source nodes on which file watching will operate. This must be done even if only one node will be used as the source.

3. Create a plan to copy the fileset from the source node to the desired target node. It is recommended that the plan be configured to run directly when the file trigger occurs. An alternative to running the plan directly is to configure an NFM remote server interface command to be invoked instead. This alternative approach is discussed at the end of this section. The plan should have the following characteristics

- At a minimum the plan should include a function that transfers the files from the source node (where the file trigger occurred) to a target node. Although a watch file configuration may be used to watch any number of source nodes, via the node group, it will run a separate instance of the specified plan as needed for each node.
-
- It is not necessary to put a specific node name in the source node field for the transfer function. You may instead use the special designation `$CTLNODE.` This value if found, is automatically substituted with the actual node name of the source node that triggered this plan instance. This allows the creation of a generic plan that can be used for any number of nodes.
-
- It is recommended that the plan transfer function make use of the same fileset that is specified in the watch file configuration, however this is not required. Another fileset could be used if, for example, additional files need to be sent during the transfer, that are not included in the trigger file list.
-
- It is very important to delete the source files after transferring them. This is best accomplished by checking the field "Delete source file after completion" in the plan function. If the trigger files are not deleted after transferring them, the trigger will continually be reset, which will cause the plan to run repeatedly.

The plan that is triggered by the appearance of a file is not required to transfer the file; this is just a likely scenario. The trigger mechanism can be used to invoke any plan. It is however important to make sure that the plan removes the trigger file at some point before its completion.

Refer to: "Configuration," section "PLANS" for more information on creating a plan.

4. Create the watch file configuration (described in detail in the following section). Once created, the watch file mechanism will automatically activate (unless the disabled checkbox is set). All of the source nodes will be contacted and configured to begin watching for the files. It is therefore recommended that the initial node group contain a single test node to make certain it is working as expected. Additional nodes may later be added to the node group.

Field Name / Button Name	Field Description
Watch File	This is a user-selected name for a watch file configuration. Type in a new entry, or select an existing entry from the drop down list.
Fileset Name	This field should be set to fileset whose entries will be "watched" for on the specified nodes. The fileset may contain fixed names or wildcard expressions.
Use Plan	Check this field and select a plan to invoke when a file trigger occurs on one of the designated nodes.
Use nfmi Command	Alternatively to using a plan, check this field and type in an nfmi command to use instead when a file trigger occurs. See an additional description of this option later in this section.
Poll Interval & Static Interval	These fields specify the discovery checking (or polling) interval and static interval. Both fields are in seconds and default to 10. See the section below for a detailed description of these fields.

More on Watch File processing

Once a watch file configuration is properly defined it will send the appropriate configuration, including the list of files, to each of the designated nodes. The NFM client on each node will automatically scan for files matching the list every # seconds (where # is the polling interval). Once the NFM client determines that there are any matching files, it waits for an additional # seconds (the static interval) and rechecks to see if the list of matching files has changed. If it determines that anything has changed, it will continue waiting for the static interval. The file trigger will not occur until the list of matching files remains unchanged for the full amount of the static interval. This helps prevent a transfer from occurring while the files are still actively being changed.

The list of files will be considered changed if any of the following conditions is detected:

- There are new matching files that were not present during the last check.
- Files that were detected previously are no longer present.
- The size of a previously detected file has changed.
- The time stamp of a previously detected file has changed.

If it is possible to control the manner in which the files are created, certain procedures will also help prevent the premature transfer of a file:

- Have the file created and written to under a different name that does not match entries on the trigger list. Once the file is complete, rename it to its final name.
- Similarly, have the file created in a different directory on the same device and then copied to its final location for pick up.
- Set the trigger to a secondary, dummy file name. Create the dummy file after the actual file or files are finished. Remove the dummy file as part of the transfer processing.

Monitoring Watch File activity

Since plans are used to perform the transfers, the plan instances will automatically appear in the "Plan Activity Screen" as they become active.

If an error in the configuration or other condition prevents this from happening, audit messages will be created. These are viewable from the "Audit Trail" (or History) screen. Watch for these to appear if expected plan instances do not get created. Keep in mind that with the default interval values, it may take more than 20 seconds for a transfer to be attempted after a matching file is placed in the source directory.

Using an NFMI command instead of a plan

As specified in the watch file configuration, an NFMI command may be invoked instead of a plan when a file trigger occurs. This command will most likely perform an `nfm callplan` to initiate a file transfer of the fileset, but it may also take full advantage of the NFM command line interface, which may allow some options to be used that might not otherwise be available from just running a particular plan.

To us this option, you must first configure the remote command. See Chapter 8, "TOOLS," section "NFM remote server interface" for a full description of setting up a remote command. Build a command as if it were being issued directly from one of the watch file nodes. The command will be issued automatically, on behalf of this node, when the file trigger occurs.

Configuring watch files at the end node

Everything in this section so far describes configuring the watch file feature on the NFM server using the user interface (GUI). The necessary information is automatically sent to the NFM clients on the appropriate nodes and stored locally in text configuration files.

It is also possible to set the configuration on an individual node directly on the NFM client computer, which basically involves editing the appropriate text files and then restarting the NFM client to begin watch file processing. This might be necessary under the following conditions:

> The node has a `DEMAND` type connection, and the `CHECKIN` field from the client configuration file is set to N. This means that the client node cannot be contacted directly from the NFM server and the node is not configured to check in to the NFM server at startup. (The `CHECKIN` option being enabled would allow the automatic download of a server configured

watch file configuration). This sort of setup might be deployed, if it is desirable that the NFM client never tries to contact the NFM server, until such time that a watch file condition is triggered. For more information on the client configuration file options, see Chapter 6, “Advanced Functions” section “Encryption and connection options.”

The text file `watchfiles.cfg` in the NFM client home directory contains each of the watch file configurations, active for this node, in a separate stanza that contains basically the same data combined from a server watch file record along with the file names specified in the fileset definition. Create a locally defined entry by copying a stanza from an automated entry and making the following changes:

Assign a new name for the first field `WATCHFILES=`*`Name`*`.` The name must not match an existing watch file record that might later be assigned to this node or the stanza may be over-written by the server one of the same name.

Set the next field to `AUTOENTRY=N.` This marks the stanza as being locally defined and it will not be affected by any updates from the server. Any stanzas that do not have this setting will be subject to automatic deletion if updated configurations are sent down from the NFM server.

NFM GUI

OPERATIONS

This section covers scheduling a plan to run, monitoring and controlling plans. This includes the following GUI operations and screens:

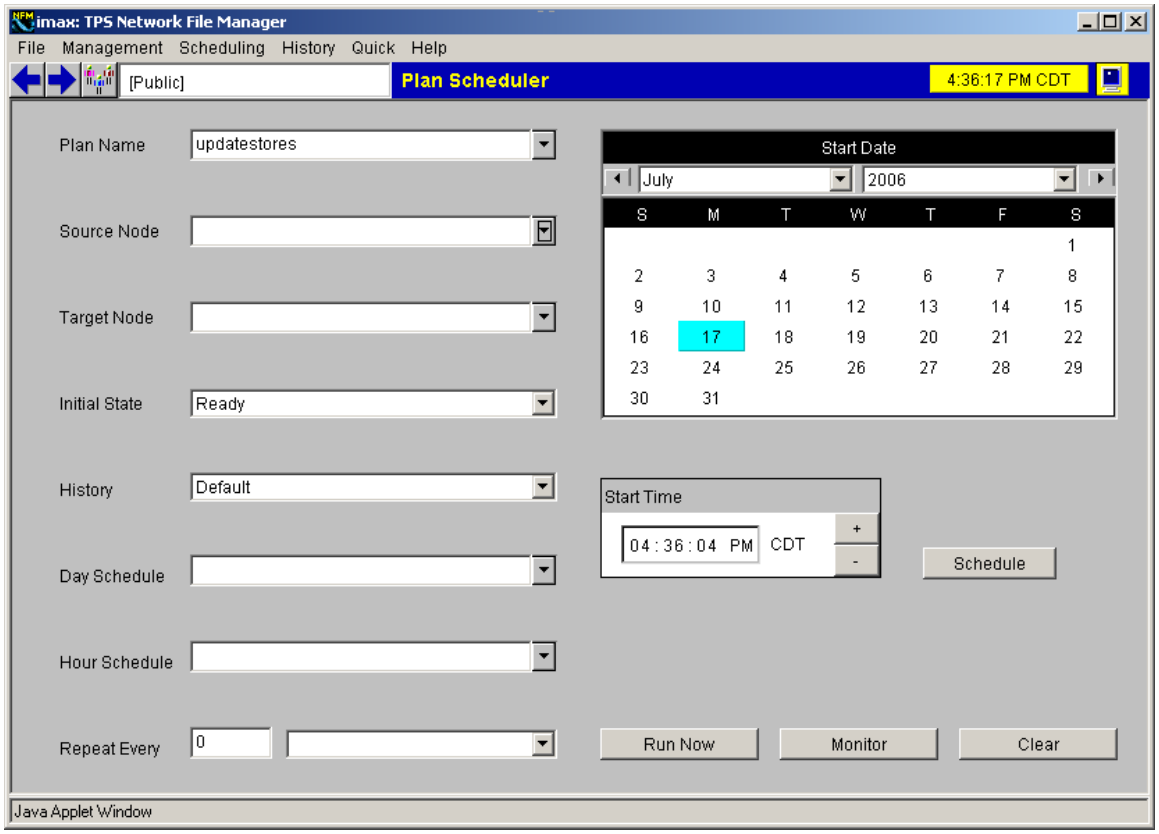
Plan Scheduler
Plan Monitor
Plan Activity
Day Scheduler
Hour Scheduler
Audit Trail

Plan Scheduler

This screen will allow you to schedule plans to run. After entering the appropriate information, click on Schedule button to schedule a plan or Run Now to run the plan immediately. Once a plan is scheduled to run, it will be assigned a unique plan instance ID.

Scheduling Plan Scheduler

(to access Plan Scheduler screen)



Plan Name: updatestores

Source Node:

Target Node:

Initial State: Ready

History: Default

Day Schedule:

Hour Schedule:

Repeat Every: 0

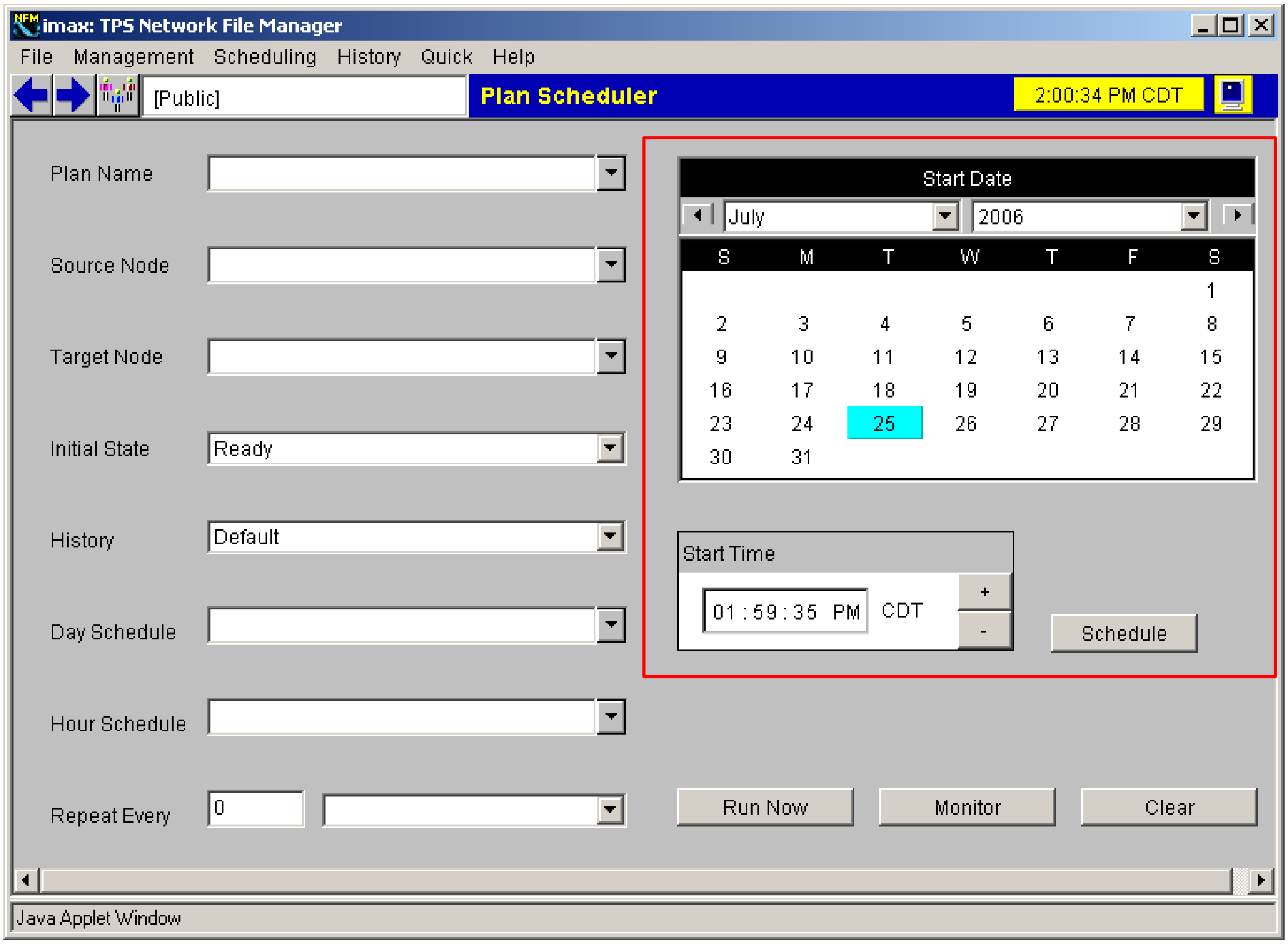
Start Date: July 2006

S	M	T	W	T	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Start Time: 04 : 36 : 04 PM CDT

Field Name / Button Name	Field Description
Plan Name	The desired plan name to run (Required)
Source Node	Specifies the default source node or node group. If there is no source specified in the Plan header, or the Plan function, this value will be used.
Target Node	Specifies the default target node or node group. If there is no target specified in the Plan header, or the Plan function, this value will be used.

Field Name / Button name	Field Description
Initial State	This sets the initial state of the plan. The default value of Ready is usually used, but the plan may be submitted in the Hold state meaning a user must manually resume it.
History	Override the corresponding History field in the Plan header. By specifying Default the Plan header history value is used. (See "Plans" for a description of the History)
Day Scheduler	This specifies a customized calendar that is used in conjunction with automatically rescheduling the plan at predefined intervals (see Day Schedules).
Hour Scheduler	This specifies a customized hour table that is used in conjunction with automatically rescheduling the plan at predefined intervals (see Hour Schedules).
Repeat Every	These two fields instruct the scheduler to automatically reschedule this plan on a given interval. The interval can range from seconds to years. Once the plan is submitted, it will only show once in the submitted plan list. Each time the plan is finished running it will resubmit itself on the appropriate time interval. When a more elaborate schedule is needed, day and hour tables may also be specified.
Schedule	Schedule the plan for execution
Run Now	Run the plan immediately. This will ignore the calendar and start time fields.
Monitor	If you have submitted a plan click on this button to take you directly to the Plan Monitor. In order to view a different plan instance, go to the Plan Activity screen and double click on an entry from the list.



[Public]

Plan Scheduler

2:00:34 PM CDT

Plan Name:

Source Node:

Target Node:

Initial State: Ready

History: Default

Day Schedule:

Hour Schedule:

Repeat Every: 0

Start Date: July 2006

S	M	T	W	T	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

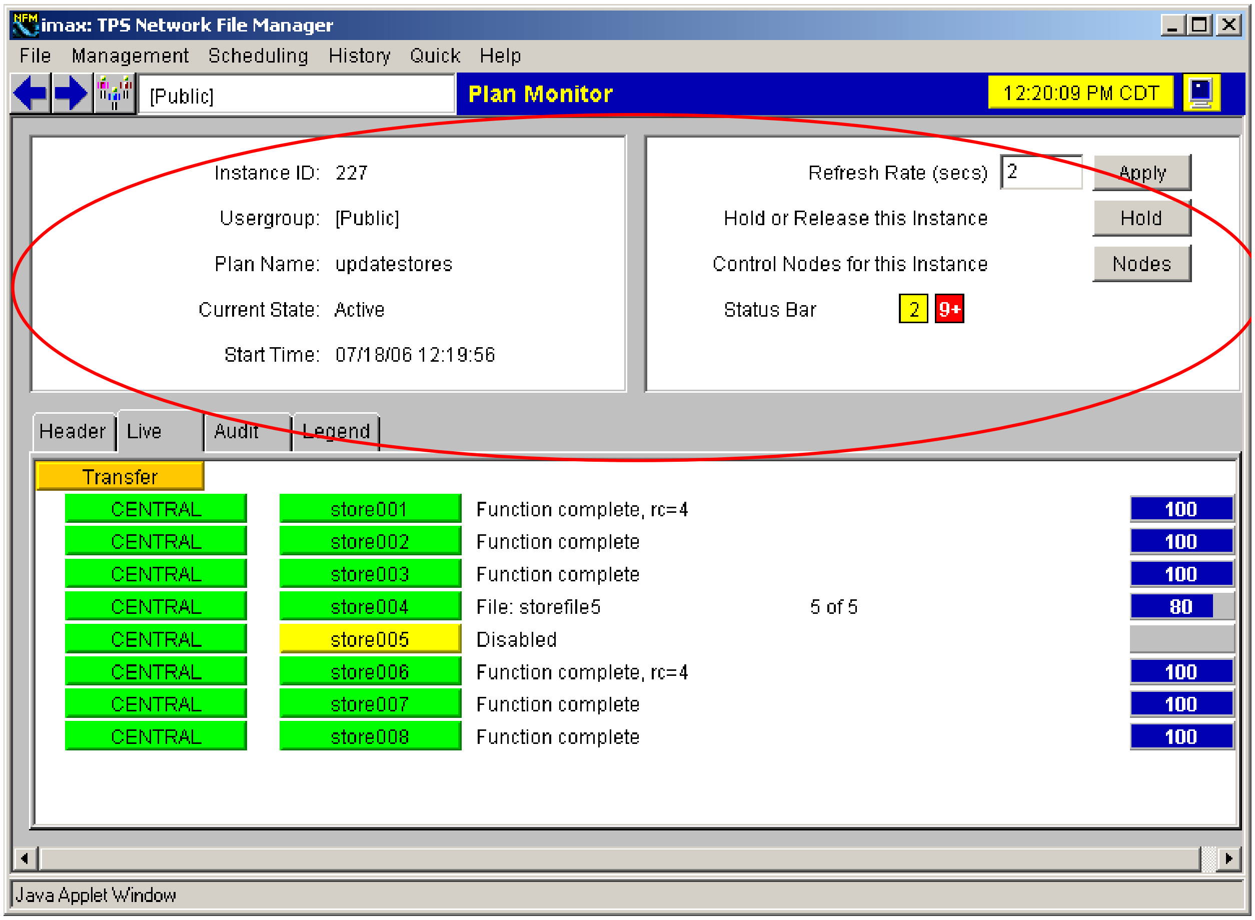
Start Time: 01 : 59 : 35 PM CDT

Use the calendar and Start time boxes to indicate the initial scheduled time for the plan to run. If a plan is setup to restart automatically on a given interval, this interval will be based on the initial start time. Once the plan is submitted a unique numerical plan instance ID is assigned to the plan. Any automatic resubmission of the plan will also generate a unique plan ID.

Plan Monitor

The Plan Monitor screen is not directly accessible from the pull down menu but is reached in one of two ways: Once a plan is submitted from the Plan Scheduler screen, the user may click on the Monitor button to go directly to this screen and display that plan instance. Otherwise, any user, with the correct NFM permissions, may go to the Plan Activity screen and double click on an entry.

This screen shows the current status of an individual plan instance. The current state of this plan instance will determine which tab (Header, Live, Audit, Legend) will be selected. The tabs will automatically switch when the plan changes state. The user may however, manually change to any of the available tabs at any time. This is necessary for example; if the user wishes to examine audits that have been generated even while the plan is still active.



Instance ID: 227

Usergroup: [Public]

Plan Name: updatestores

Current State: Active

Start Time: 07/18/06 12:19:56

Refresh Rate (secs) 2

Hold or Release this Instance

Control Nodes for this Instance

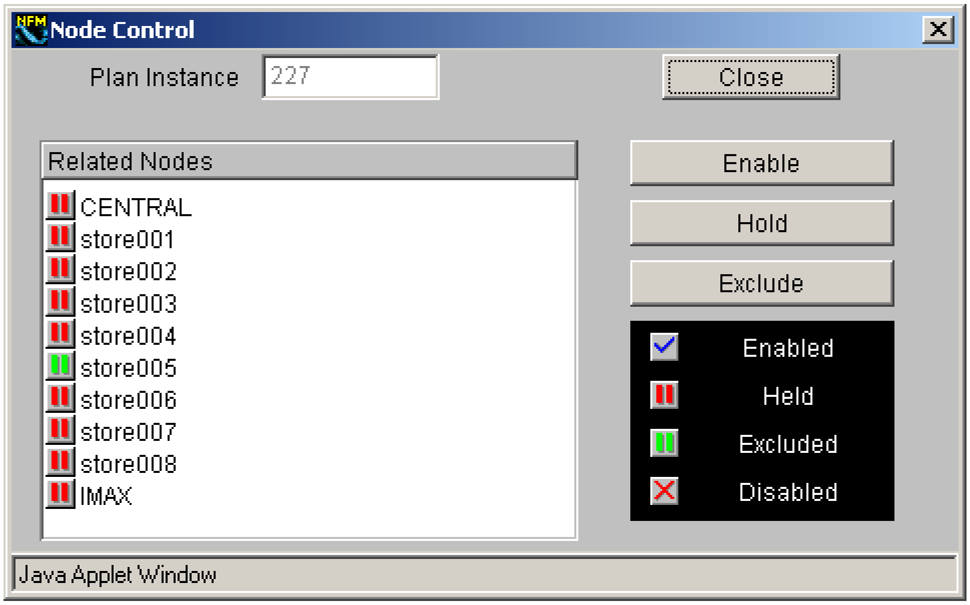
Status Bar 2 9+

Transfer

CENTRAL	store001	Function complete, rc=4		100
CENTRAL	store002	Function complete		100
CENTRAL	store003	Function complete		100
CENTRAL	store004	File: storefile5	5 of 5	80
CENTRAL	store005	Disabled		
CENTRAL	store006	Function complete, rc=4		100
CENTRAL	store007	Function complete		100
CENTRAL	store008	Function complete		100

Field Name / Button Name	Field Description
Instance ID	A unique ID the plan was given when it was scheduled

Field Name / Button Name	Field Description
Usergroup	This is the usergroup that this plan instance belongs to. This should be the same as the plans unsergroup.
Plan Name	The name of the plan
Current State	The state of the plan instance (scheduled, active or finished)
Start Time	When this plan instance was started
Refresh Rate (secs) Apply	How often the screen is refreshed
Hold or Release this Instance Hold	Hold or Release the plan instance
Control Nodes for this Instance Nodes	Select node(s) to enable, disable, hold, or exclude from this plan instance
Status Bar	Displays counters for the following items: Blue box: # of held nodes White box: # of excluded ndes Yellow box: # of warning audits generated Red box: # of error audits generated


Node Control

Plan Instance 227

Related Nodes

🟥 CENTRAL
🟥 store001
🟥 store002
🟥 store003
🟥 store004
🟩 store005
🟥 store006
🟥 store007
🟥 store008
🟥 IMAX

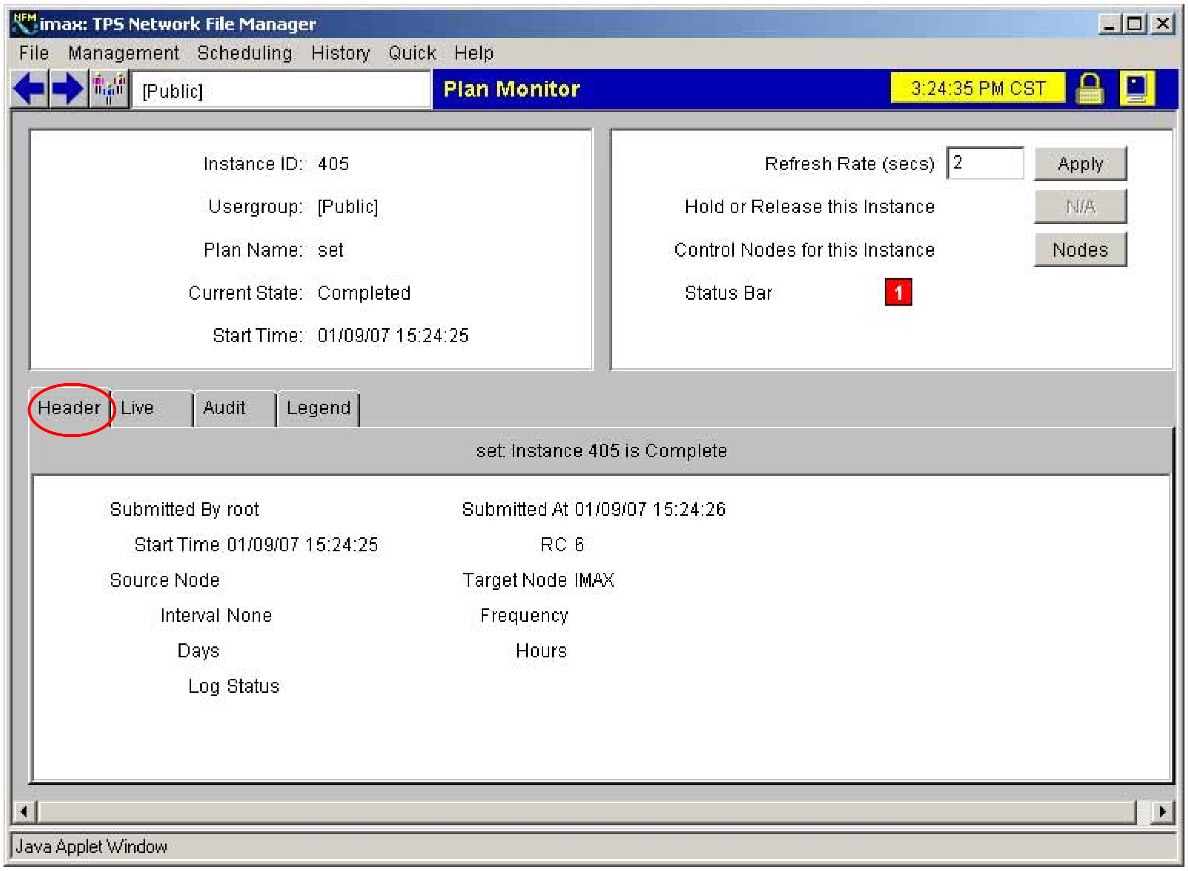
Java Applet Window

(Control Nodes popup screen. The user may select one more items from this list and click on the following buttons to act on the selected entries)

Field / Button Name	Description
Plan Instance	The unique Plan ID assigned to the plan when it was scheduled.
Enable	Resume a held node or it will enable a previously excluded node. Also if the entire plan instance was put on hold the individual node is still enabled.
Hold	Puts a node or group of nodes on hold.
Exclude	Excludes the selected nodes from any further activity in this plan instance. This is similar to the node being defined as off-line in the node record, but it only applies to this particular plan instance.

Header Tab

The Header tab contains a summary of the information that was entered from the Plan Scheduler. If the plan has not started yet, a countdown is displayed.



Instance ID: 405
Usergroup: [Public]
Plan Name: set
Current State: Completed
Start Time: 01/09/07 15:24:25

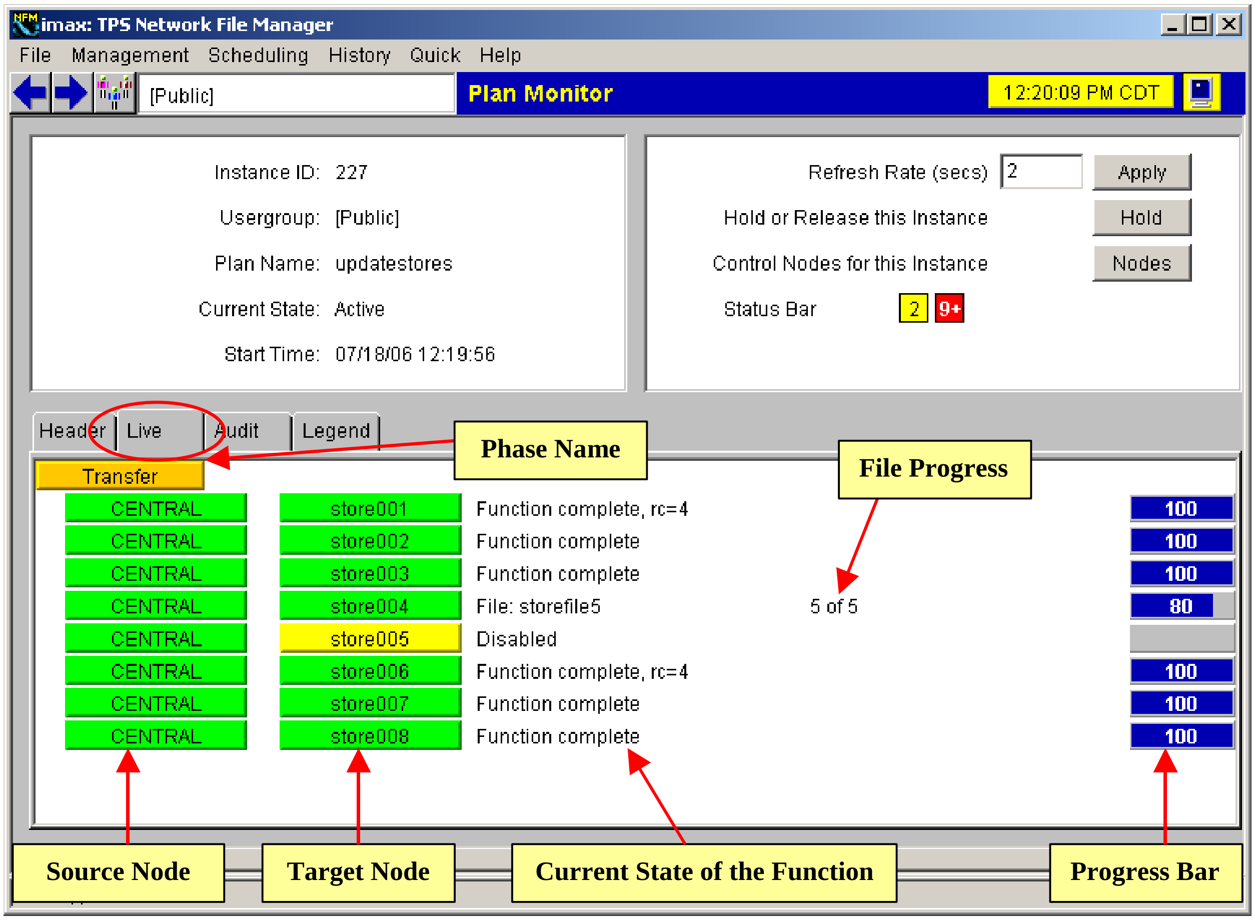
Refresh Rate (secs) 2
Hold or Release this Instance
Control Nodes for this Instance
Status Bar 1

set: Instance 405 is Complete

Submitted By root	Submitted At 01/09/07 15:24:26
Start Time 01/09/07 15:24:25	RC 6
Source Node	Target Node IMAX
Interval None	Frequency
Days	Hours
Log Status	

Live Tab

Live tab shows the progress of the plan when the plan is currently active. The Live tab shows the current Phase and the current state of any function contained within the Phase.



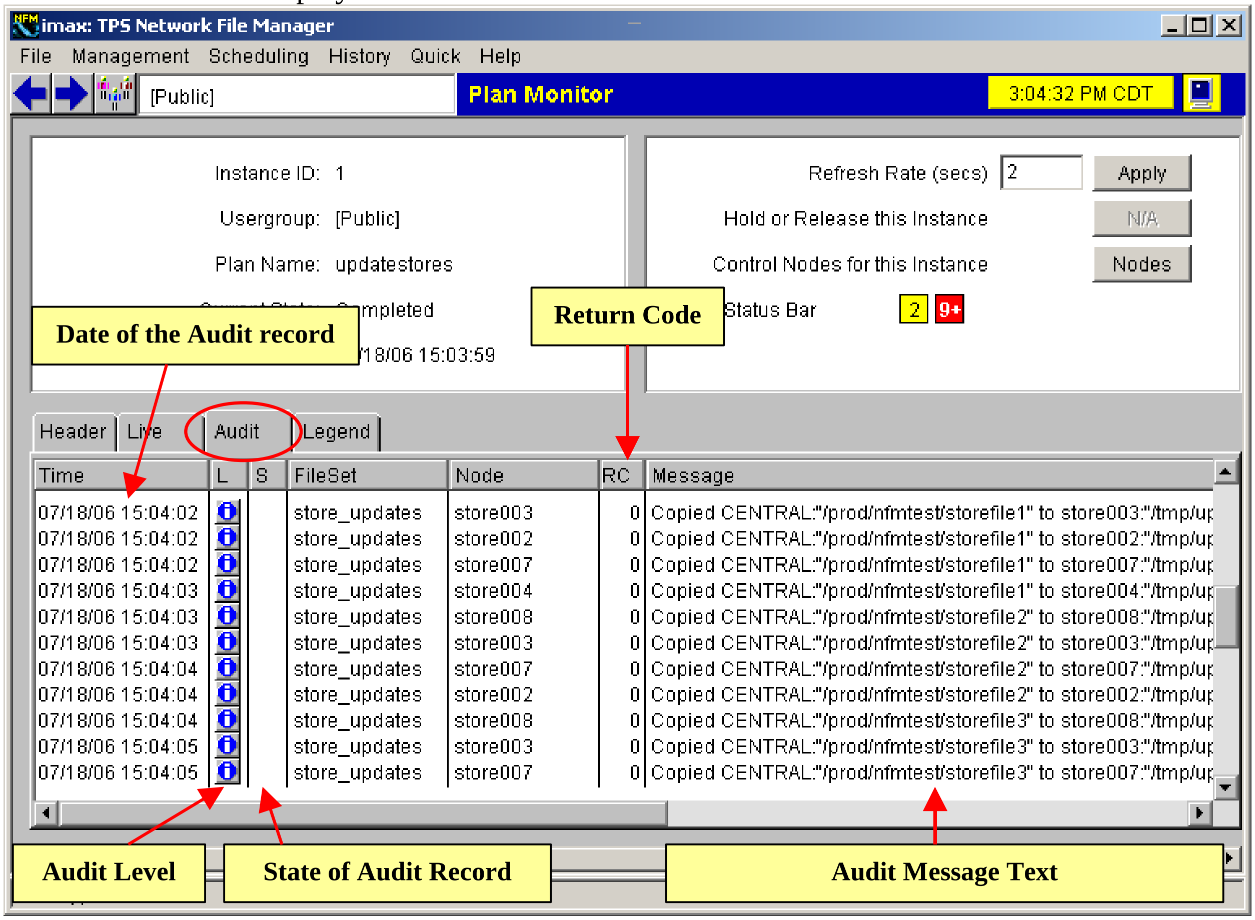
Instance ID: 227
Usergroup: [Public]
Plan Name: updatestores
Current State: Active
Start Time: 07/18/06 12:19:56

Refresh Rate (secs) 2
Hold or Release this Instance
Control Nodes for this Instance
Status Bar 2 9+

CENTRAL	store001	Function complete, rc=4		100
CENTRAL	store002	Function complete		100
CENTRAL	store003	Function complete		100
CENTRAL	store004	File: storefile5	5 of 5	80
CENTRAL	store005	Disabled		
CENTRAL	store006	Function complete, rc=4		100
CENTRAL	store007	Function complete		100
CENTRAL	store008	Function complete		100

Audit Tab

The Audit tab shows all of the audits generated so far by the plan instance. The History level selected in the Plan Detail or in the Plan Schedule will determine the amount of information displayed.



Instance ID: 1

Usergroup: [Public]

Plan Name: updatestores

Current State: Completed

/18/06 15:03:59

Refresh Rate (secs) 2

Hold or Release this Instance

Control Nodes for this Instance

```
Time               L  S  FileSet        Node      RC  Message
07/18/06 15:04:02  ℹ     store_updates  store003   0  Copied CENTRAL:"/prod/nfmtest/storefile1" to store003:"/tmp/up
07/18/06 15:04:02  ℹ     store_updates  store002   0  Copied CENTRAL:"/prod/nfmtest/storefile1" to store002:"/tmp/up
07/18/06 15:04:02  ℹ     store_updates  store007   0  Copied CENTRAL:"/prod/nfmtest/storefile1" to store007:"/tmp/up
07/18/06 15:04:03  ℹ     store_updates  store004   0  Copied CENTRAL:"/prod/nfmtest/storefile1" to store004:"/tmp/up
07/18/06 15:04:03  ℹ     store_updates  store008   0  Copied CENTRAL:"/prod/nfmtest/storefile2" to store008:"/tmp/up
07/18/06 15:04:03  ℹ     store_updates  store003   0  Copied CENTRAL:"/prod/nfmtest/storefile2" to store003:"/tmp/up
07/18/06 15:04:04  ℹ     store_updates  store007   0  Copied CENTRAL:"/prod/nfmtest/storefile2" to store007:"/tmp/up
07/18/06 15:04:04  ℹ     store_updates  store002   0  Copied CENTRAL:"/prod/nfmtest/storefile2" to store002:"/tmp/up
07/18/06 15:04:04  ℹ     store_updates  store008   0  Copied CENTRAL:"/prod/nfmtest/storefile3" to store008:"/tmp/up
07/18/06 15:04:05  ℹ     store_updates  store003   0  Copied CENTRAL:"/prod/nfmtest/storefile3" to store003:"/tmp/up
07/18/06 15:04:05  ℹ     store_updates  store007   0  Copied CENTRAL:"/prod/nfmtest/storefile3" to store007:"/tmp/up
```

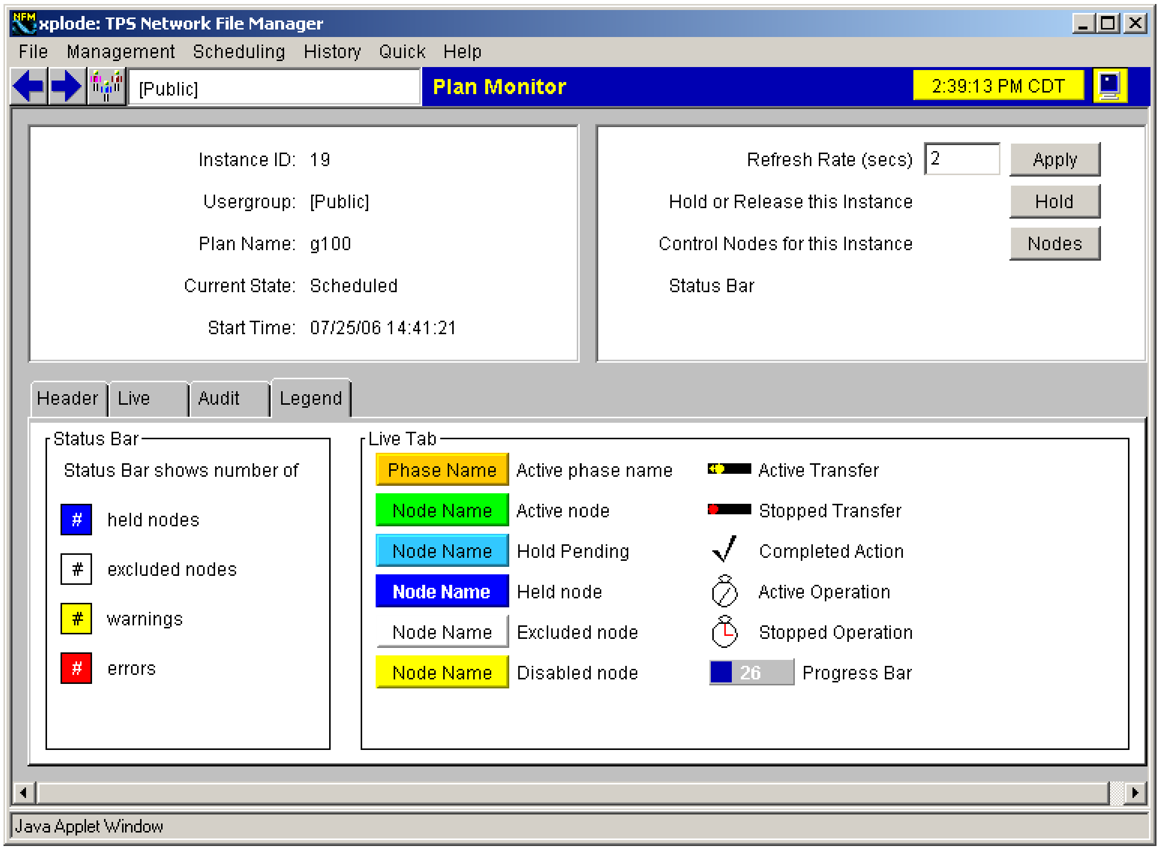
Some fields need to be expanded on:

Field Name	Field Description
Level	Audits will be one of three types: ℹ Informational ⚠ Warning ☒ Error
State	Indicates if the audit message has been read or not (Only for Warning and Errors). ⊘ Not read Read
RC	The highest return code value returned by the function or phase. The higher

Field Name	Field Description
	the return code the more severe the error (see Appendix A: Plan Return Codes).
Message	The message text of the audit record. Depending on the type of audit, this may contain an event description, warning or error message, or command output. Some audits contain multiple lines of text. These messages will appear with a '…' at the end. Double click on these entries to bring a popup window with the complete message.

Legend Tab

The Legend tab is a visual depiction and description for many icons and color codes displayed on the Plan Monitor screen.



Instance ID: 19

Usergroup: [Public]

Plan Name: g100

Current State: Scheduled

Start Time: 07/25/06 14:41:21

Refresh Rate (secs) 2

Hold or Release this Instance

Control Nodes for this Instance

Status Bar

Status Bar

Status Bar shows number of

- # held nodes
- # excluded nodes
- # warnings
- # errors

Live Tab

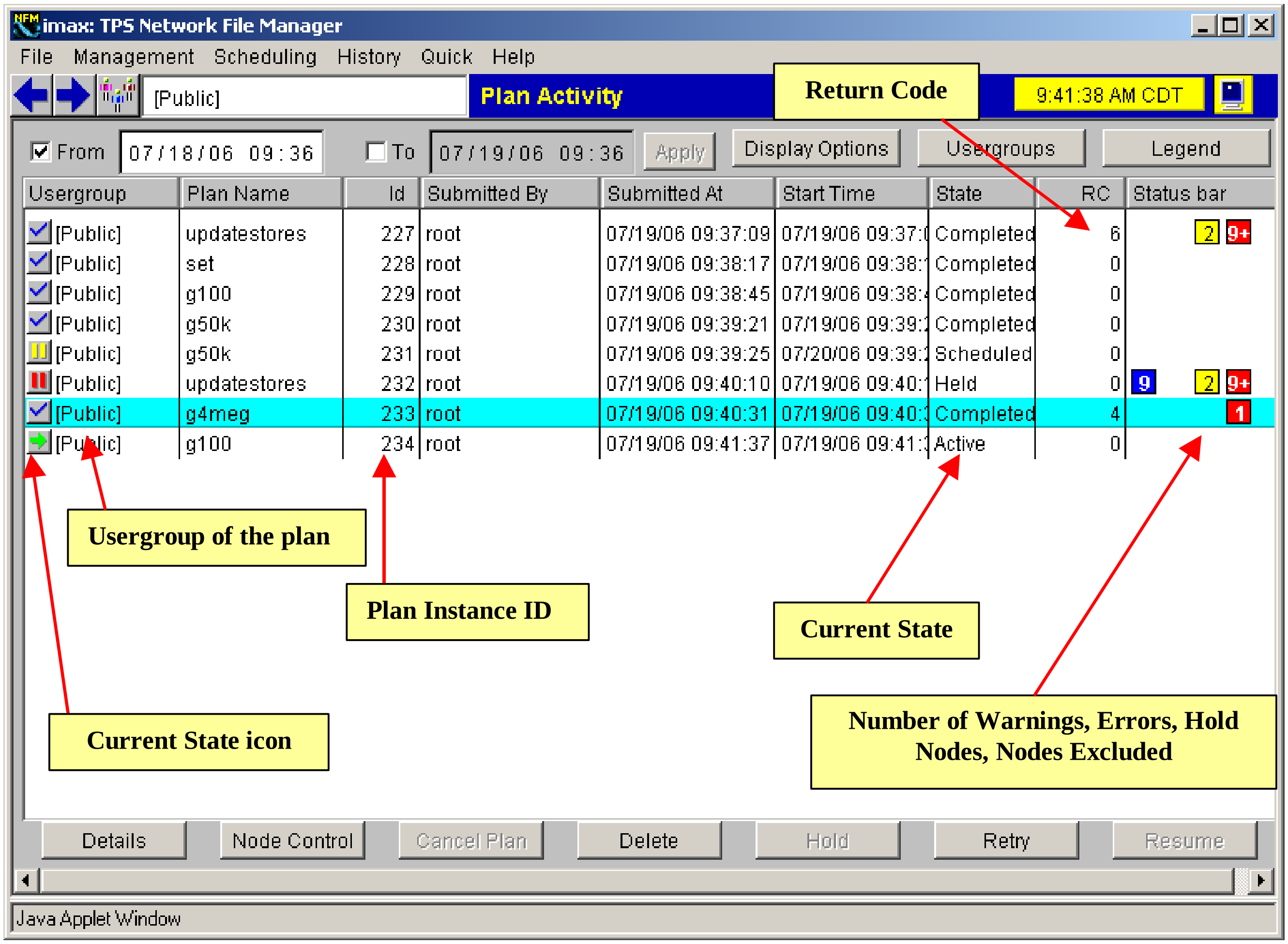
Phase Name	Active phase name		Active Transfer
Node Name	Active node		Stopped Transfer
Node Name	Hold Pending	✓	Completed Action
Node Name	Held node		Active Operation
Node Name	Excluded node		Stopped Operation
Node Name	Disabled node	26	Progress Bar

Plan Activity

Plan Activity is a summary screen for all plan instances. Various information about each plan instance is listed along with its current state and return code (if finished). Double clicking on an entry at any time, will transfer control to the [Plan Monitor]() screen for the selected plan instance. Like the Plan Monitor screen, this screen automatically refreshes itself to display changing plan information.

Scheduling Plan Activity

(to access Plan Activity screen)



Usergroup	Plan Name	Id	Submitted By	Submitted At	Start Time	State	RC	Status bar
✓ [Public]	updatestores	227	root	07/19/06 09:37:09	07/19/06 09:37:	Completed	6	2 9+
✓ [Public]	set	228	root	07/19/06 09:38:17	07/19/06 09:38:	Completed	0	
✓ [Public]	g100	229	root	07/19/06 09:38:45	07/19/06 09:38:	Completed	0	
✓ [Public]	g50k	230	root	07/19/06 09:39:21	07/19/06 09:39:	Completed	0	
‖ [Public]	g50k	231	root	07/19/06 09:39:25	07/20/06 09:39:	Scheduled	0	
‖ [Public]	updatestores	232	root	07/19/06 09:40:10	07/19/06 09:40:	Held	0	9 2 9+
✓ [Public]	g4meg	233	root	07/19/06 09:40:31	07/19/06 09:40:	Completed	4	1
→ [Public]	g100	234	root	07/19/06 09:41:37	07/19/06 09:41:	Active	0	

Return Code

Usergroup of the plan

Plan Instance ID

Current State

Current State icon

Number of Warnings, Errors, Hold Nodes, Nodes Excluded

** Display Options can show / hide columns thus your screen columns might not match this screen shot.

Some fields / buttons that need to be expanded on:

Field / Button name	Field Description
Usergroup	The usergroup that the plan instance belongs to. (This screen will only show you plan instances that A) belong to usergroups that you are a member of and B) are not being filtered out.).
State (icon)	☑ Completed ⏸ Held ⏸ Scheduled ➡ Active ☒ Aborted ? Rejected 🗑 Cancelled ➡ Hold Pending
State (text)	Current state of the plan ID. State can be Completed, Held, Active, Scheduled, Cancelled, or Interrupted
RC	The highest return code value returned by the function or phase. The higher the return code the more severe the error (see Appendix A: Plan Return Codes).
Status Bar	Contains the number of: # Held Nodes # Excluded Nodes # Warnings # Errors
Display Options	Displays a window that allows you to change the fields (columns) displayed on this screen.
Legend	Displays a window explaining the different icons and their meanings.
Usergroups	Displays a window that allows you to change which Usergroups' plan instances are displayed on this screen.
Details	With a plan ID selected, brings up the Plan Monitor screen
Node Control	Brings up the Node Control window allowing you to select node(s) to enable, disable, or exclude from this plan instance
Retry	Restarts the same plan instance from the beginning and attempts to perform only those actions that failed previously. 'Enable Plan Retry'

Field / Button name	Field Description
	checkbox for the plan must be on in order to use this feature.

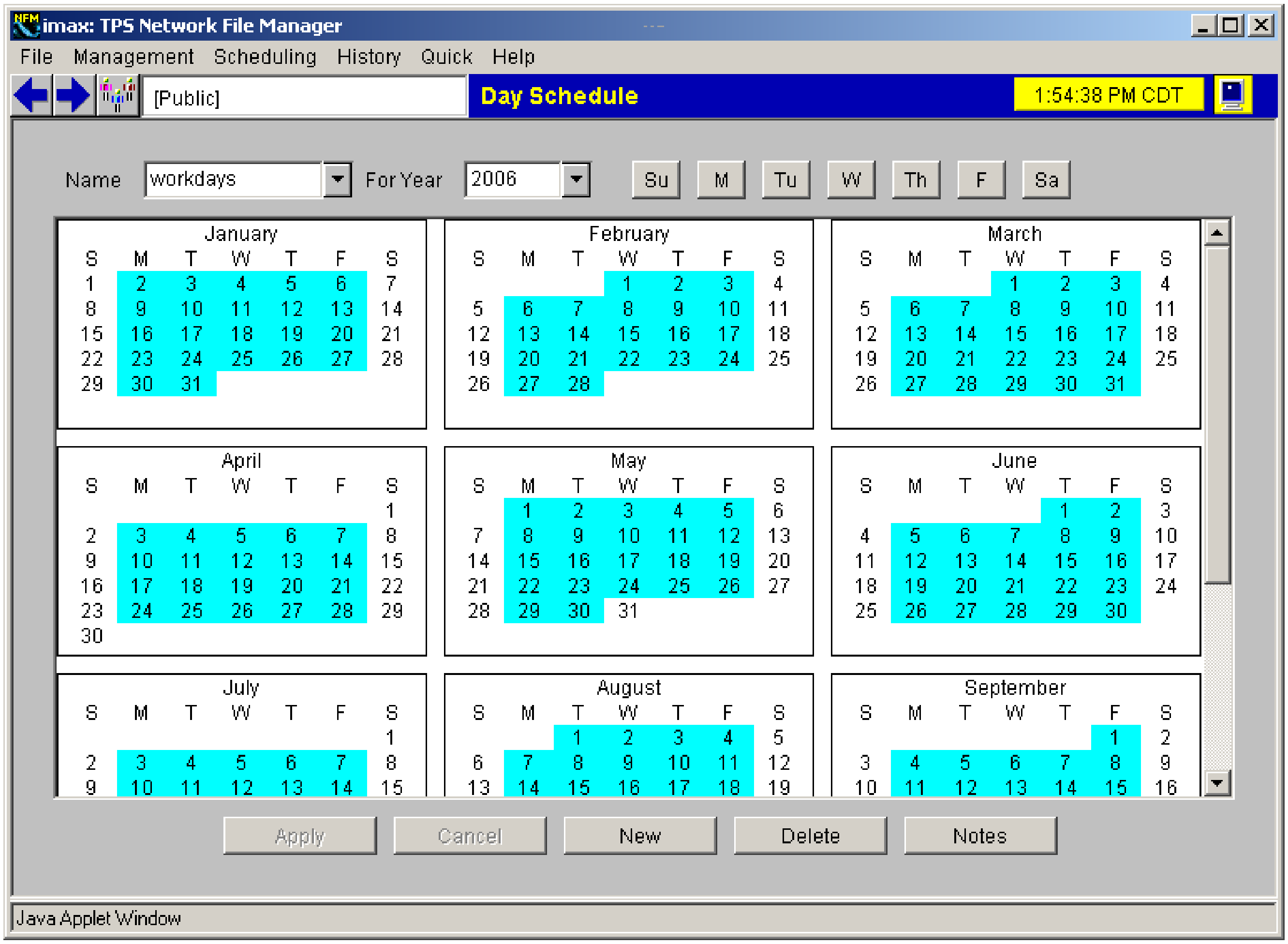
Custom audit messages can be created using the NFM command line interface (`PUTAUDIT` sub-command). This can be called from within a plan, or directly from a script.

Day Schedule

Day Schedule maintains custom calendars that tailor the exact days to run plans that are to be automatically repeated. Once created, the schedule is specified in the Day Schedule field on the Plan Scheduler to associate it with a given plan instance.

Scheduling Day Schedule

(to access the Day Schedule screen)


imax: TPS Network File Manager
File Management Scheduling History Quick Help
[Public]
Day Schedule
1:54:38 PM CDT
Su M Tu W Th F Sa
Apply Cancel New Delete Notes
Java Applet Window

Name workdays For Year 2006

January

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

February

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28				

March

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

April

S	M	T	W	T	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						

May

S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

June

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

July

S	M	T	W	T	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15

August

S	M	T	W	T	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19

September

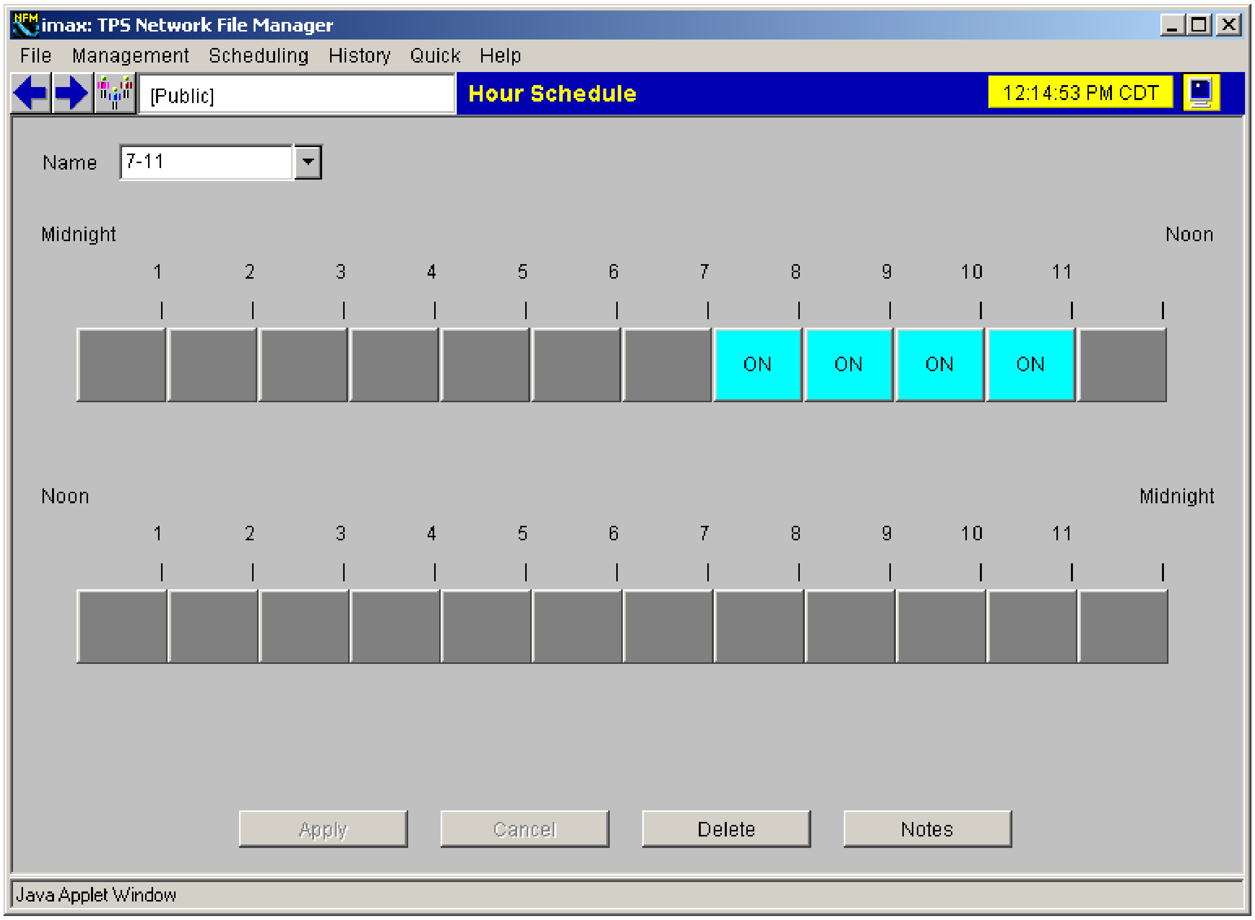
S	M	T	W	T	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16

Hour Schedule

Hour Schedule is similar to the day schedule but for use on an hourly basis instead. Once created, the schedule is specified in the Hour Schedule field on the Plan Scheduler to associate it with a given plan instance.

Scheduling Hour Schedule

(to access the Hour Scheduler screen)



[Public]

12:14:53 PM CDT

Name 7-11

Midnight Noon

1	2	3	4	5	6	7	8	9	10	11	
							ON	ON	ON	ON	

Noon Midnight

1	2	3	4	5	6	7	8	9	10	11	

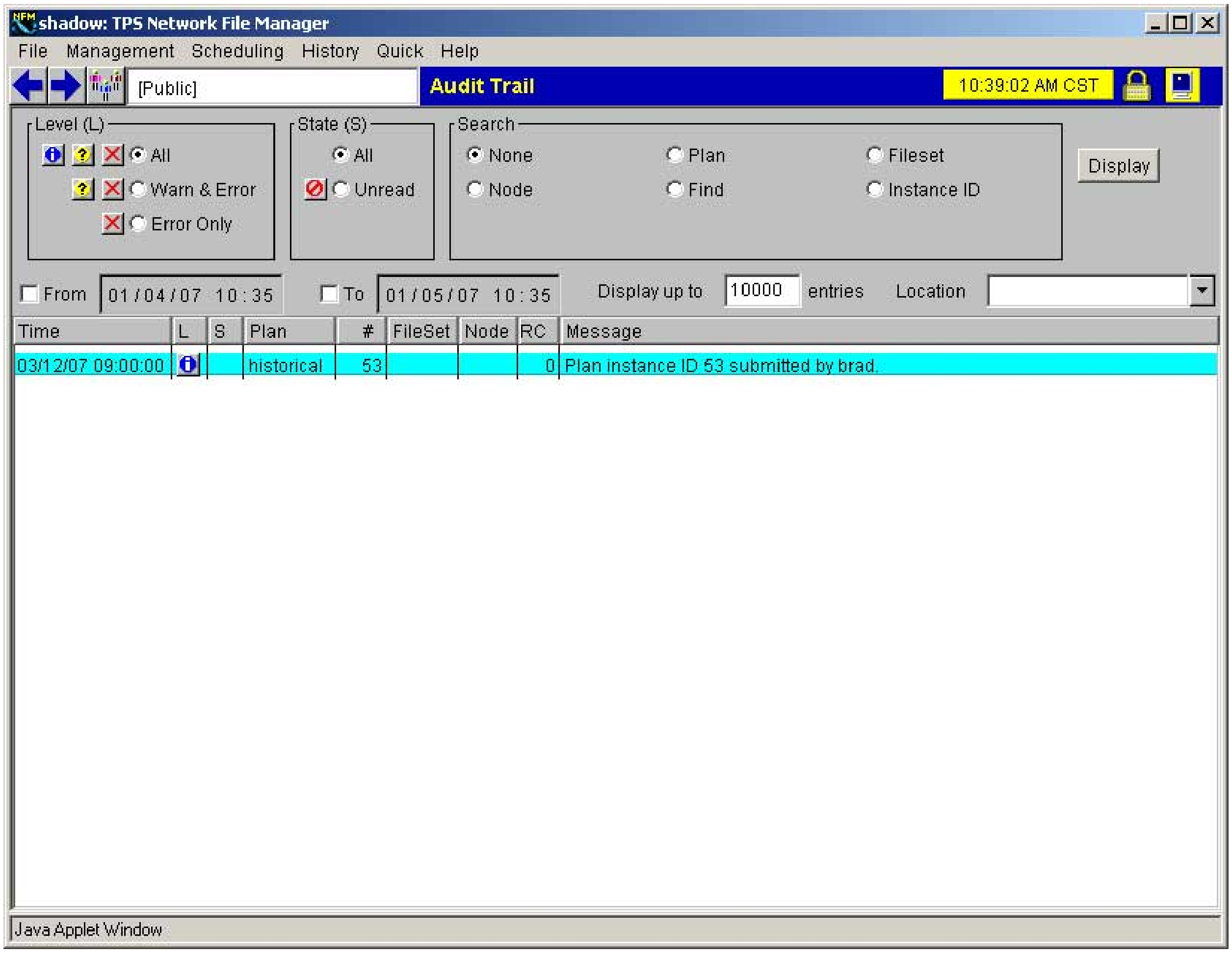
Audit Trail

Audit messages may be created in a variety of ways. The majority of audit messages will be generated from plan activity. The amount of detail that will be included will depend upon the configuration of the history field on a per-plan basis. Each audit message will be one of three types, which includes ☒ errors, ? warnings, and ℹ informational audits. Audits are also generated during configuration changes if this option is enabled in the System Settings screen.

The Audit Trail screen is used to filter and view audit messages. Unlike the Plan Monitor Audit folder which views only a given plan instance's audits, this screen allows viewing all of the audit records, or to display a subset of them based on a particular filtering criteria.

History Audit Trail

(to access the Audit Trail screen)



Level (L): ● All ○ Warn & Error ○ Error Only

State (S): ● All ○ Unread

Search: ● None ○ Plan ○ Fileset ○ Node ○ Find ○ Instance ID

□ From 01/04/07 10:35 □ To 01/05/07 10:35 Display up to 10000 entries Location

Time	L	S	Plan	#	FileSet	Node	RC	Message
03/12/07 09:00:00	ℹ		historical	53			0	Plan instance ID 53 submitted by brad.

Some fields need to expanded on:

Field Name	Field Description
Location	This field allows the user to display audits from an alternate data source other than the current NFM database. This may be an archive location or any directory that contains audit tables copied from the database directory or perhaps from another NFM Server. Alternate data sources may be added with the NFM command line interface (reference the sub-command `putasource`.
Level (L)	Audits will be one of three types: ⓘ Informational ? Warning ☒ Error
State (S)	If the audit message has been read or not (Only for Warning and Errors). ⊘ Not read Read
RC	The highest return code value returned by the function or phase. The higher the return the more severe the error (see Appendix A: Plan Return Codes).
Message	The message text of the audit record. Depending on the type of audit, this may contain an event description, warning or error message, or command output. Some audits contain multiple lines of text. These messages will appear with a '…' at the end. Double click on these entries to bring a popup window with the complete message.

TPS®/NFM Training Guide

Section 2

ADVANCED OPTIONS

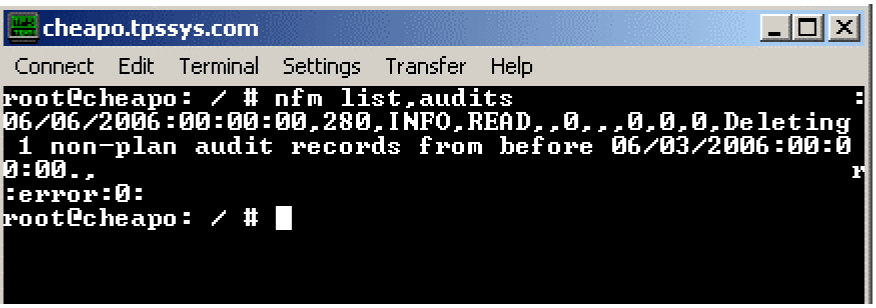
Local Command Line Interface

A flash demo is available online at http://www.tps.com/nfmdemo/LCI.htm

NFM Local Command Line Interface

What is the Local Command Line Interface?

The Local Command Line Interface provides access to a variety of nfm commands through the NFM server's operating system's command prompt. This feature can be helpful when you want a third party program or script to interact with NFM *(For example: you want another program to schedule an NFM plan)*. The Local Command Line Interface allows the user to accomplish almost any task that can be performed through the NFM GUI interface.



```
root@cheapo: / # nfm list,audits                      :
06/06/2006:00:00:00,280,INFO,READ,,0,,,0,0,0,Deleting
 1 non-plan audit records from before 06/03/2006:00:0
0:00.,                                                r
:error:0:
root@cheapo: / #
```

Configuring the NFM Local Command Line Interface

Where can I find the NFM Local Command Line Interface?

The NFM Local Command Line Interface is installed as part of the NFM Server product and are accessible with the use of the 'nfm' command line executable located in the following directories:

Operating System	Directory
Windows	C:\Program Files\TPS Systems\NFMServer
AIX	/usr/lpp/tps/nfm/bin
UNIX	/opt/tps/nfm/bin

Before we are able to use the 'nfm' commands, we need to log into the NFM GUI and create a new user account that will use the NFM Local Command Line Interface.

Configuring the NFM Local Command Line Interface

What else do I need to configure?

Now that we created a user who will use the NFM Local Command Line Interface, you will need to setup three environment variables on the NFM Server:

```
NFMUSER=steve
NFMPASSWORD=something
NFMUGID=<ID of your Usergroup> **
```

These environment variables will be used to authenticate your NFM permissions each time you run 'nfm' from the command prompt. It is recommended that you store these environment variables in the user's .profile script (UNIX) or in System Environment Variables (Windows).

** Optionally NFMUGID is used to change the current usergroup. For companies that do not utilize Usergroups, this environment variable is not required.

Using the NFM Local Command Line Interface

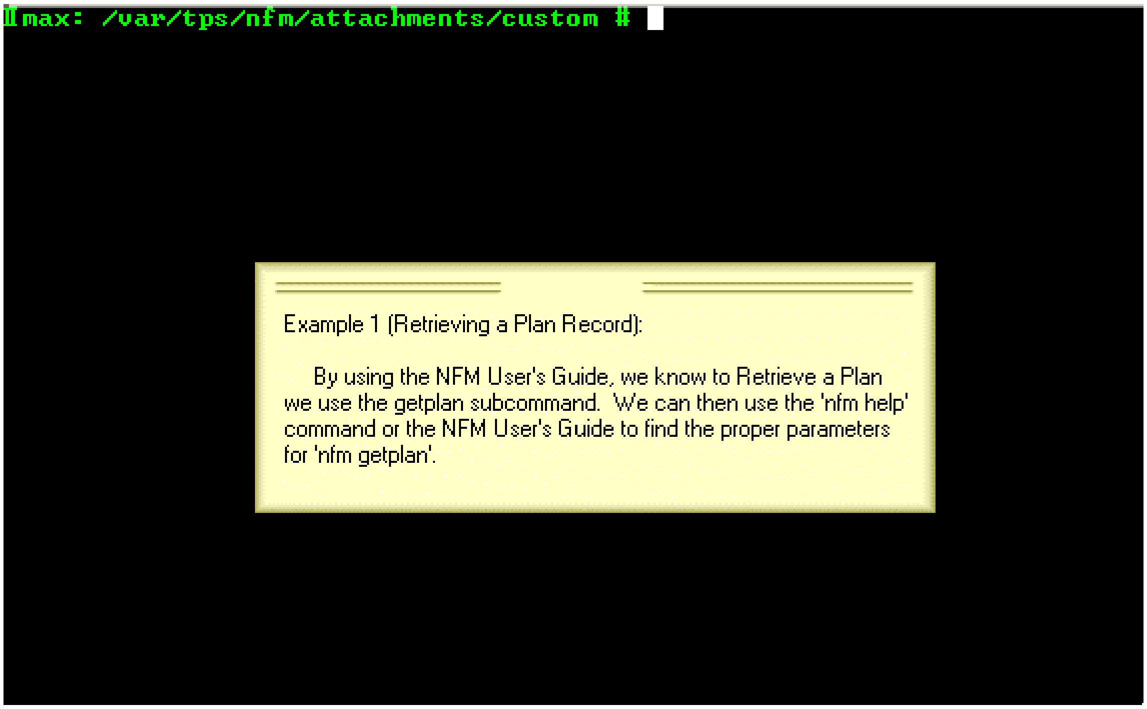
How do I Use the NFM Command Line Interface?

With the NFM Local Command Line Interface configured, you can now start using the '*nfm*' executable from the NFM Server's command line. Here is a short list of NFM commands available from the NFM Local Command Line Interface:

canceljob, deletefileset, deletejob, deletemodel, deletenode, deletenodegroup, deleteplan, deleteuser, getfileset, getjobstatus, getmodel, getnode, getplan, holdjob, listclients, listfilesets, listjobs, listjobsx, listmodels, listnodes, listnodegroups, listnodemembers, listplans, listusers, putaudit, putfileset, putmodel, putnode, putplan resumejob, retryjob, statnode, submitplan

The TPS/NFM User's Guide describes detailed information regarding each command and all NFM commands and syntax. You can also use the '*nfm help*' command to display a full list of commands.

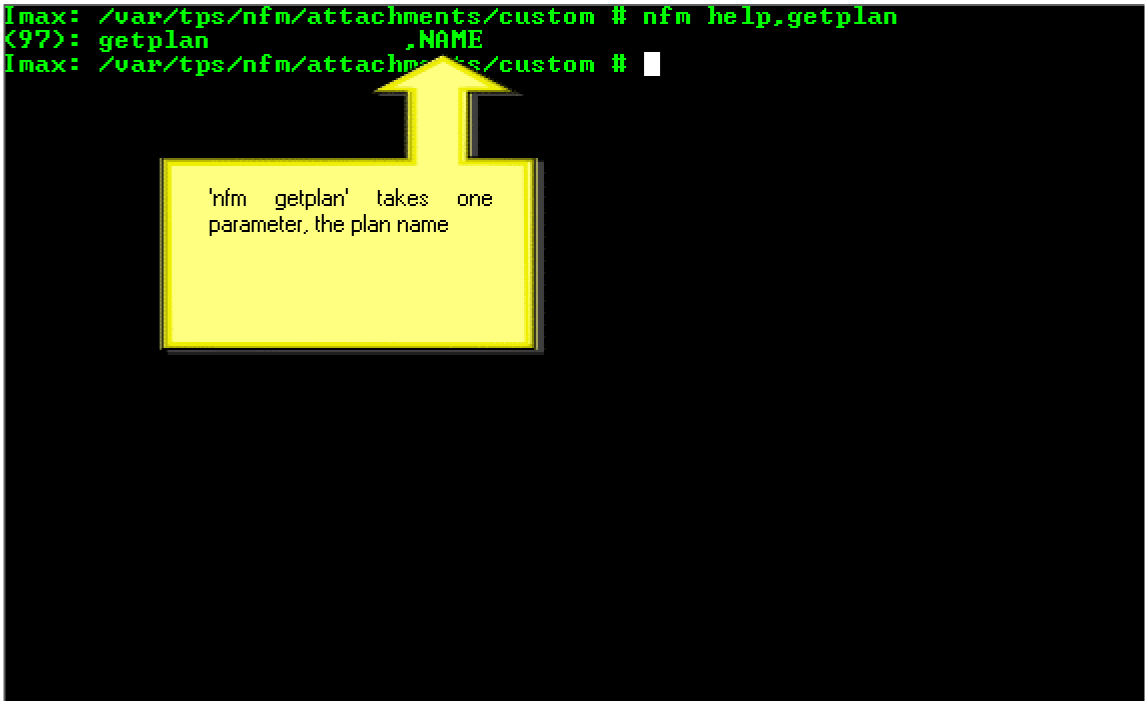
In the meantime, we will go over several examples and cover many of the common commands.



```
Imax: /var/tps/nfm/attachments/custom #
```

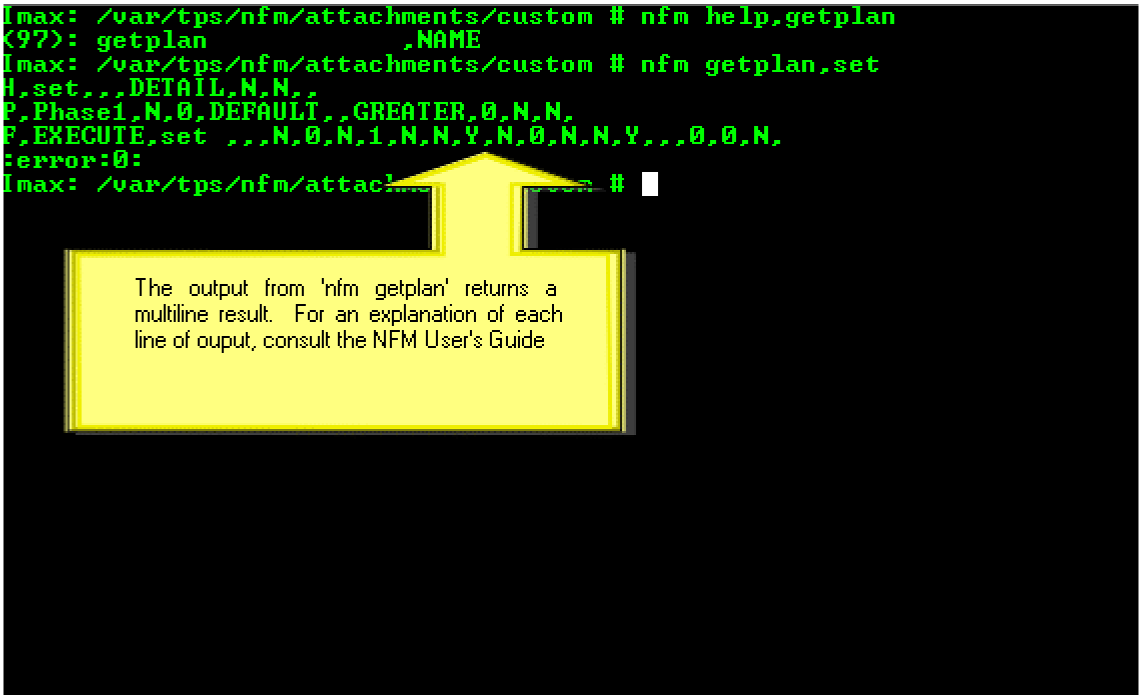
Example 1 (Retrieving a Plan Record):

By using the NFM User's Guide, we know to Retrieve a Plan we use the getplan subcommand. We can then use the 'nfm help' command or the NFM User's Guide to find the proper parameters for 'nfm getplan'.



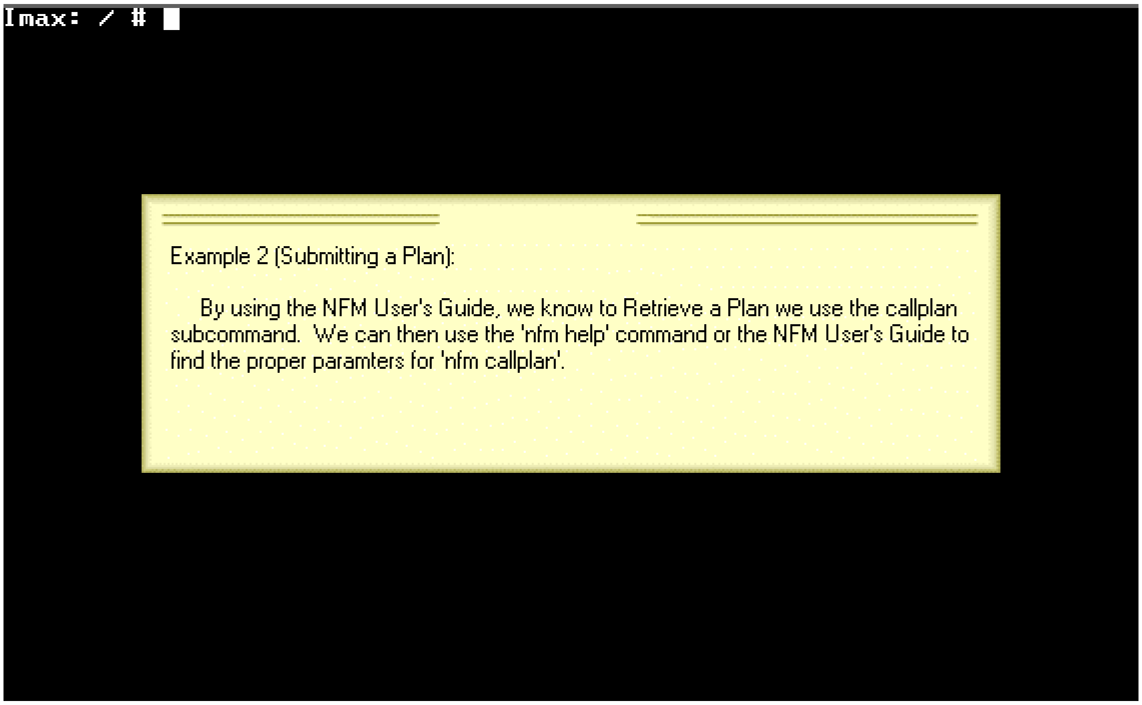
```
Imax: /var/tps/nfm/attachments/custom # nfm help,getplan
(97): getplan              ,NAME
Imax: /var/tps/nfm/attachments/custom #
```

'nfm getplan' takes one parameter, the plan name



```
Imax: /var/tps/nfm/attachments/custom # nfm help,getplan
(97): getplan                ,NAME
Imax: /var/tps/nfm/attachments/custom # nfm getplan,set
H,set,,,DETAIL,N,N,,
P,Phase1,N,0,DEFAULT,,GREATER,0,N,N,
F,EXECUTE,set ,,,N,0,N,1,N,N,Y,N,0,N,N,Y,,,0,0,N,
:error:0:
Imax: /var/tps/nfm/attac               # 
```

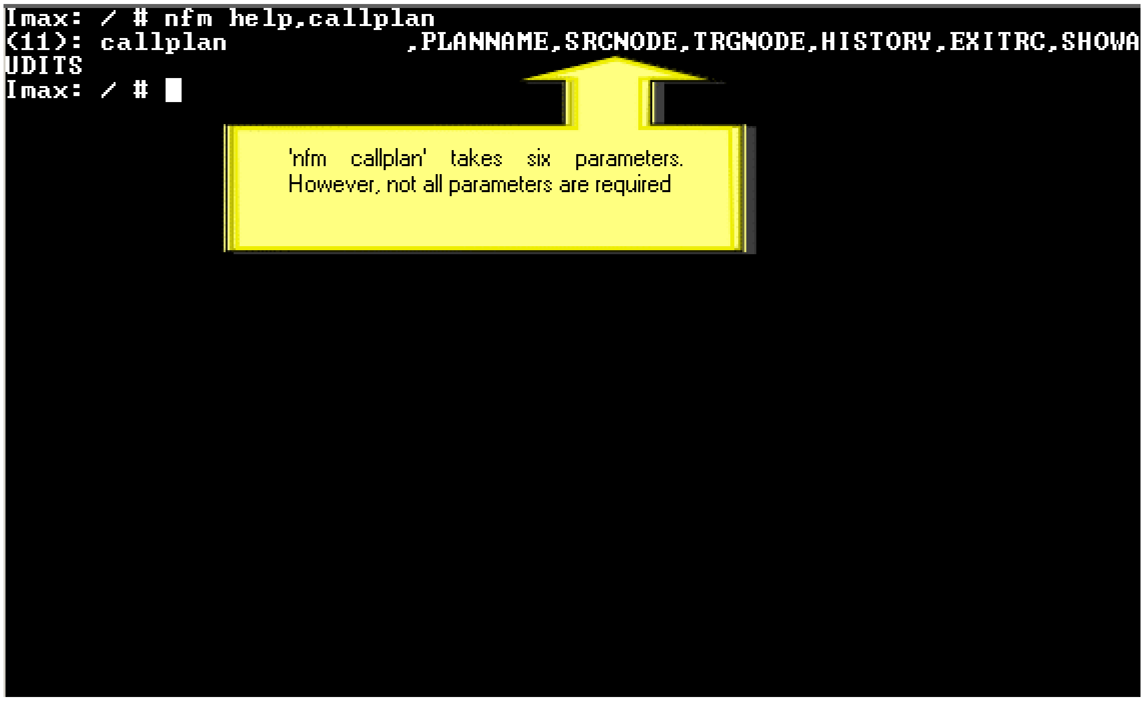
The output from 'nfm getplan' returns a multiline result. For an explanation of each line of ouput, consult the NFM User's Guide



```
Imax: / # 
```

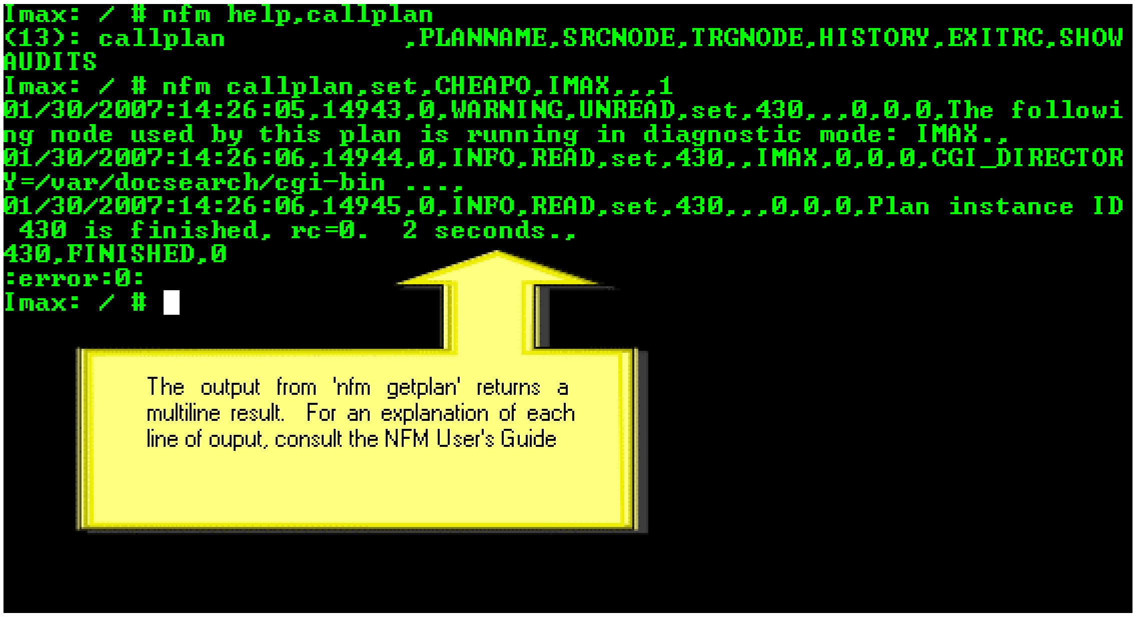
Example 2 (Submitting a Plan):

By using the NFM User's Guide, we know to Retrieve a Plan we use the callplan subcommand. We can then use the 'nfm help' command or the NFM User's Guide to find the proper paramters for 'nfm callplan'.



```
Imax: / # nfm help,callplan
(11): callplan            ,PLANNAME,SRCNODE,TRGNODE,HISTORY,EXITRC,SHOWA
UDITS
Imax: / # █
```

'nfm callplan' takes six parameters. However, not all parameters are required

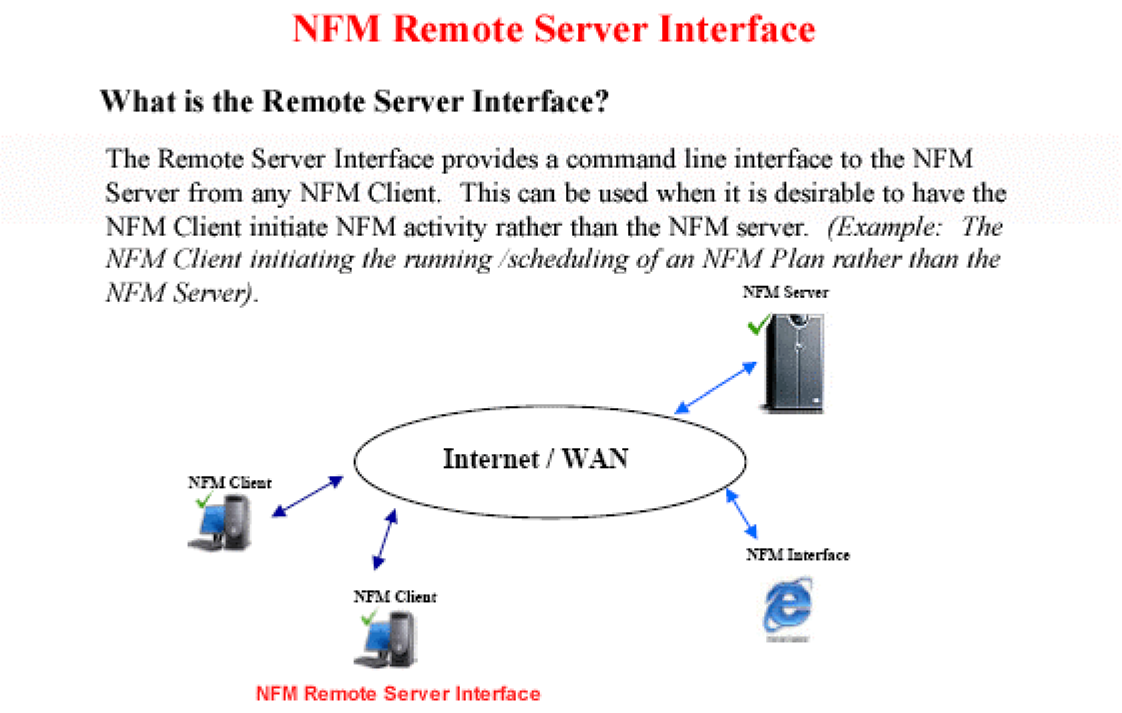


```
Imax: / # nfm help,callplan
(13): callplan            ,PLANNAME,SRCNODE,TRGNODE,HISTORY,EXITRC,SHOW
AUDITS
Imax: / # nfm callplan,set,CHEAPO,IMAX,,,1
01/30/2007:14:26:05,14943,0,WARNING,UNREAD,set,430,,,0,0,0,The followi
ng node used by this plan is running in diagnostic mode: IMAX.,
01/30/2007:14:26:06,14944,0,INFO,READ,set,430,,IMAX,0,0,0,CGI_DIRECTOR
Y=/var/docsearch/cgi-bin ...,
01/30/2007:14:26:06,14945,0,INFO,READ,set,430,,,0,0,0,Plan instance ID
 430 is finished, rc=0.  2 seconds.,
430,FINISHED,0
:error:0:
Imax: / # █
```

The output from 'nfm getplan' returns a multiline result. For an explanation of each line of ouput, consult the NFM User's Guide

Remote Server Interface

A flash demo is available online at http://www.tps.com/nfmdemo/RSI.htm

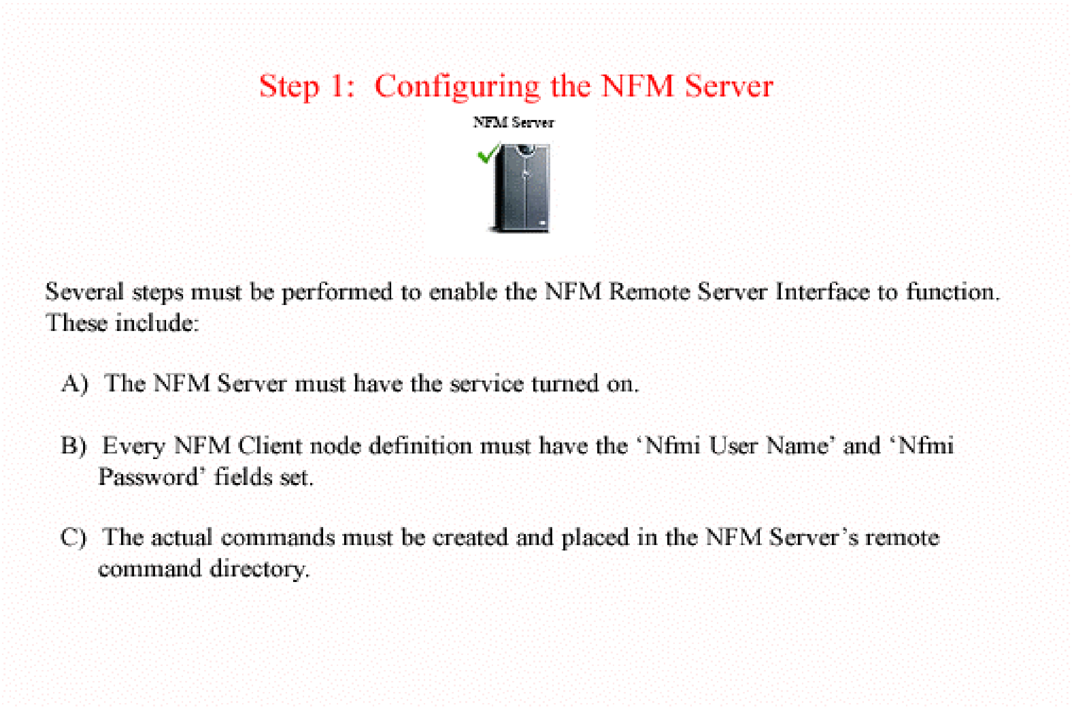


NFM Remote Server Interface

What is the Remote Server Interface?

The Remote Server Interface provides a command line interface to the NFM Server from any NFM Client. This can be used when it is desirable to have the NFM Client initiate NFM activity rather than the NFM server. *(Example: The NFM Client initiating the running /scheduling of an NFM Plan rather than the NFM Server).*

NFM Server
Internet / WAN
NFM Client
NFM Client
NFM Interface
NFM Remote Server Interface



Step 1: Configuring the NFM Server

NFM Server

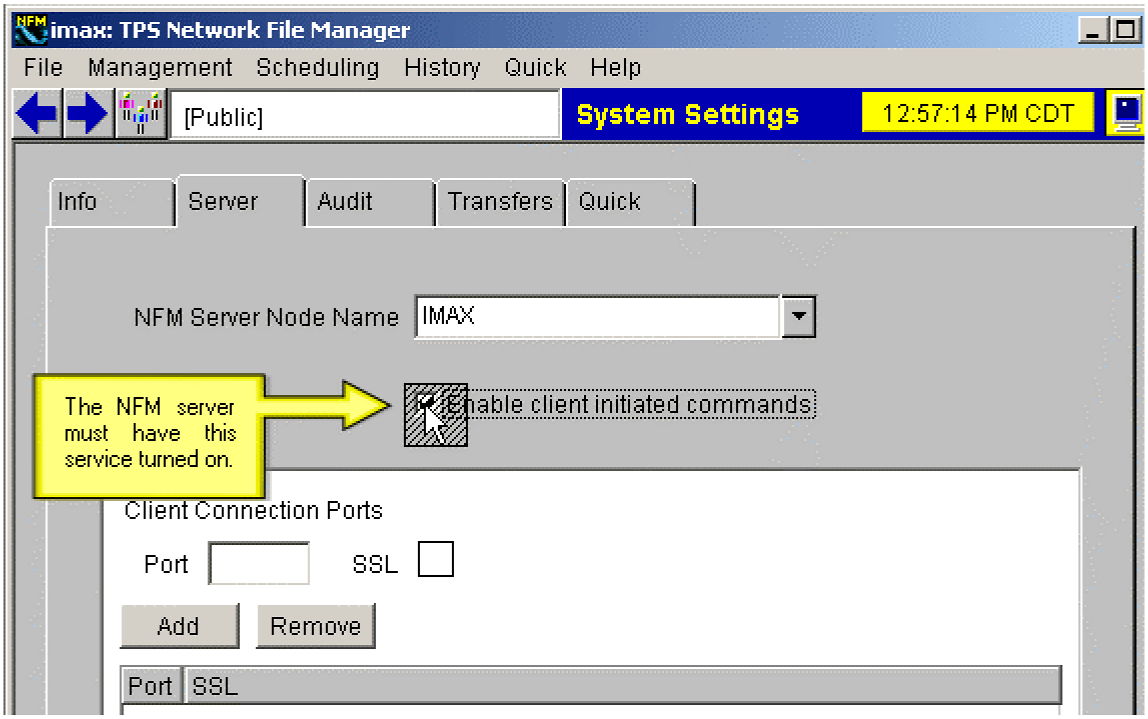
Several steps must be performed to enable the NFM Remote Server Interface to function. These include:

A) The NFM Server must have the service turned on.

B) Every NFM Client node definition must have the 'Nfmi User Name' and 'Nfmi Password' fields set.

C) The actual commands must be created and placed in the NFM Server's remote command directory.

Step 1A: The NFM Server must have the service turned on.



NFM Server Node Name: IMAX

☑ Enable client initiated commands

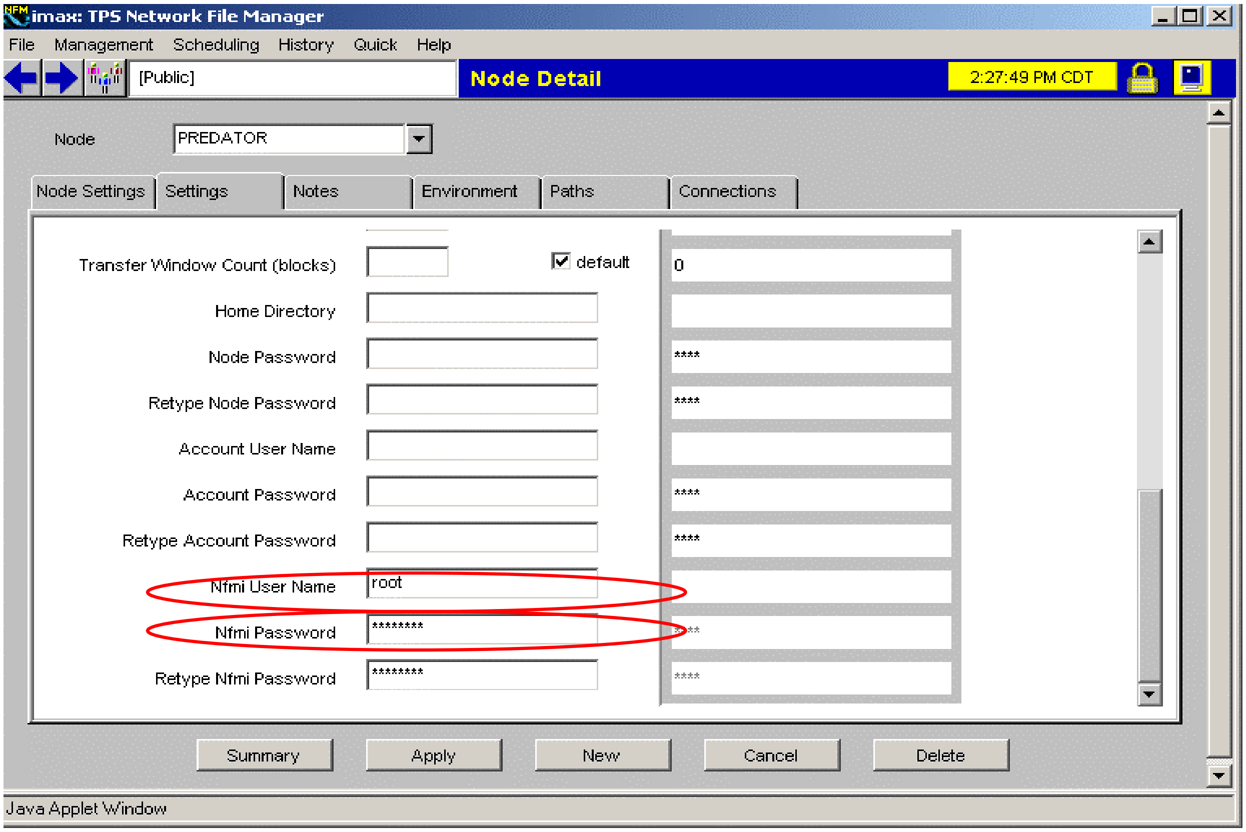
The NFM server must have this service turned on.

Client Connection Ports

Port □ SSL □

Port	SSL

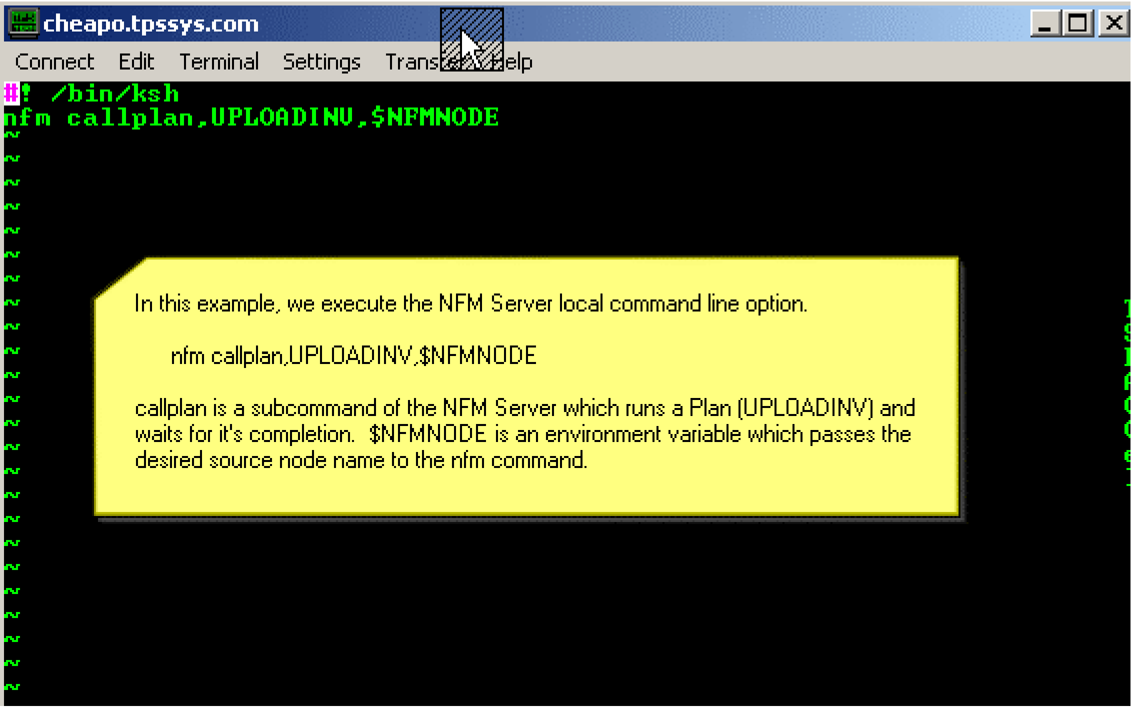
Step 1B: Every NFM Client must have `nfmi User Name` and `nfmi Password` set.



Node: PREDATOR

Transfer Window Count (blocks)		☑ default	0
Home Directory			
Node Password			****
Retype Node Password			****
Account User Name			
Account Password			****
Retype Account Password			****
Nfmi User Name	root		
Nfmi Password	********		****
Retype Nfmi Password	********		****

Step 1C: The actual commands, saved as scripts, must be created and placed in the NFM Server's remote command directory (UNIX: `/var/tps/nfm/commands` and for `Windows: /Program Files/TPS Systems/NFMServer/commands`)



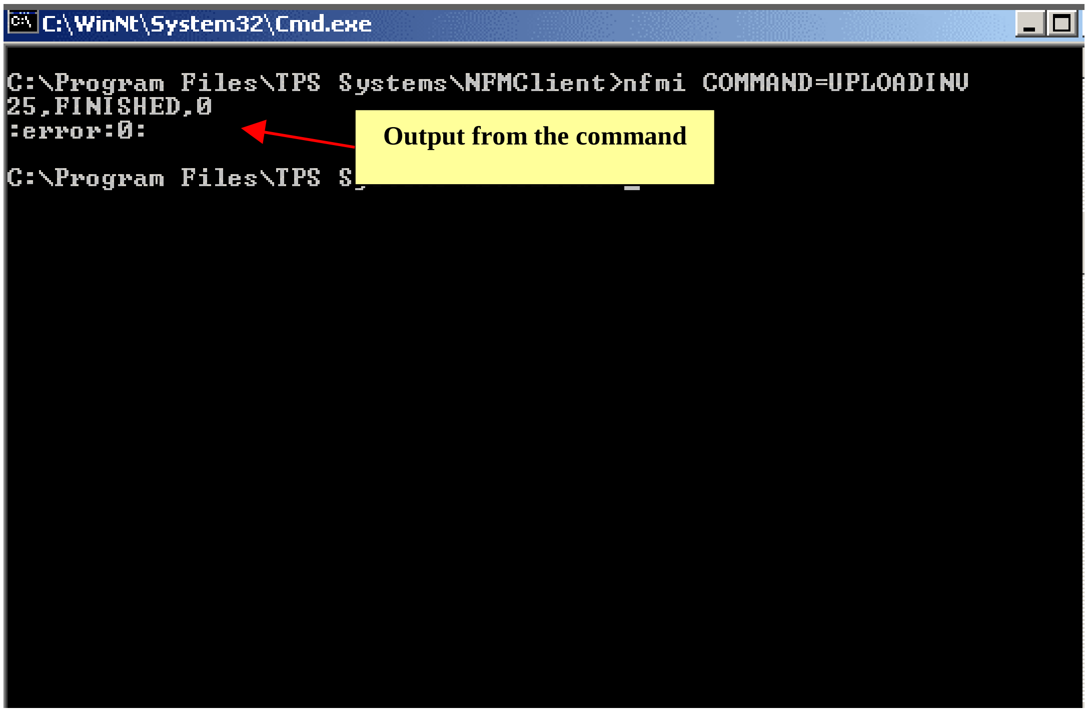
```
#! /bin/ksh
nfm callplan,UPLOADINV,$NFMNODE
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

In this example, we execute the NFM Server local command line option.

nfm callplan,UPLOADINV,$NFMNODE

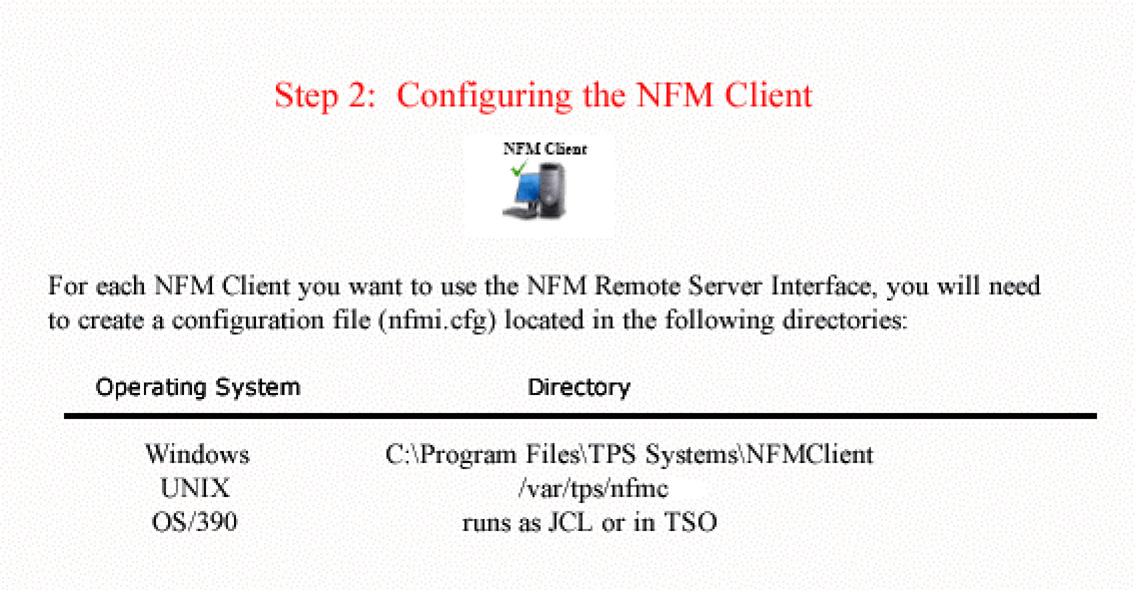
callplan is a subcommand of the NFM Server which runs a Plan (UPLOADINV) and waits for it's completion. $NFMNODE is an environment variable which passes the desired source node name to the nfm command.

Now we can run this command remotely from the NFM Client using the `nfmi` executable. This example will run the UPLOADINV script listed above.



```
C:\Program Files\TPS Systems\NFMClient>nfmi COMMAND=UPLOADINV
25,FINISHED,0
:error:0:

C:\Program Files\TPS S
```

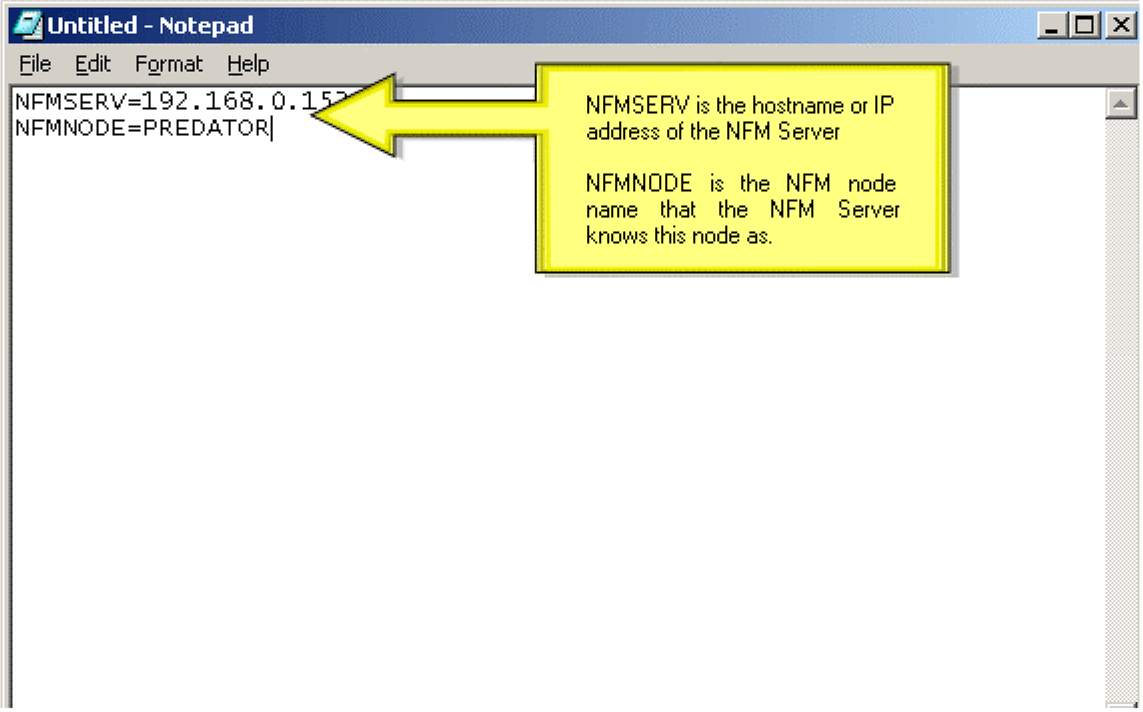
Output from the command



Step 2: Configuring the NFM Client

For each NFM Client you want to use the NFM Remote Server Interface, you will need to create a configuration file (nfmi.cfg) located in the following directories:

Operating System	Directory
Windows	C:\Program Files\TPS Systems\NFMClient
UNIX	/var/tps/nfmc
OS/390	runs as JCL or in TSO



```
NFMSERV=192.168.0.15
NFMNODE=PREDATOR
```

NFMSERV is the hostname or IP address of the NFM Server

NFMNODE is the NFM node name that the NFM Server knows this node as.

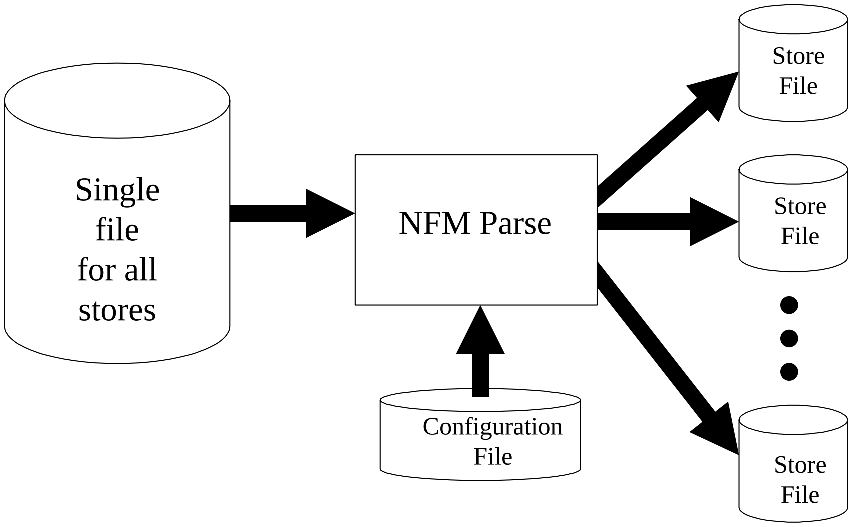
(Editing the NFM Client configuration file. This is a Windows example so we've used notepad; use your favorite text editor for other platforms.)

NFM Parse

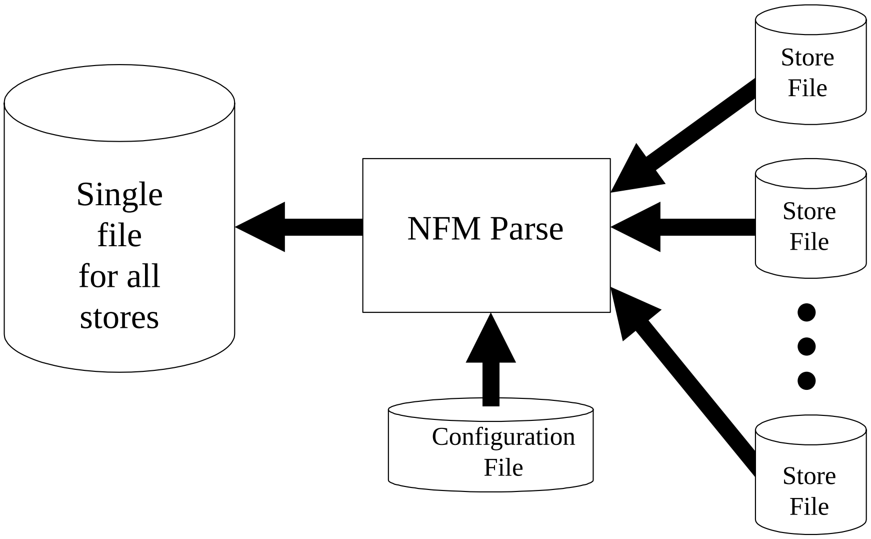
The nfmparse program is a stand-alone program that runs from the command line on an NFM server computer. This command can be used to separate records from a single data file into multiple files that may be destined for multiple nodes (see the figure below). Parsing an input file requires that header records exist, between the data for each node. This header record must contain two things: 1) a string constant that distinguishes it as a header record, and 2) a node identification field that can be mapped to an output file for each node.

A simple text configuration file controls a parsing operation. The configuration file tells nfmparse among other things:

How big the records are
How to recognize header records
How to find the store ID in the header records
How to name the individual output store files



The nfmparse command also performs the reverse operation of combining files into a single data file. This can be a simple concatenation, or an option can be specified to insert a header record between the data that is built from each input file. See the following figure.



Sample Parse

In the following example the file `HOSTDATA` contains data for three stores as identified by the 5-digit store number at offset 4 within each header record (identified by the leading `XXXX`).

File: HOSTDATA:

```
XXXX0000195200307250104570004A0616000734037143SR 0
D0960000900001037143SR
D0960001550001037143SR
D0960003410003037143SR
D0960007810003037143SR
D0960009350001037143SR
D0960010200001037143SR
D0960010700001037143SR
D0960015830001037143SR
XXXX0000205200307250104570004A2125002437037144SR 0
D1070000190001037144SR
D1070000850001037144SR
D0966217570001037143SR
XXXX00003105200307250101560004A1632001928037149SR 0
D1070000190001037149SR
D1070000850001037149SR
D1070001400001037143SR
D1070019760001037149SR
D1070024690001037149SR
D1070028250001037143SR
D1070066670001037149SR
D1070100720001037149SR
D1070101790001037143SR
```

The following configuration file is used to control the parsing operation.

File: parse1.cfg:

```
file=/prod/nfmd/parse/HOSTDATA
delimited=\n
nodesubstitution=store%s
header=XXXX
headerposition=0
membername=cgoorders
storeposition=4
storelength=5
memberdelimiter=\n
```

The configuration file identifies the format of the input file and defines rules for the creation of the extracted store data files. After running the following command:

```
nfmparse options=parse1.cfg
```

The following files are produced:

File: cgoorders.store00001:

```
D09600009000001037143SR
D09600015500001037143SR
D09600034100003037143SR
D09600078100003037143SR
D09600093500001037143SR
D09600102000001037143SR
D09600107000001037143SR
D09600158300001037143SR
```

File: cgoorders.store00002:

```
D10700001900001037144SR
D10700008500001037144SR
D09662175700001037143SR
```

File: cgoorders.store00003:

```
D10700001900001037149SR
D10700008500001037149SR
D10700014000001037143SR
D10700197600001037149SR
D10700246900001037149SR
D10700282500001037143SR
D10700666700001037149SR
D10701007200001037149SR
D10701017900001037143SR
```

The output files are now ready to be shipped out to their corresponding store nodes. Additional options are not demonstrated here. The input or output data could be fixed length (not delimited with new lines). Also the header data could have been included in the output using another option.

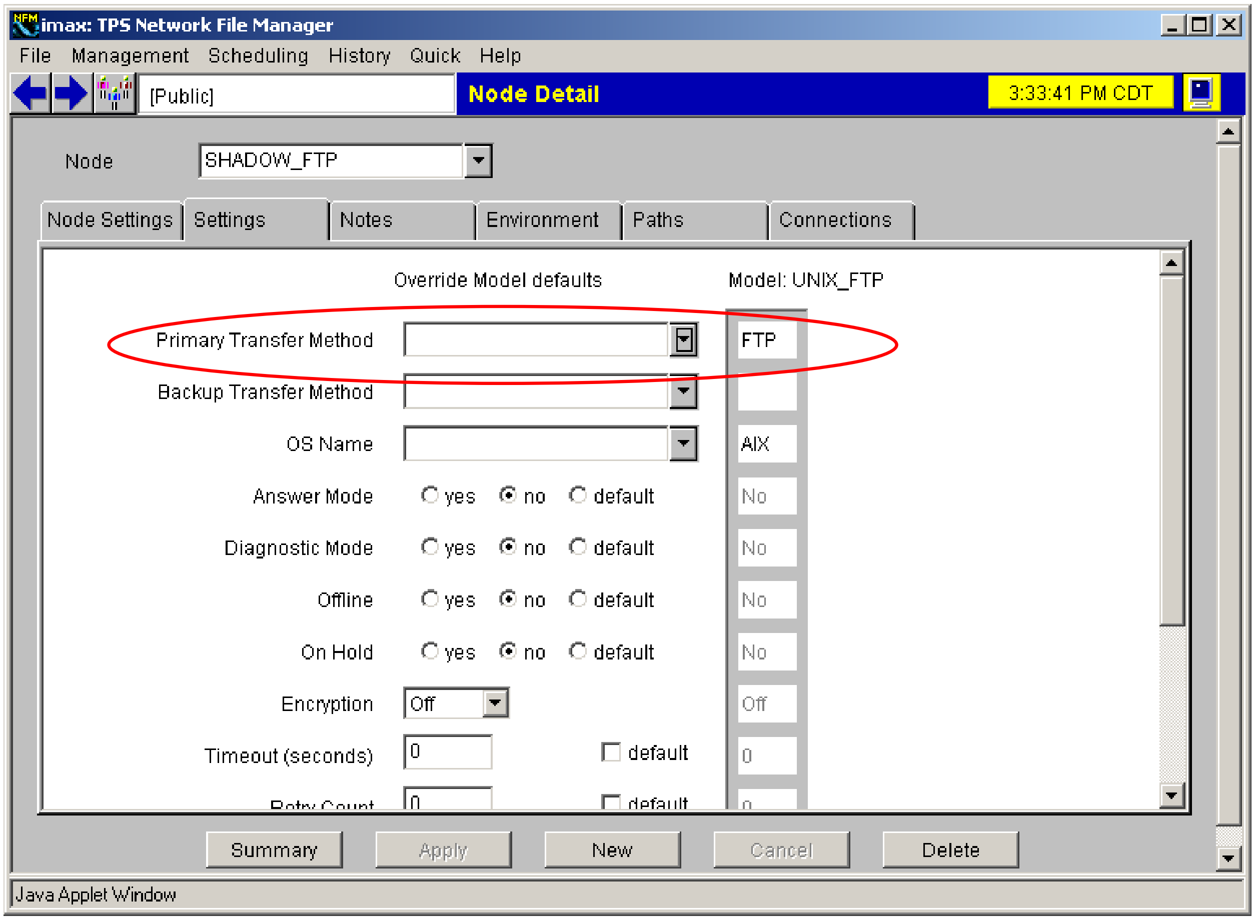
FTP and Secure FTP Nodes

NFM incorporates the use of computers running FTP servers as a basic node type, available for use with normal file activity (transfers, deletes, etc.). This type of node does not require the NFM client to be installed on it in order to be used by NFM, but the support is limited compared to a regular NFM client node. These limitations include:

- The “Execute” type function within a plan is not available for FTP nodes.
- Checkpoint restart is not supported
- Streaming file transfer in not supported
- Plan Monitor screen will not show the same level of detail for the progress bar. Instead of a byte count and a progress bar reflecting the total number of bytes transferred, the progress bar reflects the percentage of the total number of files transferred.
- It is not possible to send files directly between two NFM Non-Client nodes. One node must be a NFM Client node the other may be a `FTP` or `SFTP` Server (NFM Non-Client node).

Because of these limitations, the use of the NFM Client is always preferred. `FTP` and `SFTP` should only be used when it is otherwise not possible to use the NFM client.

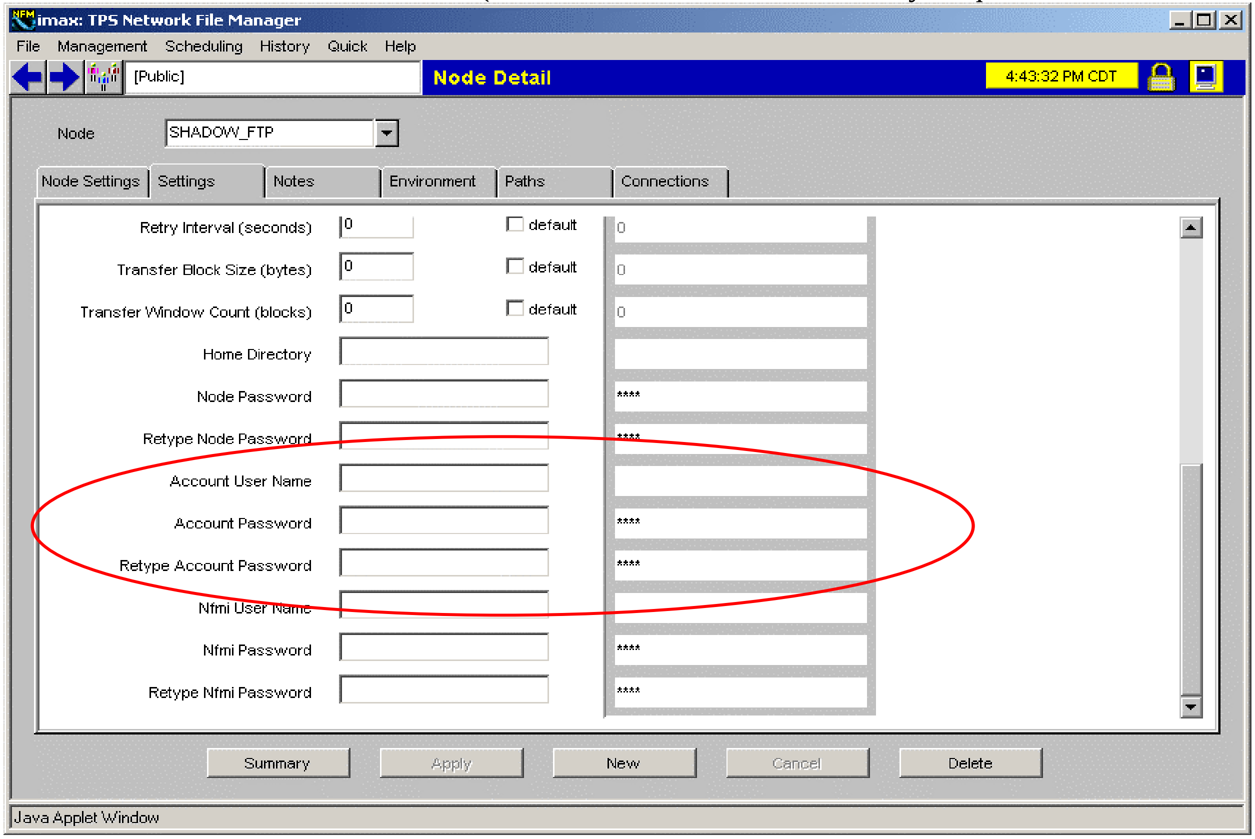
An `FTP` or `SFTP` nodes is created similarly to any other node, but the “Primary Transfer Method” field or the “Backup Transfer Method” in the node record should be set to “`FTP`”, “`SFTP` (`SSH`)”, or “`Kermit FTP`”.



Node: SHADOW_FTP

	Override Model defaults	Model: UNIX_FTP
Primary Transfer Method		FTP
Backup Transfer Method		
OS Name		AIX
Answer Mode	○ yes ● no ○ default	No
Diagnostic Mode	○ yes ● no ○ default	No
Offline	○ yes ● no ○ default	No
On Hold	○ yes ● no ○ default	No
Encryption	Off	Off
Timeout (seconds)	0 □ default	0
Retry Count	0 □ default	0

In addition, the following two fields must be set to supply login parameters: “Account User Name” and “Account Password”. (SFTP and Kermit FTP only require “Account User Name”)



Node: SHADOW_FTP

Retry Interval (seconds)	0	□ default	0
Transfer Block Size (bytes)	0	□ default	0
Transfer Window Count (blocks)	0	□ default	0
Home Directory			
Node Password			****
Retype Node Password			****
Account User Name			
Account Password			****
Retype Account Password			****
Nfmi User Name			
Nfmi Password			****
Retype Nfmi Password			****

For FTP type nodes it is not imperative to specify the “OS Name”, you may select generic for all types except OS/390 FTP node, here you must set it to OS/390.

Customizing an FTP, Kermit FTP or SFTP template

NFM uses a generic FTP template to perform FTP, Kermit FTP and SFTP commands (such as cd, put, get, etc.). However, when the generic template is not adequate, you can create or customize your own template for your plan. This section includes steps for making a customized FTP, Kermit or SFTP template:

Using the NFM Local Command Line Interface, we’ll list existing NFM FTP, Kermit and SFTP templates using the following command:

`cd /var/tps/nfm/attachments/custom` (UNIX)

or

```
C:\Program Files\TPS
                  Systems\NFMServer\attachments\custo
                  m (Windows)
```

If the directory does not exist, create it.

```
nfm list,methodas
```

This displays a list similar to this:

```
ftp.delete,
ftp.delete_w,
ftp.dir,
ftp.filter,
ftp.KermitFTP.common,
ftp.KermitFTP.delete,
ftp.KermitFTP.dir,
ftp.KermitFTP.pull,
ftp.KermitFTP.pull_w,
ftp.KermitFTP.push,
ftp.KermitFTP.push_w,
ftp.mkdir,
ftp.pull,
ftp.pull_w,
ftp.push,
ftp.push_w,
ftp.rmdir,
ftp.sftp.delete,
ftp.sftp.delete_w,
ftp.sftp.dir,
ftp.sftp.mkdir,
ftp.sftp.pull,
ftp.sftp.pull_w,
ftp.sftp.push,
ftp.sftp.push_w,
ftp.sftp.rc,
ftp.sftp.rmdir
```

(continues)

2) Extract the template that most resembles the one you want to customize. Template names are named according to what they do. For example, ftp.pull is a template that pulls a file from the FTP Server. Files ending with `_w` are templates that allow you to do wildcards.

   (In this example we will work with the NFM ftp.push template)

   ```
   nfm get,methoda,ftp.push
   ```

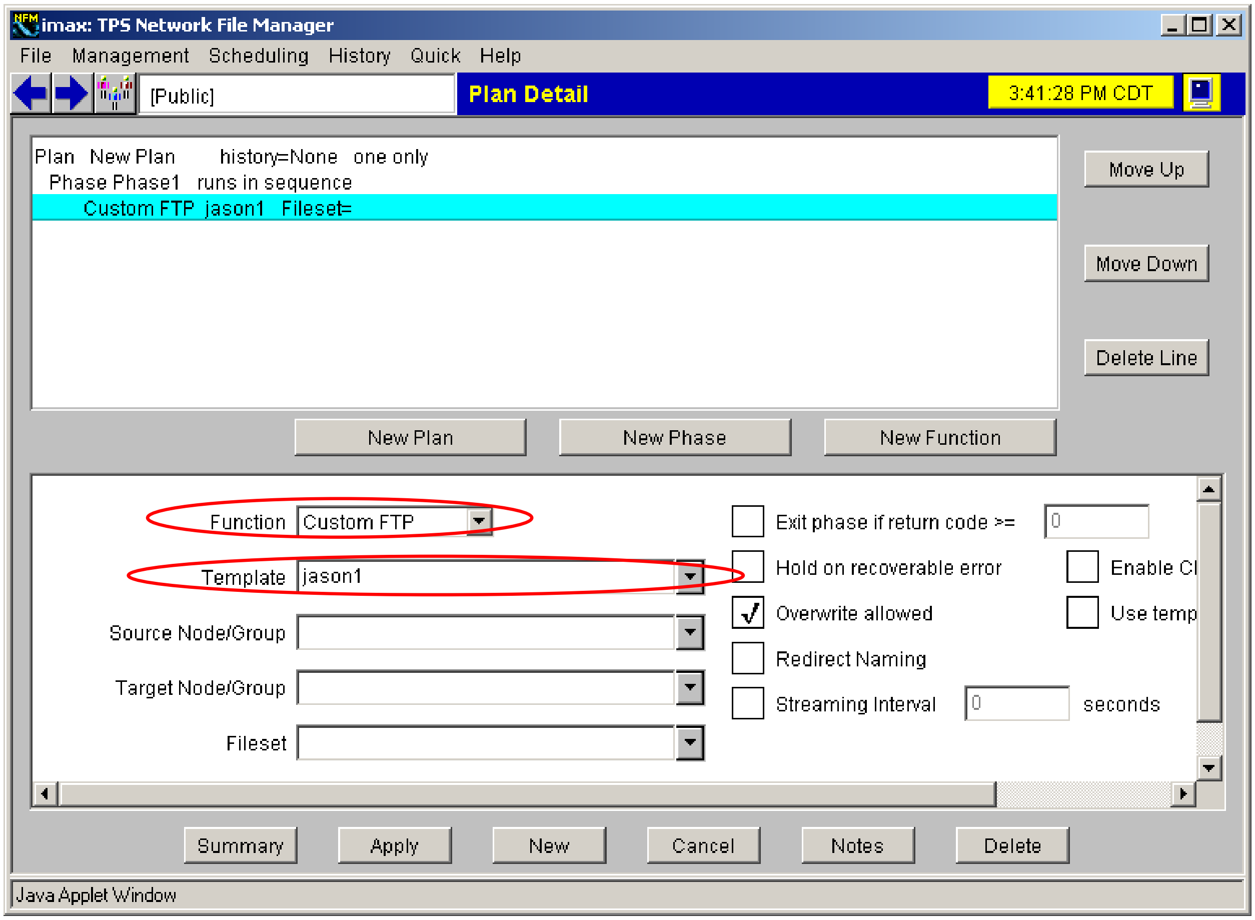
3) Rename the ftp.push file and edit it

ftp.push template

```
-----------------------------------------------------------------
open $(TRGNODE)
```

```
quote user $(FTPUSER)
quote pass $(FTPPASSWORD)
$(FILETYPE)
put "$(SRCFILE)" "$(TMPFILE1)"
rename "$(TMPFILE2)" "$(TRGFILE)"
quit
```

Notice the use of environment variables. In this example, the variables, designating with the `$(variable_name)`, are NFM specific environment variables that are substituted when the template in the plan is run. You can also use Node and Model environment variables or NFM system environment variables. Once you're done editing your template you can specify that template inside the plan's function.



```
Plan  New Plan      history=None  one only
  Phase Phase1  runs in sequence
      Custom FTP  jason1   Fileset=
```

Function: Custom FTP
Template: jason1
Source Node/Group:
Target Node/Group:
Fileset:

□ Exit phase if return code >= 0
□ Hold on recoverable error
☑ Overwrite allowed
□ Redirect Naming
□ Streaming Interval 0 seconds
□ Enable Cl
□ Use temp

Secure FTP

NFM also provides secure FTP functionality when used in conjunction with either the "Kermit FTP" client program or the "Secure FTP" (or SFTP) as part of the SSH protocol standard. Similar to basic FTP, these protocols are used by specifying the appropriate "Transfer Method" fields in the node or model, in this case "Kermit FTP" and "SFTP (SSH)" respectively.

Acquiring the client software programs for these protocols, as well as satisfying any licensing requirements, is the customer's responsibility. These two client programs are recommended for use with the NFM product due to their widespread availability and acceptance within the industry.

The term SFTP tends to be re-used for different things. As it is referenced here, it refers to the command-line client program `sftp` that provides secure FTP over SSH.

NOTE REGARDING SUPPORT:

NFM provides the functionality to run `SFTP` and `FTP` through the operating system's command prompt. Since `SFTP` and `FTP` are third party applications, TPS will only provide support for problems with NFM executing `SFTP` or `FTP`. Always try to establish a connection, setup encryption keys, etc. from the command prompt first before using NFM. Problems related to the connection (example: the key exchange or configuration), `SFTP/FTP` command line options, etc.; are outside the scoop of NFM and thus, the customers' responsibility.

An excellent source of information on this protocol is available at http://www.openssh.org.
For more information on Kermit, please visit http://www.columbia.edu/kermit/.

Filtering FTP (Error) Messages

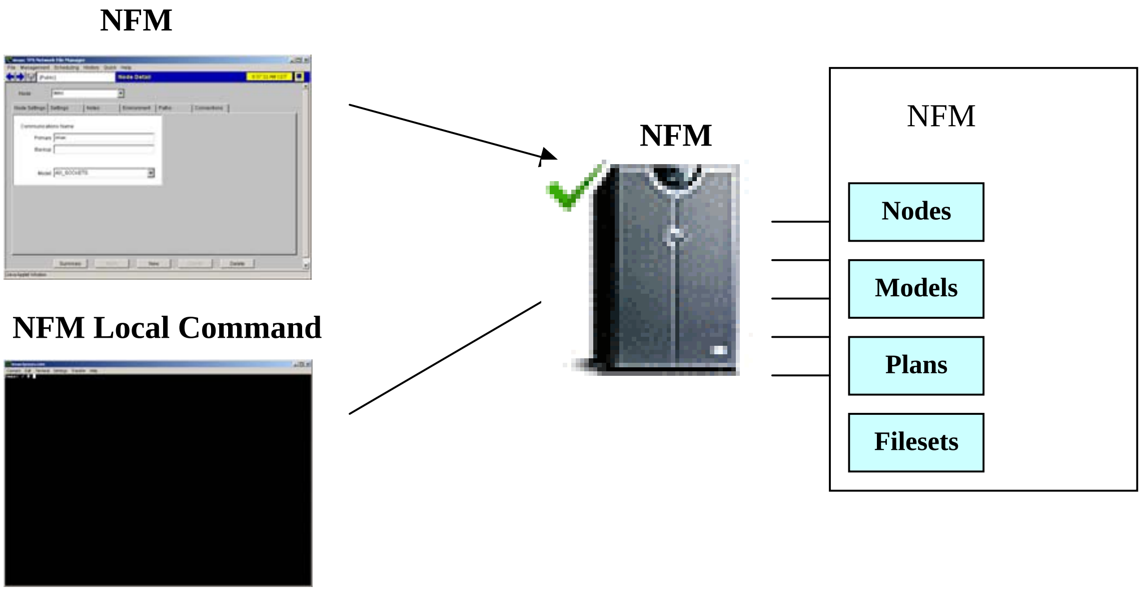
When the NFM Client establishes an FTP connection it will treat ANY messages received back from the FTP Server as errors and log them in the Audits database. This will cause the plan to exit with a bad return code (non-zero). To filter out these messages and treat them as non-errors you will need to edit the `filter.ftp` file in the NFM Server's attachments directory (see '*Installing the NFM Server package*' for the full path).

Example of an `filter.ftp` text file:

```
220-
220-######################################################################
220-Welcome to your FTP server!
220-Unauthorized access is criminal
```

NFM Server Database Components

All configurable items (Users, Nodes, Models, and others located under 'Management' toolbar) as well as non-configurable items (Audits) of the NFM system reside in a database format located on the NFM Server. This database is accessible through the NFM GUI Interface and the NFM Local Command Line Interface.



The NFM Database is located in the following directory:

Operating System	Directory
Windows	`C:\Program Files\TPS Sytems\NFMServer\dba\data`
UNIX	`/var/tps/dba/data`

The NFM Server uses several tables to store record information. Some GUI screens access several tables to populate the screen (for example: the Node screen uses ENVIRONMENTS, HARDPROF, HARDWARE, NOTES, OPSYS, MODELS). Below is a list and description of tables in the NFM database:

Table Name	Description
AUDITS	Audits
CONFIGS	Node Connection Entries (Node and Model's Connection tab)
DAYS	Day Scheduler

Table Name	Description
ENVIRONMENTS	Environment Variables (User Defined)
EXTRACTS	Performance Extracts
FILESETS	Filesets
HARDPROF	Hardware Profile
HARDWARE	Hardware
HOURS	Hour Scheduler
JOBNODES	Node Status per Plan Instance (when a job is put on hold, etc.)
JOBS	Job Activity
MCPROFS	Multicast Profiles
METHODATT	Non-Client Node (FTP) templates
METHODENV	Non-Client Environment Variables
METHODS	Configuration per Platform
MODELS	Models
MODEMS	Modems
NODEACCESS	Per User Node Access Configuration
NODEGROUPS	Nodegroups
NODES	Nodes
NOTES	Notes
OPSYS	Operating System definitions
PATHS	Paths Permissions
PLANS	Plans
QPLANS	Quick Plans
STATSD	Stats Detail (used for Statistical Gathering)
STATSH	Stats Header (used for Statistical Gathering)
SYSTEM	System Settings
TRACKING	Plan Retry (markers)
USERGROUPS	Usergroups
USERPERMS	Users Permissions per User/Usergroup
USERSN	Users Configuration

Any files named with a .bak extension are backup copies before the database was reorganized. A reorganized database usually occurs when there is an upgrade of the NFM Server and the database needs to be restructured.

Extracting database information

There are several ways to extract or view the NFM database records.

The best way to view a database is to use the nfmfile command. This provides you with field names and values displayed on the screen in an easy to read format.

Example: Viewing the NODES database

`nfmfile dump -f NODES` (See Appendix D: NFM Commands & Parameters for options)

```
Record 7137
--------------------------------
USERGID              : 0
NAME                 : IMAX_FIREWALL
MODEL                : AIX_SOCKETS
MODELUGID            : 0
COMNAME              : 198.81.193.168
COMNAME2             :
OFFLINE              : 0
ONHOLD               : 0
ENCRYPT              : 0
DIAGNOSTIC           : 1
ANSWERMODE           : 0
MODTIME              : 1154019037
OPSYS                :
METHOD               :
METHOD2              :
TIMEOUT              : 0
RETRYCT              : 0
RETRYINT             : 0
TRANSBLKSZ           : 0
TRANSWINCT           : 0
HOMEDIR              :
```

Data can be extracted directly from the database tables into text files using the `nfm export` command. This presents the data in text form that can be edited and imported back to the database. This can be useful for entering large amounts of configuration data when using the GUI might be cumbersome (for example, initially defining 1000 nodes to the system).

Example: Extracting the NODES database to a text file

```
nfm export,out.txt, NODES

out.txt file
-----------------------------------------------------------------
------------------------------------------
USERGID="0"    NAME="ADEPT"    MODEL="WIN98_SOCKETS"    MODELUGID="0"
COMNAME="adapt"       COMNAME2=""        OFFLINE="N"        ONHOLD="N"
ENCRYPT="ENCRYPTOFF"          DIAGNOSTIC="N"          ANSWERMODE="N"
MODTIME="07/14/2006:19:33:06"    OPSYS=""     METHOD=""     METHOD2=""
TIMEOUT="0"  RETRYCT="0"  RETRYINT="0"  TRANSBLKSZ="0"  TRANSWINCT="0"
HOMEDIR=""
```

Importing database information

To import a text file into a NFM database use the `nfm` command.

Example: Importing a text file to the NODES database

```
nfm import,in.txt,NODES
```

Reorganizing the database files

A reorganizing of the NFM databases happen every time you install the NFM Server either via an upgrade or a fresh install. This will either create the database, if it doesn't exist, or check an existing database to make sure the database layout (field names, sizes, etc.) is current with NFM. We provide a utility program (`nfmreorg`) if you need to reorganize the database manually:

`nfmreorg`	Shows which databases need to be reorganized
`nfmreorg –r`	Reorganizes all databases

If `nfmreorg` needs to change a database a backup file will be created:
`<database_name>.bak`

OS/390 Considerations

Install and startup

Components are installed on OS/390 using the `TSO RECEIVE` command to install a product image file. This file should be placed in a temporary dataset on the target computer.

Starting the NFM client:

> The NFM client must be started and left executing on OS/390 for it to respond to incoming request from the NFM server. The load module can be executed as a ‘started task,’ or simply executed as a batch job.

Example JCL for starting the NFM Client for OS/390:

```
//TPS01//TPS01NFM JOB (TPS0),'TPS.SYS',
//             CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//RUNNFMC  EXEC PGM=NFMC,REGION=4096K,PARM='-T'
//STEPLIB  DD DISP=SHR,DSN=TPS01.NFMC.LOAD
//NFMJOB   DD SYSOUT=(A,INTRDR)
//NFMLOG   DD SYSOUT=A
//NFMERR   DD SYSOUT=A
//
```

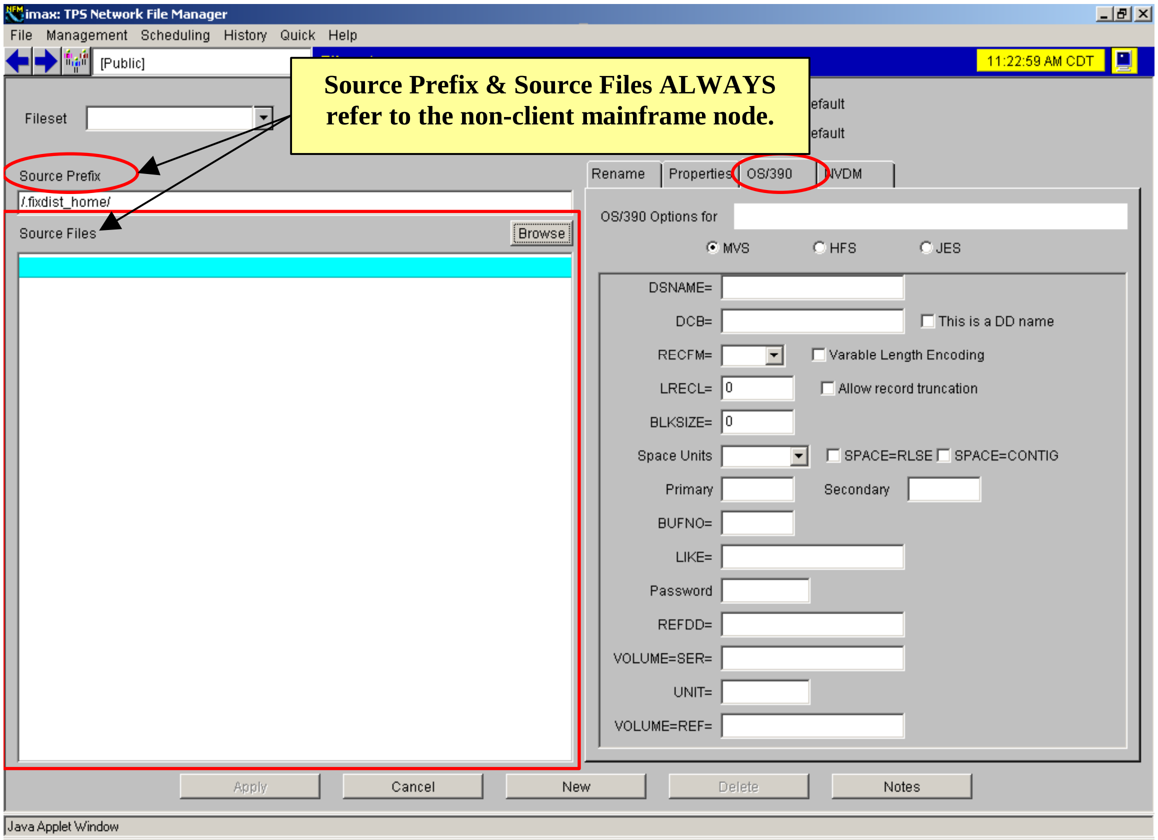
Starting the NFM remote server interface:

> The NFM remote server interface program, or NFMI, is available in `TPS01.NFMC.LOAD(NFMI)`. It can be loaded as a batch program using JCL, or as a direct interactive program for TSO.

Example JCL for starting the NFM remote server interface for OS/390:

```
//TPS01NFI JOB (TPS0),'TPS.SYS',
//             CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//RUNNFMI  EXEC PGM=NFMI,REGION=4096K,PARM='CMDFILE=CMD.DATA'
//STEPLIB  DD DISP=SHR,DSN=TPS01.NFMC.LOAD
//NFMLOG   DD SYSOUT=A
//NFMERR   DD SYSOUT=A
//
```

Filesets



[Public]

Source Prefix & Source Files ALWAYS refer to the non-client mainframe node.

Source Prefix: /.fixdist_home/

Source Files

OS/390 Options for

● MVS ○ HFS ○ JES

DSNAME=

DCB= □ This is a DD name

RECFM= □ Varable Length Encoding

LRECL= 0 □ Allow record truncation

BLKSIZE= 0

Space Units □ SPACE=RLSE □ SPACE=CONTIG

Primary Secondary

BUFNO=

LIKE=

Password

REFDD=

VOLUME=SER=

UNIT=

VOLUME=REF=

** When files are transferred to and from a mainframe, the Source Prefix and Source Files fields always apply to the non-mainframe node, regardless of the direction of the transfer. **

NFM only deals with “sequential” datasets, PDS/PDSE members and HFS files.

Misc Notes

Language Environment (LE) runtime options can be specified in the `PARM=string`. These LE options control how memory is used, what kind of dumps are created, etc.

The `builtin` (programmer default) LE options are:

```
   ANYHEAP(1M,512K,ANYWHERE,KEEP)
   BELOWHEAP(128K,32K,KEEP)
   LIBSTACK(0,0,FREE)
   STACK(16K,4K,ANYWHERE,KEEP,0,0)
   HEAP(4M,512K,ANYWHERE,KEEP,0,4K)

HEAPPOOLS(ON,40,1,120,1,408,1,560,1,1264,2,2048,2,2272,2,2936,7,5
816,2,32152
              ,90,64000,13,65536,51)
   THREADSTACK(ON,32K,32K,ANYWHERE,KEEP,0,0)
```

We recommend ALL31(ON) if that is not the installation default:

```
   //RUNNFMC EXEC PGM=NFMC,REGION=4096K,
   //                   PARM='ALL31(ON)/-T'
```

Note the / after LE options, before program arguments.

There are many other LE options.
The best values for each are installation dependent.
For complete details, see the following IBM publications:
  Language Environment Programming Guide SA22-7561-05
  Language Environment Programming Reference SA22-7562-06

4690 Considerations

Install and startup

The installation procedure for the NFM client on the 4690 OS involves simply creating a directory, copying the binary files into this directory, and then arranging for the `nfm` client program to be started (usually as a background application). For information on creating a background application, reference the 4690 User's Manual.

File naming conventions

As with all operating system platforms, it is the users' responsibility to construct filenames that are compatible with the file system in use on any given operating system. The 4690 OS file system looks very similar to Windows. The file system may be limited to a single name (8 characters maximum) and dot extension (3 characters maximum) for any given file within a subdirectory. Forward and backward slashes are supported as directory separators but forward slashes are recommended. Environment variables are called "logical names" on the 4690 and those that end in a colon **:** typically refer to directory locations and may be included as part of a filename within NFM.

Program execution

All commands that are executed from within plan functions on the 4690 OS are automatically done from within a batch program regardless of whether or not `Use Shell` is checked in the plan. Environment variables supplied by NFM are available as "logical names" to function commands.

Environment Variables

The NFM environment variables enhance the functionality of the NFM system by providing a flexible way to pass information to functions or filesets of an NFM plan instance. There are two types of environment variables available, user configured and NFM provided. The user-configured variables consist of those variables defined on the Model and Node definitions. This provides a means of customizing filenames and executed commands so they can perform different operations based on the current node they are working with. The NFM provided environment variables are automatically generated during a plan execution. These variables are listed below.

Environment variables can be used in two different ways. Variables can be referenced in an executed command of plan function as well as prefix and filename fields that are referenced from within a fileset. This is done in the form `$(VARIABLE)` inserted anywhere in the text of these fields. In this case the substitution is done on the NFM server prior to the command or filenames being used during plan execution. All NFM environment variables are also passed down to the NFM client during a plan execute type function, and are available in local context as real operating system environment variables.

User configured environment variables

The user configured environment variables are defined on the Node and Model screens under the ‘Environment’ tab. An environment variable assigned to a model will be available for substitution to any nodes defined to use that model, unless the same environment variable name is specified in the Node record where it will take precedence. Consider the following example:

> I have 100 store nodes, half of which are Windows/95 computers, the other half Windows/NT. I have a software package that resides in `C:/myapp` on the 95 computers, but resides in `D:/myapp` on the NT computers. The exception to this is two of the NT computers; `store099` and `store100` have it installed in `F:/myapp`. By setting up the following environment variables, I can send a single file to every store’s correct location with a single command.

```
  Nodes STORE001 through STORE050 share the same model called WIN95.
  Nodes STORE051 through STORE100 share the same model called WINNT.
  The WIN95 model record has the environment variable MYAPP=C:/myapp.
  The WINNT model record has the environment variable MYAPP=D:/myapp.
  The node records STORE099 and STORE100 each have the environment
variable
     MYAPP=F:/myapp.   (This overrides the default model value of
D:/myapp for
     these two stores.)
  Now define a fileset definition called MYAPPFILES with the following:
     Source File=/tmp/data.001
     Rename File=$(MYAPP)/data.001
  Finally define a plan with a function that copies the following:
     Transfer MYAPPFILES from source_node CENTRAL to target_node
ALLSTORES (ALLSTORES is a node group that contains all 100 stores).
```

Variable substitution during plan execution will provide the desired resulting copies:

```
CENTRAL:/tmp/data.001 → STORE001:C:/myapp/data/data.001
. . .
CENTRAL:/tmp/data.001 → STORE050:C:/myapp/data/data.001
CENTRAL:/tmp/data.001 → STORE051:D:/myapp/data/data.001
. . .
CENTRAL:/tmp/data.001 → STORE098:D:/myapp/data/data.001
CENTRAL:/tmp/data.001 → STORE099:F:/myapp/data/data.001
CENTRAL:/tmp/data.001 → STORE100:F:/myapp/data/data.001
```

NFM provided environment variables

The following list represents the NFM provided environment variables. These, along with the user defined environment variables from the current source or target node (or their models), are available for substitution during a plan execute function. When used as part of the command itself they should be referenced as above with the `$(VARIABLE)` convention. The variables are also sent down to the target client node and established as operating system environment variables, so they will be available from within a batch or script using the appropriate operating system syntax.

`NFMEXECS`	Current executable command string.
`NFMPHASE`	Current phase name.
`NFMPHSRC`	Highest function return code so far in current phase.
`NFMPRPRC`	Previous phase return code.
`NFMPLAN`	Current plan name.
`NFMPLNID`	Current plan instance ID number.
`NFMSNAME`	Current source file's base name (no prefix) after expanding wildcard entries (see example below).
`NFMSNODE`	Current source node name.
`NFMSCOMM`	Current source node communications name.
`NFMSERV`	The name of the node representing the server.
`NFMTNODE`	Current target node name.
`NFMTCOMM`	Current target node communications name.
`NFMTMODL`	Target node model name.
`NFMTOPER`	Target node operating system name.
`NFMUGID`	The usergroup ID of the user who submitted this plan.
`NFMUSER`	The name of the user who submitted this plan.
`NFMWILD`	Current source file's wildcard segment of base name after expanding wildcard entries.

(The remaining variables represent the date and time on the NFM server when the current function was started):

`NFMTM_YR`	Year (4 digits).
`NFMTM_Y2`	Year (2 digits).
`NFMTM_MO`	Month (2 digit, `01-12`).
`NFMTM_DY`	Day (2 digit, `01-31`).
`NFMTM_JD`	Julian date or day of year (3 digit, `001-366`).
`NFMTM_HR`	Hour (2 digit `01-12`).

NFMTM_AP AM/PM indicator (AM or PM).
NFMTM_HM Hour in military time (00 - 23).
NFMTM_MI Minutes (00 - 59).
NFMTM_SE Seconds (00 - 59).

Consider the following example:

I have a directory on a node called /tmp that I wish to backup once a day. I do not know ahead of time the names of the files so I want to select every file in the directory by using wildcards. I would like to rename each file by adding the current day of the year (Julian date) as a string appended to the end of each file name. By doing this I will create a collective backup whereby each new day that this plan is run I will be creating a different extension and therefore a new copy of each file in the target directory.

```
 Nodes will be called STORE001 and STOREBACK.
 Define a fileset definition called backfiles with the following:
    Source Prefix=/tmp/          Rename Prefix=/tmp.back/
    Source File=*                Rename File=$(NFMSNAME).$(NFMTM_JD)
 Plan  backplan  will  copy  the  fileset  backfiles  from  STORE001  to
STOREBACK.
```

For this example, let’s say we have three files: file1, file2, and file3 in the /tmp directory currently. Let’s also assume it is February 1st. The Source File specification of asterisks indicates to pick up each file in the directory. As each file is copied the target filename substitutes the $(NFMSNAME) with the source files base name, and substitutes the $(NFMTM_JD) with the current Julian day of year to cause the desired copying.

```
 STORE001:/tmp/file1 → STOREBACK:/tmp.back/file1.032
 STORE001:/tmp/file2 → STOREBACK:/tmp.back/file2.032
 STORE001:/tmp/file3 → STOREBACK:/tmp.back/file
```

NFM Multiple Server Support

Multiple NFM servers may be configured to work collectively. This type of implementation presents itself to the end user as a single "virtual" NFM server. This allows administrators or other authorized users complete access to all NFM activity across multiple servers with a single login through the existing user interface. This single system look and feel is maintained even while the actual workload (plan processing, file transfers, etc.) may be spread out among multiple physical servers.

This is accomplished by having the multiple servers share a common NFM database. All data in the database is available to all servers in the collective, which makes the single virtual system look identical to all servers.

There are two types of NFM server designations, "`Primary`" and "`Secondary`." A primary server is one that controls and accesses its own database in a local file system. A secondary server accesses an NFM database remotely over a network connection to a primary NFM server computer. Among a group of NFM servers working collectively, there can only be one primary server, and any number of secondary servers.

The division of work among the servers is controlled using the existing usergroups (the physical partitions that allow NFM items to be logically subdivided among different groups of users). Each usergroup may be assigned a "`home`" server computer. Any time a plan from the usergroup is scheduled to run, the home server computer will actually perform the work. The plan may still be monitored and controlled from a different server, but the actual processes and connections involved will occur on the home server.

In addition to plan activity, an NFM server provides a user interface session to users that are logged in through the browser interface (or stand-alone interface). A user that is logging in to NFM may actually connect in to any of the NFM servers (`primary` or `secondary`) that are participating in the collective virtual server, and they will see the same content. It is recommended however, that users connect in to the "home" server of the usergroup that they belong to (or will spend most time working with). There are two reasons for this. First, a user will get better session performance logging in to the server that is running their plans (especially while monitoring plan activity). Second, and perhaps more important, if a secondary server loses network connectivity to the primary (and thus the database), it is still possible for it to perform processing independently. This means that the server may still be able to run plans, connecting to local nodes, and so on, even without being in contact with the primary server. In the event that this occurs, a user will need to be logged in directly to this NFM server to monitor and control local activity.

Configuration

When an NFM server is installed on a computer, it automatically creates a local database and assumes the role of a primary NFM server. Configuring a multi-server setup basically consists of separately installing multiple NFM servers (each one being a separate primary by default), and then re-configuring all but one of them to serve as secondary servers.

Since a stand-alone NFM server is already serving as a primary, no special consideration needs to be given about its role when it is installed. A customer may choose to add secondary servers years later to an existing NFM server. However, when installing a new server that may or may not become a secondary, consideration should be given to what its initial role will be. A primary server that is converted to a secondary server will lose any database items that were created on it while it existed as a primary. (Actually, existing database items may be saved off and re-integrated back into a collective setup, using the export and import functions, but this will involve additional setup work).

The following steps should be taken to create a secondary server and integrate it into a multiple server environment.

1. The primary server should already be installed, configured, and working properly.

2. A server that will become the secondary server should have the NFM server software installed on it. This will temporarily create an additional primary server running independently. Do not create any configuration items (nodes, models, etc.) on this server, as they will be lost after reconfiguring the server.

   Note: Existing data will not be erased; it will simply not be referenced any longer once the server is accessing the remote database. If necessary, the data can be recovered.

3. Log into NFM on the primary server and create a secondary server record representing the secondary server being added (choose a unique name for it (the name "`SERVER02`" is used here as an example). Refer to the section "Servers," in Chapter 4, "Configuration" for more details on working with server configuration records.

4. Stop the NFM server process on the secondary server.

5. Edit the NFM server configuration file on the secondary server. The file should be in one of the following locations and should initially look similar to the one shown below.

   UNIX: `/var/tps/dba/nfmfcp.cfg`
   Windows: `C:\Program Files\TPS Systems\NFMServer\dba\nfmfcp.cfg`

```
SERVERNAME=SERVER01
FCPCONFIG=/var/tps/dba/nfmfcp.cfg
```

This represents a default configuration for a primary server. Change the value of the field `SERVERNAME` to the exact name chosen for the new server record created in Step 3 above.

```
SERVERNAME=SERVER02
FCPCONFIG=/var/tps/dba/nfmfcp.cfg
```

Note: The second field references the NFM database configuration file, which will be changed in the next step.

6. Edit the NFM database configuration file on the secondary server. The file should be in one of the following locations and should initially look similar to the one shown below.

   UNIX: `/var/tps/dba/nfmfcp.cfg`
   Windows: `C:\Program Files\TPS Systems\NFMServer\dba\nfmfcp.cfg`

```
TYPE=LOCAL
DATAPATH=/var/tps/dba/data
ADDR=
PORT=
```

This represents a configuration for a database found on the local file system. Change the `TYPE` from `LOCAL` to `REMOTE` and set the `ADDR` field to the IP address or hostname of the primary NFM server computer:

```
TYPE=REMOTE
DATAPATH=/var/tps/dba/data
ADDR=204.23.13.66
PORT=
```

- The `DATAPATH` field is now ignored since the `TYPE` field is set to `REMOTE`.
- The `ADDR` & `PORT` fields are now active. The `PORT` field may be left blank which will default to `16016`.

7. Restart the NFM server process on the secondary server.

   At this point you should now be able to login to the secondary server with the user interface and see the same content you would see if you were logged into the primary server. You will of course need to log in as a user that is defined to the primary server, as you should now be using the shared NFM database. Take note of any initial messages that are displayed while logging in as they may indicate a problem with the setup. No plans will run on this server until the following Step 8 is completed.

8. Create one or more new usergroups whose workload will be performed on this secondary server (you may change an existing usergroup to now utilize the new server). To perform this step, use the Usergroup menus under configuration (see "Usergroups" in Chapter 4, "Configuration"). Assign a server to a usergroup by setting the "Home Server" field in the "`Home Server`" tab in the `Usergroup Detail` screen to the appropriate server record. After this is done, the corresponding server should automatically perform any plans that are scheduled for this usergroup.

Backup Servers

Any existing server may be defined as a backup to another server. This can be done in the `Server Detail` screen (see section "Server," in Chapter 4, "Configuration" for the specific field settings). The following restrictions should be considered when setting up a backup server.

- In order to operate properly, a `backup server` must have the same access to needed resources as the server that it is a backup of. This includes any node connections, as well as any local files or programs that are involved with a plan's functions.

- A `backup server` will automatically begin running plans after the take over interval (defined in the server) has expired on a per plan instance basis. If the assigned server recovers, it may continue to perform plans even while some of the plans it missed may have been started by the backup.

- Because a `backup server` keeps the NFM database replicated, it will determine when to start a plan instance based on its own copy of the data. If the master database is unavailable, the backup will continue to operate on the locally cached data. This is ideal of course if the primary server has crashed and the master database is unavailable. However, if communications is lost between a server and its backup, it is possible that both servers will attempt to run the same plan instance. Care should be taken in the configuration of the take over interval, as well as the general design of the plans to minimize the impact of this.

- A server that is designated as a backup may still be a fully functioning server in its own right (it does not have to exist just to be a backup). Assuming that a server and its backup are generally doing unrelated work, the effects of having to do backup work on a server should simply be an increased amount of workload.

- A server that is a backup to another may also have its own backup; however, the workload from any given server will only be taken over by its immediate backup and not cascade to a backup of a backup. Two servers may be designated as backups to each other.

TPS®/NFM Training Guide

Section 3

SECURITY COMPONENTS

User

The NFM system maintains its own set of records that controls user's access to the NFM system. A user must sign on/off the NFM system and individual permissions are maintained that control the user's abilities to read/modify the different components of the NFM System.

The User screen allows creating, viewing or changing of individual user records to the NFM system. A new user may be typed in, or an existing user selected from the UserName field. Once selected, the various parameters associated with a given user are available with most fields being logically divided among five different folders.

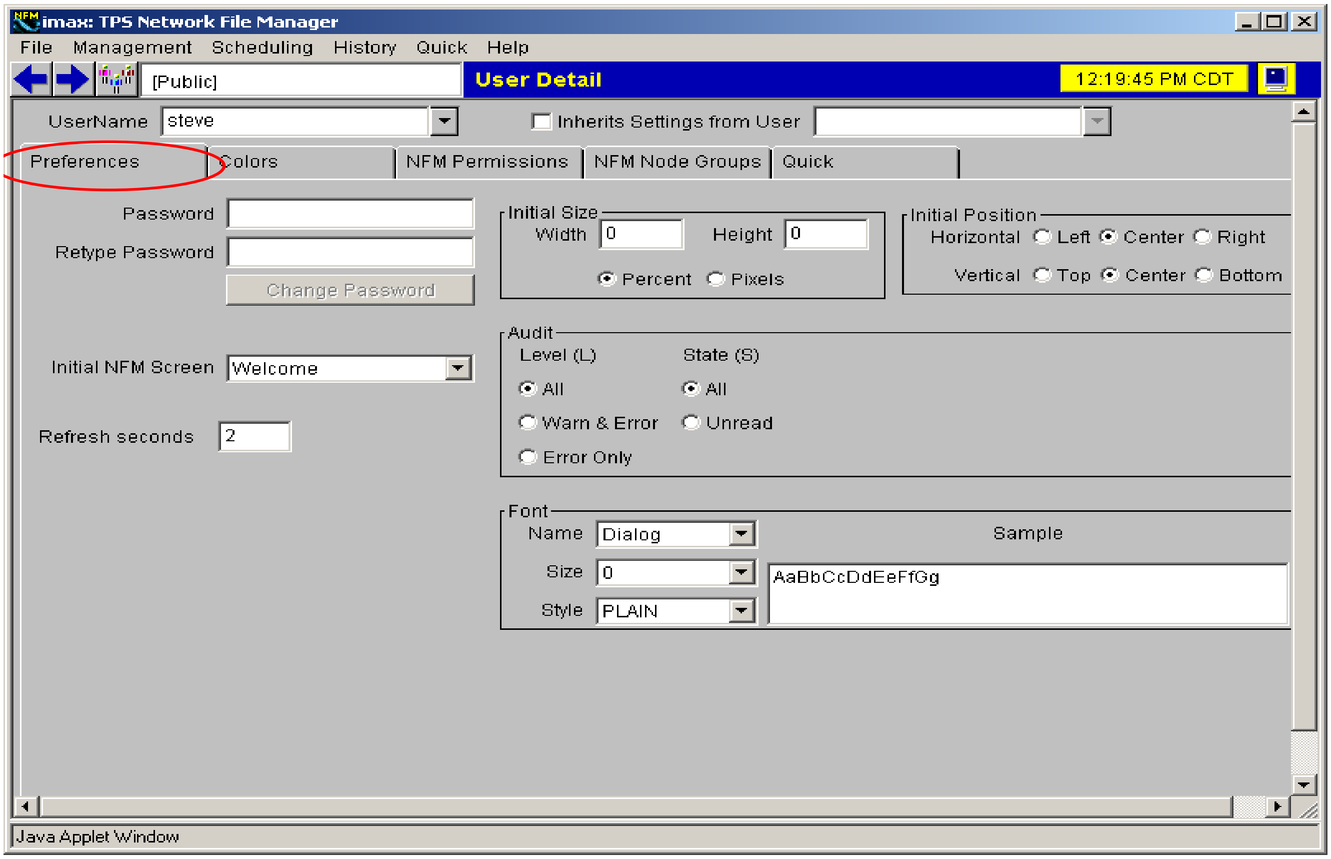
NOTE: **Only a user designated as an Administrator or a (Usergroup's) Sub-Administrator may look at a user summary screen and may modify other user's permissions. Non-administrators will have access to only the first two tabs ('Preferences' and 'Colors') of their own user record that will allow them to modify their password and their preferences.**

Management | User ▸ | User Detail

(to access User screen)

Preferences Tab

This tab includes initial settings when the user logs into the NFM system.



UserName: steve
☐ Inherits Settings from User

Password:
Retype Password:
Initial NFM Screen: Welcome
Refresh seconds: 2

Initial Size
Width: 0 Height: 0
● Percent ○ Pixels

Initial Position
Horizontal ○ Left ● Center ○ Right
Vertical ○ Top ● Center ○ Bottom

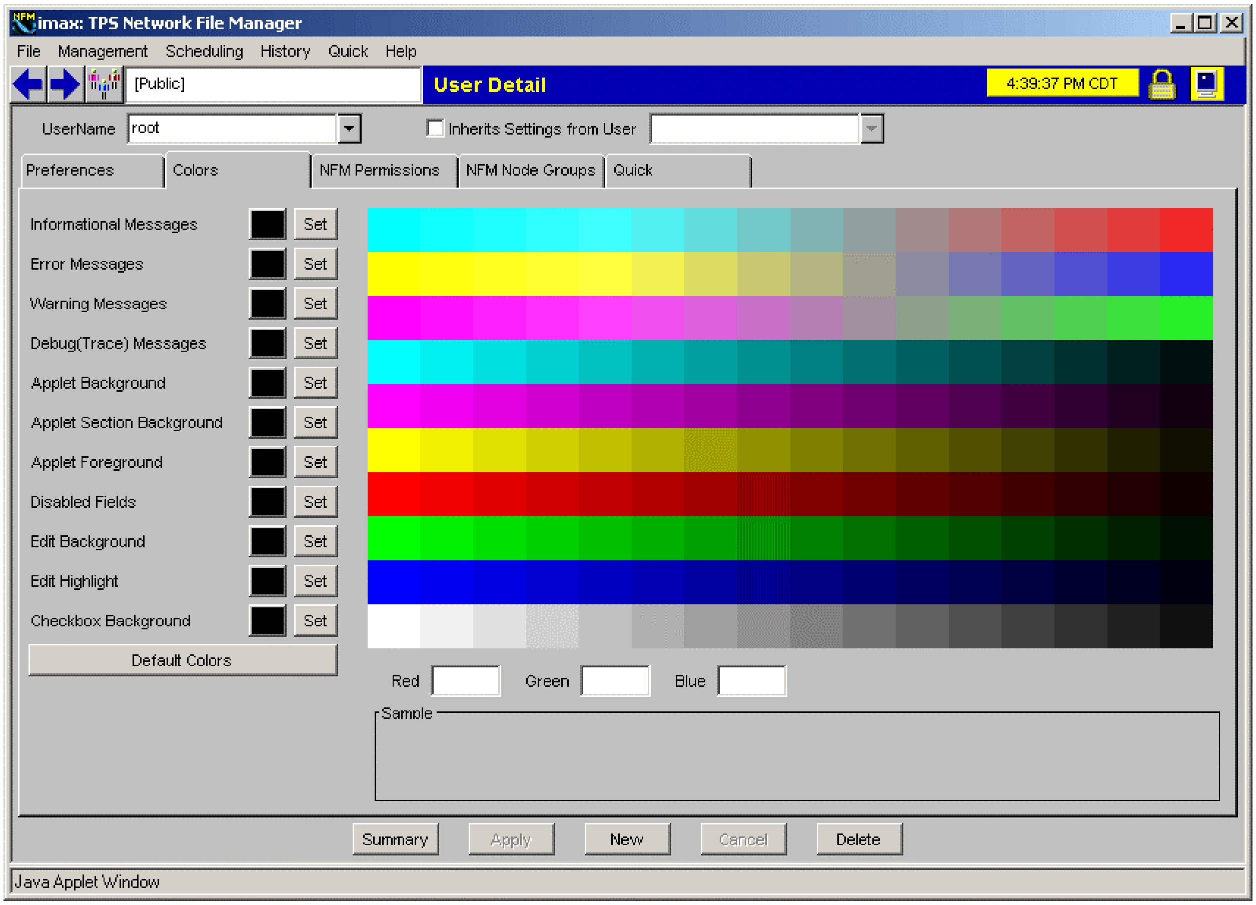
Audit
Level (L): ● All ○ Warn & Error ○ Error Only
State (S): ● All ○ Unread

Font
Name: Dialog
Size: 0
Style: PLAIN
Sample: AaBbCcDdEeFfGg

Field Name	Field Description
UserName	The username that a person will use to log into the NFM system. This field is limited to 16 characters.
Inherit Settings from User	This field allows an administrator to exactly duplicate all of the users settings from another user. This loads all the settings as a template with the exception of the password field.
Password / Retype Password	Password a user will use to login to the NFM system with. A user may change his own password, or an administrator may change any persons. This field is limited to 8 characters.
Initial NFM Screen	The screen that will be displayed after the user logs in.
Refresh seconds	This value defines the default screen refresh rate, in seconds, used while viewing auto-updating screens. These include screens such as the Plan Activity or Plan Monitor screens.
Audit	These selection boxes configure the default parameters used when viewing audit records in the History drop down menu. (see Plans and Plan Scheduler)

Colors Tab

The Colors tab maintains alternative color preferences for the user.



imax: TPS Network File Manager

[Public]

User Detail

4:39:37 PM CDT

UserName root

□ Inherits Settings from User

Informational Messages ■ Set

Error Messages ■ Set

Warning Messages ■ Set

Debug(Trace) Messages ■ Set

Applet Background ■ Set

Applet Section Background ■ Set

Applet Foreground ■ Set

Disabled Fields ■ Set

Edit Background ■ Set

Edit Highlight ■ Set

Checkbox Background ■ Set

Red

Green

Blue

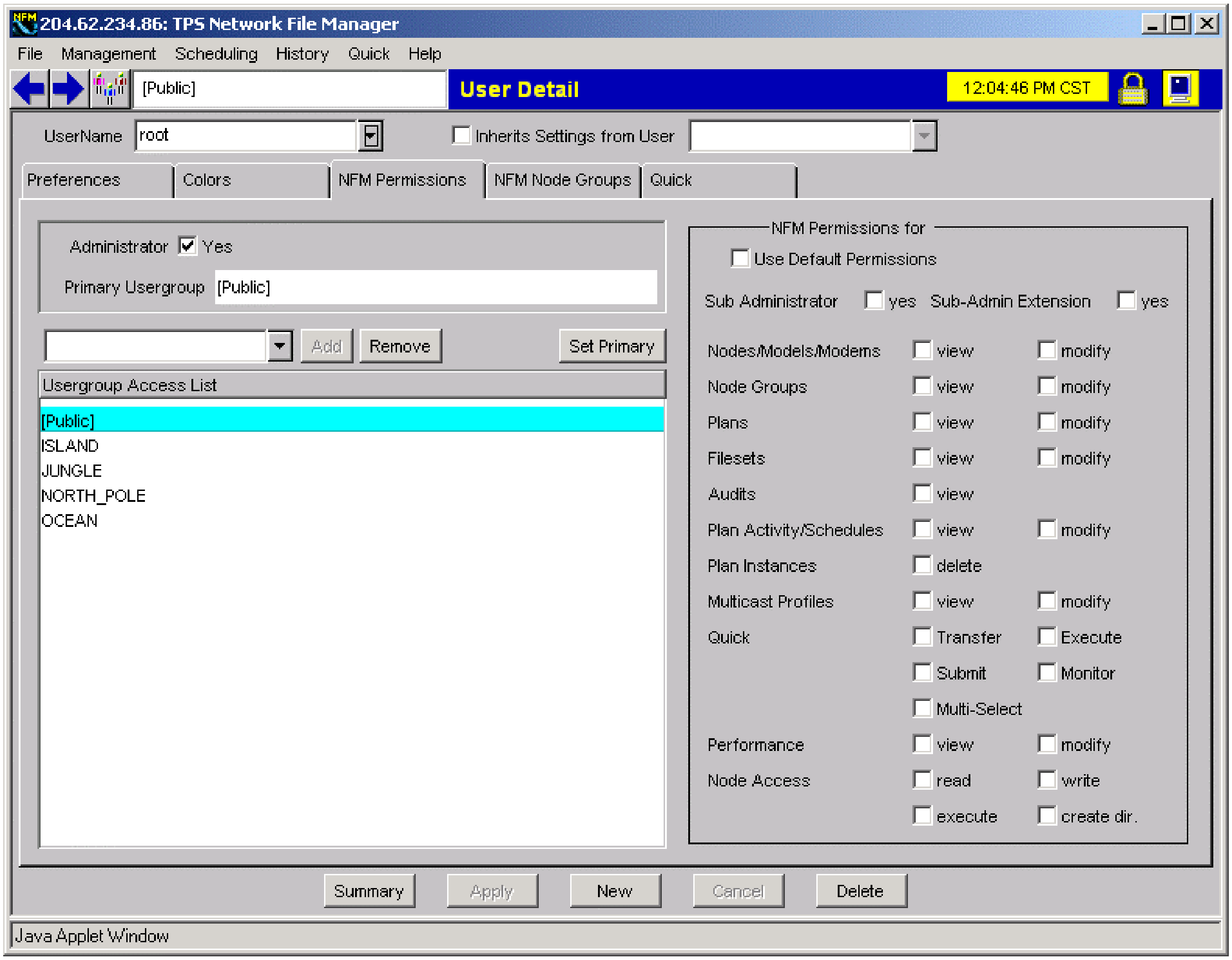
Sample

Java Applet Window

NFM Permissions Tab

(** Accessible only to administrators**)

The Permissions tab allows an administrator to control another user's access to usergroups in the system. If no usergroups have been added, at least one, "[Public]," will exist by default. For companies choosing not to divide users into usergroups, all users will belong to "[Public]".



204.62.234.86: TPS Network File Manager

[Public]

User Detail

12:04:46 PM CST

UserName root

□ Inherits Settings from User

Administrator ☑ Yes

Primary Usergroup [Public]

Usergroup Access List

[Public]
ISLAND
JUNGLE
NORTH_POLE
OCEAN

NFM Permissions for

□ Use Default Permissions

Sub Administrator □ yes Sub-Admin Extension □ yes

Nodes/Models/Modems	□ view	□ modify
Node Groups	□ view	□ modify
Plans	□ view	□ modify
Filesets	□ view	□ modify
Audits	□ view	
Plan Activity/Schedules	□ view	□ modify
Plan Instances	□ delete	
Multicast Profiles	□ view	□ modify
Quick	□ Transfer	□ Execute
	□ Submit	□ Monitor
	□ Multi-Select	
Performance	□ view	□ modify
Node Access	□ read	□ write
	□ execute	□ create dir.

Java Applet Window

Field / Button Name	Field Description
Administrator	This designates the user as an administrator, giving them full access to the NFM system. This overrides any remaining check boxes that may be unchecked. This user has full access to information in any other user's record.

Field / Button Name	Field Description
Primary Usergroup	Designates this user's primary usergroup. A primary usergroup serves two purposes. It is the default working location for a user when they log on to the system. Any configuration items they view or create will be in this usergroup (unless they switch to another group). Also it indicates that this users record may be viewed or modified by another user that is designated as a sub-administrator of this usergroup. Sub-administrators are described in more detail below.
Set Primary	Sets the item highlighted in Usergroup Access List as the Primary Usergroup. When the user is logged in, their current usergroup will be their primary.
Usergroup Access List	Contains the list of usergroups that this user has access to.
NFM Permissions for	**Use Default Permissions:** Forces the use of the default permissions assigned to this usergroup for the remaining checkboxes. These defaults are maintained separately in the Usergroup screen described in the next section. The actual permissions used will be either the Primary or Secondary permissions defined to the usergroup. The Primary permissions will be used if this usergroup is designated as the user's primary, otherwise the Secondary will be in effect. **Sub-Administrator:** Indicates that this user is a sub-administrator of this usergroup. A sub-administrator is similar to an administrator in the sense that they can access other user's settings, however this ability is confined only to other users in this usergroup (that is users who have this usergroup set as their primary one). Also, unlike an administrator a sub-administrator is limited to activities designated by the remaining checkboxes. Sub-administrators may alter permissions for users within the same usergroup and assign access to other usergroups for them, but they may not elevate a user's permissions to a level greater than their own. For this reason, a sub-administrator while working with another user's permissions may see checkboxes that are inaccessible or grayed out. **Sub-Admin Extension:** Enables a sub-administrator to view and set permissions for users that are not within their usergroup, but this is limited to the permissions that apply only to this usergroup. This allows them to invite outside users to this usergroup. An outside user would be one who does not have this usergroup designated as their primary one. Consider the following example: - Barry is an overall administrator.

Field / Button Name	Field Description
	- Jim is a sub-administrator of the usergroup SALES. This means that Jim has SALES set as his primary usergroup and has Sub-administrator checked on. - Joe is a regular user (not administrator or sub-administrator) who is also in SALES. - Jef is a sub-administrator of the usergroup ACCOUNTING and also has the extension box checked. Joe does not currently have access to ACCOUNTING. He could be granted access by any one of the following ways: - Barry could add ACCOUNTING to Joe's access list (he can do anything). - Because Jim is a sub-administrator over Joe, Jim could add ACCOUNTING to Joe's access list. He could only do this however if he (Jim) already had access to ACCOUNTING himself. - Jef could add ACCOUNTING to Joe's access list. Because he has the extension flag on, he is allowed to 'invite' others into his usergroup. **Nodes/Models/Modems:** View gives authority to view these configuration items only. Modify allows adding, update or deleting items. **Node Groups:** View gives authority to view node groups. Modify allows adding, updating or deleting node groups. **Plans:** View gives authority to view plans. Modify allows adding, updating or deleting plans. **Filesets:** View gives authority to view filesets. Modify allows adding, updating or deleting filesets. **Audits:** Audit records are generated automatically and thus a single check box, view, gives a user permission to view audit records. **Plan Activity/Schedules:** View gives a user the authority to monitor a submitted plans status. Modify gives them the ability to submit plans, change submitted plans (cancel, hold, restart, etc.), and the ability to make changes to Schedule tables.

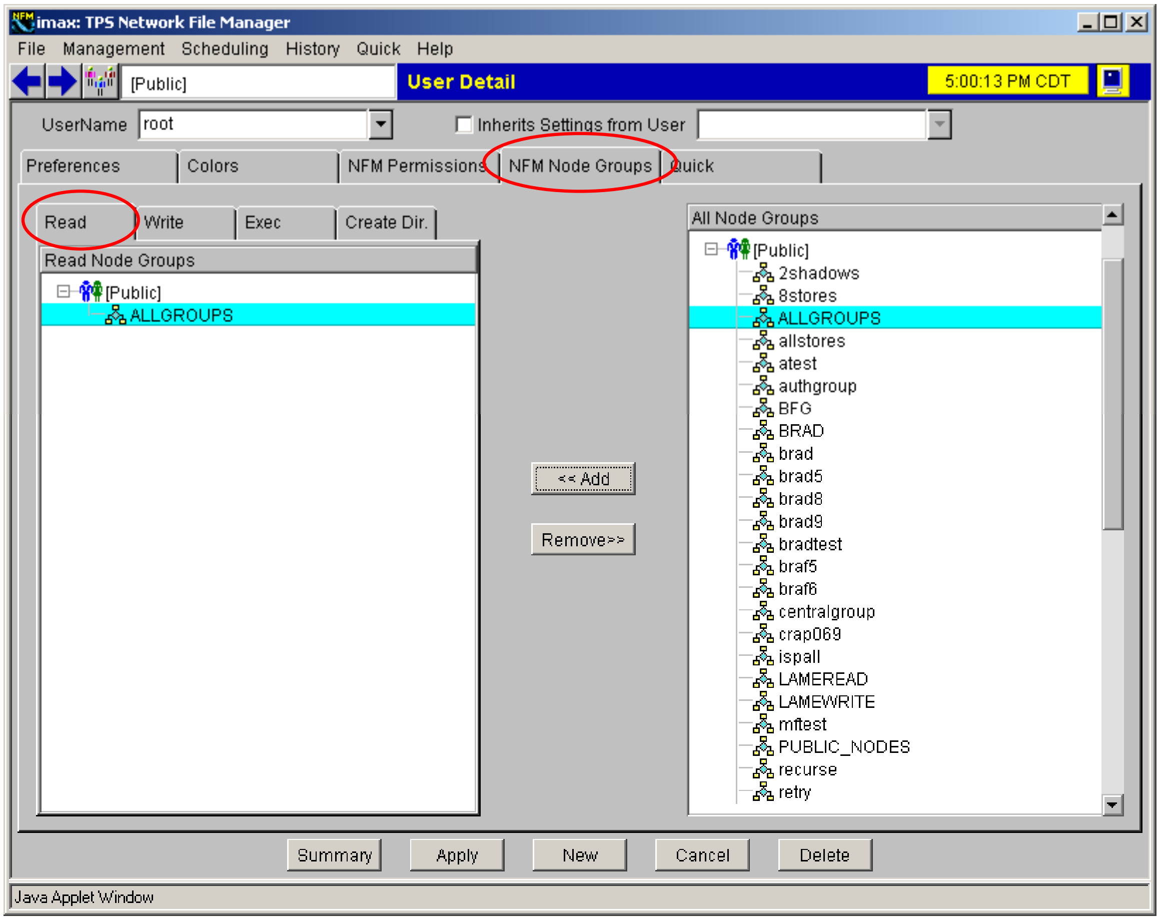
Field / Button Name	Field Description
	Plan Instances: Delete allows the user to delete plan instances from the Plan Activity screen. **Multicast Profiles:** View gives authority to view multicast profiles. Modify allows adding, updating or deleting multicast profiles. **Quick Transfer:** Gives authority to use the Quick Transfer function. **Quick Execute:** Gives authority to use the Quick Execute function. **Quick Submit:** Gives authority to use the Quick Plan type functions from the Quick List screen. **Quick Monitor:** Gives authority to monitor the plans that are performing the different Quick function types. **Quick Multi-Select:** Gives authority to select multiple source or target nodes during a Quick Transfer or Quick Execute. **Performance:** View gives authority to view performance extracts and graphs. Modify allows adding, updating or deleting performance graphs. **Node Access:** The four different checkboxes (read, write, execute & create dir) may be used to grant or deny the implied type of access for the user to all nodes belonging to the currently selected usergroup. If it is necessary to assign different levels of access on a per-node basis, use the NFM Node Groups.

NFM Node Groups Tab

(** Accessible only to administrators**)

The NFM Node Groups tab makes it possible to give / restrict specific users individual read, write, execute, and directory creation during plan execution.. This can also be used to restrict the browse node capability used in Filesets.

Since groups may contain other groups, it should only be necessary to maintain a single read and write group for each class of users. If multiple users share the same access groups, then making a new node accessible to multiple users would simply involve adding that node to the appropriate group.



[Public]

5:00:13 PM CDT

UserName root

□ Inherits Settings from User

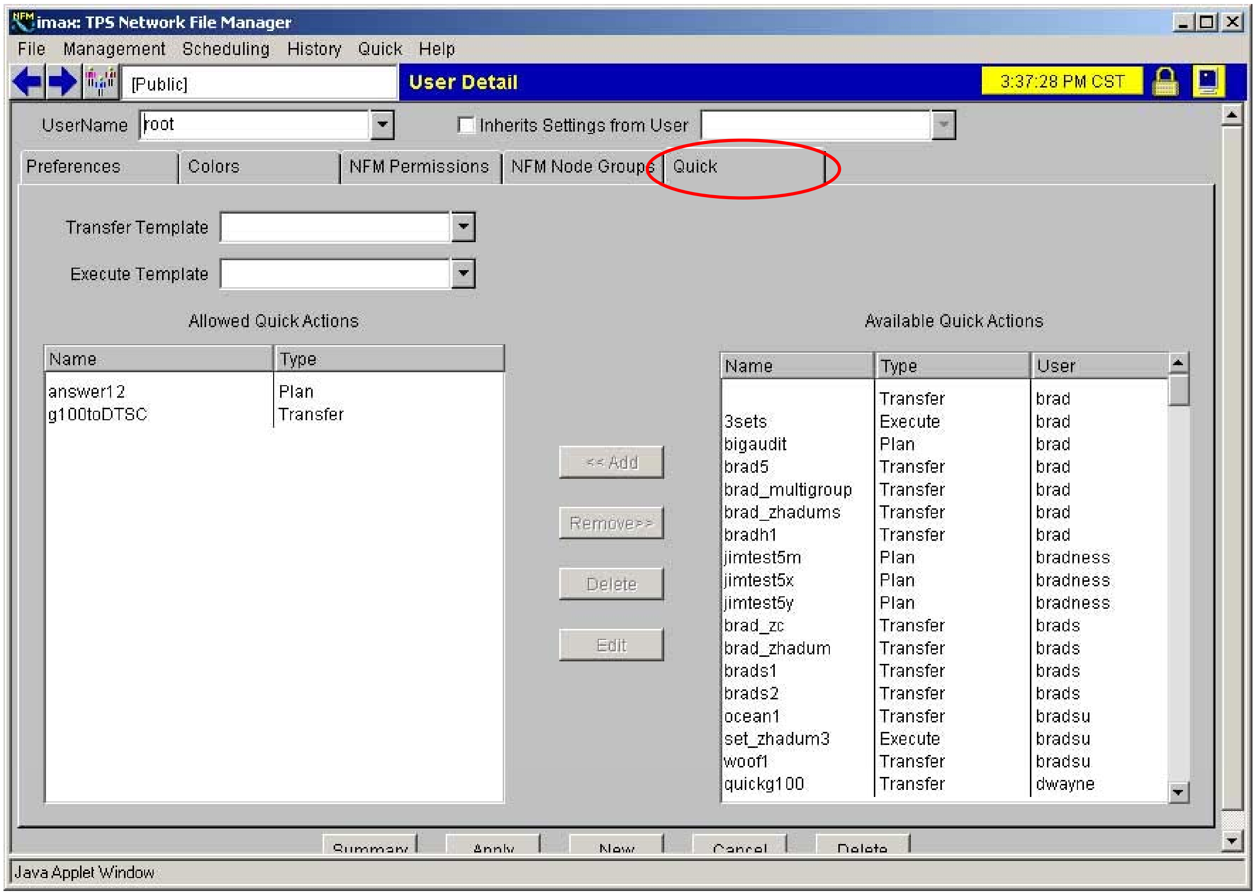
Read Node Groups
- [Public]
  - ALLGROUPS

All Node Groups
- [Public]
  - 2shadows
  - 8stores
  - ALLGROUPS
  - allstores
  - atest
  - authgroup
  - BFG
  - BRAD
  - brad
  - brad5
  - brad8
  - brad9
  - bradtest
  - braf5
  - braf6
  - centralgroup
  - crap069
  - ispall
  - LAMEREAD
  - LAMEWRITE
  - mftest
  - PUBLIC_NODES
  - recurse
  - retry

Quick Tab

(** Accessible only to administrators**)

The Quick tab grants access to certain types of Quick methods accessible by the user under the Quick menu. (What are Quick functions?)



UserName root

□ Inherits Settings from User

Transfer Template

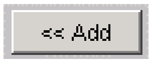
Execute Template

Allowed Quick Actions

Name	Type
answer12	Plan
g100toDTSC	Transfer

Available Quick Actions

Name	Type	User
	Transfer	brad
3sets	Execute	brad
bigaudit	Plan	brad
brad5	Transfer	brad
brad_multigroup	Transfer	brad
brad_zhadums	Transfer	brad
bradh1	Transfer	brad
jimtest5m	Plan	bradness
jimtest5x	Plan	bradness
jimtest5y	Plan	bradness
brad_zc	Transfer	brads
brad_zhadum	Transfer	brads
brads1	Transfer	brads
brads2	Transfer	brads
ocean1	Transfer	bradsu
set_zhadum3	Execute	bradsu
woof1	Transfer	bradsu
quickg100	Transfer	dwayne

Field / Button Name	Field Description
Transfer Template	This designates a default template plan to be used by this user for each Quick Transfer that they perform (see note below).
Execute Template	This designates a default template plan to be used by this user for each Quick Execute that they perform. NOTE: On the System Settings screen, the Quick folder allows setting of a system wide Transfer and Execute template. If these are set, it is not necessary to set the above two fields unless you wish to override the system defaults.
Allowed Quick Actions	This box on the right of the screen, lists all of the Quick functions (for all users) that are available to be added to the current user. These are all the Quick functions on the system and are not available to this user until added. The sub-fields of each function include: **Name** The Quick Functions assigned name. This name does not have to be unique across the system, only for a given user. Multiple users may have separate Quick Functions with the same name. **Type** Type of function. Either Transfer, Execute or Plan. **User** The user name to which this Quick Function is currently assigned to. This list will initially show all quick functions of type Plan that will in fact be a list of all plans defined to NFM. This is because any plan can be assigned as a Quick plan to individual users. The list will not show any Transfer or Execute type functions until they are first created. For an administrator to create and then assign the Transfer and Execute types of Quick functions they should first go directly to the Quick menu and create appropriate entries on the Transfer and Execute screens (as discussed in the previous section). After returning to the Quick Folder, these entries should now be available in this list.
Available Quick Actions	This box on the left of the screen lists all of the Quick functions that have already been added to the current user and are available for them to use. These entries may have previously been added using this folder, or may have been created by the individual user from the Quick menu screens directly.
	Add item(s) that are highlighted in the Available Quick Actions list to the current user.

Field / Button Name	Field Description
Remove>>	Remove the highlighted item(s) from the Available Quick Actions list.
Delete	Delete the highlighted item(s) from the Available Quick Actions.
Edit	After highlighting an entry in the Available Quick Actions list, this button will take you directly to the appropriate Quick menu screen (based on the function type) where the function may now be edited or launched.

Usergroups

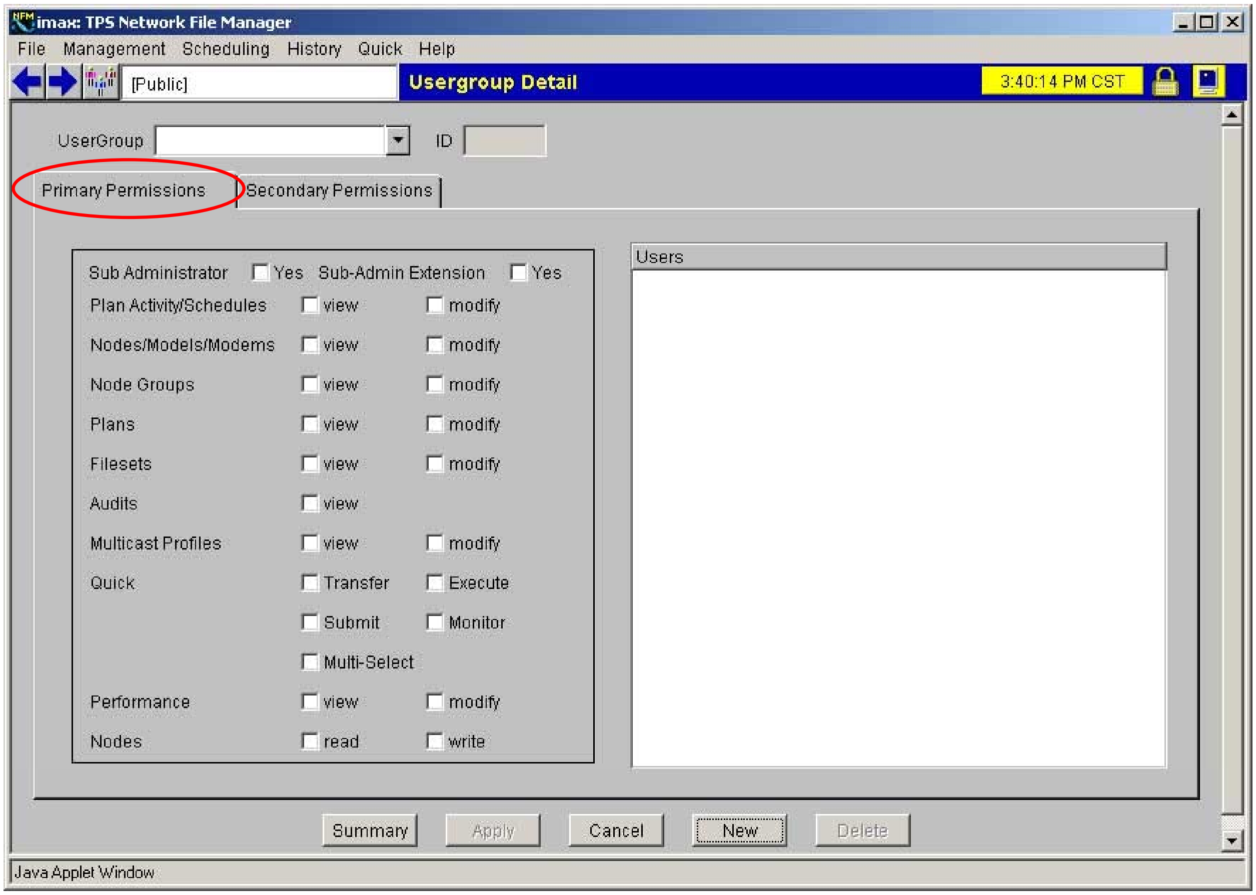
Usergroups provide the customer with the means to partition configuration data (plans, node, filesets, etc.) into separate logical groups, thus making the data easier to manage and allowing the customer to assign the individual users different access rights (or permissions) to the items that are placed in these groups. A usergroup may represent a department, a single user, or any other logical grouping that the customer cares to make. A user can belong to multiple Usergroups however they can only have one primary Usergroup, which is the default Usergroup when the user logs in. If you decide not to use Usergroups, all users will belong to the default "[Public]" Usergroup. The access of any given user to any given usergroup (and its items) may be individually configured.

Management | Usergroup ▸ | Usergroup Detail

(to access Usergroup screen)

Primary Permission Tab

Primary Permissions tab displays the default user's permissions if they have that usergroup set as their primary usergroup.



UserGroup [ ] ID [ ]

Primary Permissions | Secondary Permissions

Sub Administrator □ Yes Sub-Admin Extension □ Yes

Plan Activity/Schedules	□ view	□ modify
Nodes/Models/Modems	□ view	□ modify
Node Groups	□ view	□ modify
Plans	□ view	□ modify
Filesets	□ view	□ modify
Audits	□ view	
Multicast Profiles	□ view	□ modify
Quick	□ Transfer	□ Execute
	□ Submit	□ Monitor
	□ Multi-Select	
Performance	□ view	□ modify
Nodes	□ read	□ write

Users

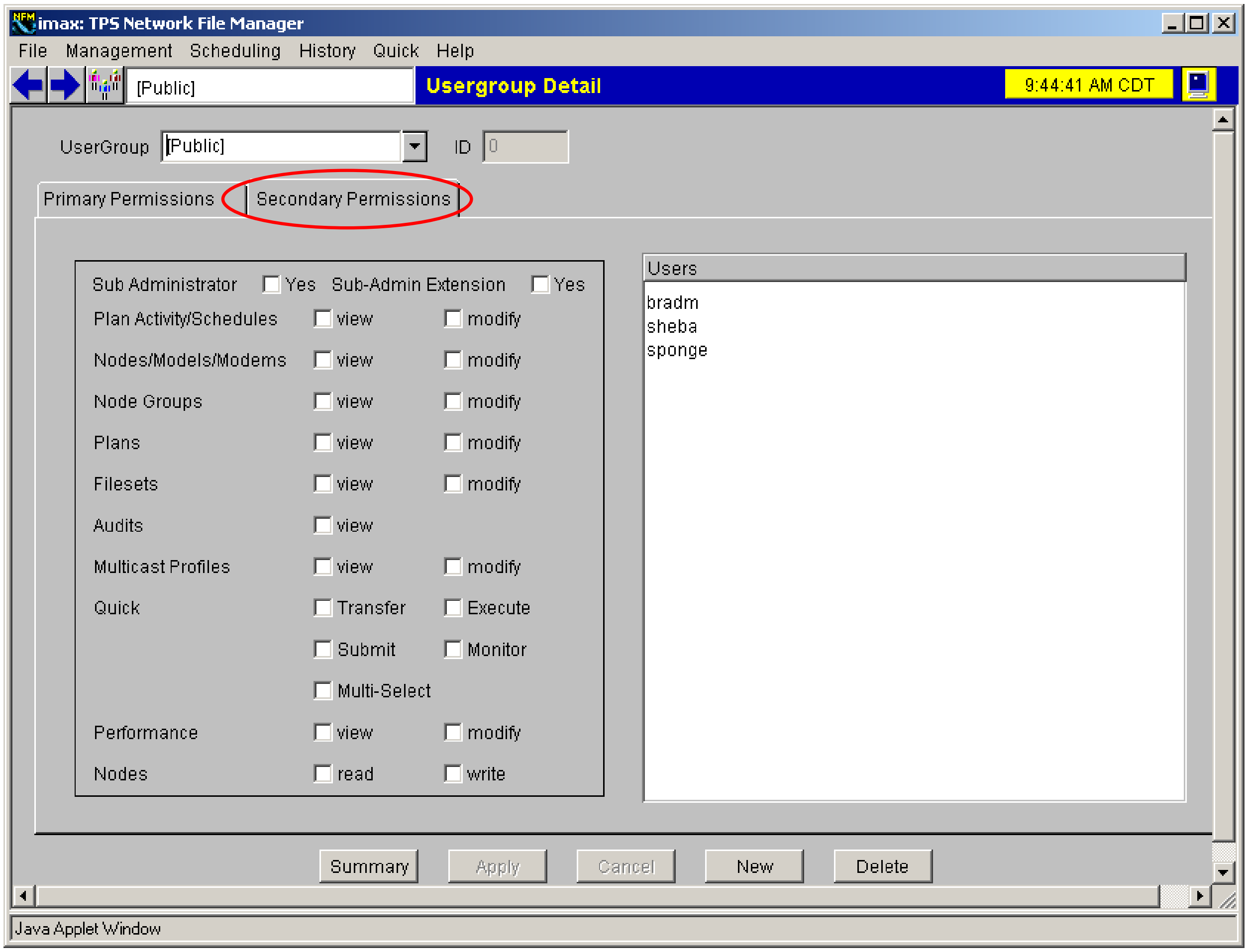
Field Name	Field Description
UserGroup	Usergroup name
ID	A unique identifier (ID) associated with this Usergroup. This is an non-editable field and used in the NFM database.

(See User NFM Permissions Tab section for description of fields)

The default permissions defined here for each usergroup will be available in the following way: If this usergroup is added to a user's access list in the User Detail screen, then by default, the secondary permissions here will be in effect for that user's access to this usergroup. If this usergroup is set as that user's primary usergroup, then by default, the primary permissions here will be in effect.

Secondary Permissions Tab

Secondary Permissions tab displays the default user's permissions for any usergroup not set as their primary usergroup.



UserGroup [Public] ID 0

Sub Administrator	□ Yes	Sub-Admin Extension □ Yes
Plan Activity/Schedules	□ view	□ modify
Nodes/Models/Modems	□ view	□ modify
Node Groups	□ view	□ modify
Plans	□ view	□ modify
Filesets	□ view	□ modify
Audits	□ view	
Multicast Profiles	□ view	□ modify
Quick	□ Transfer	□ Execute
	□ Submit	□ Monitor
	□ Multi-Select	
Performance	□ view	□ modify
Nodes	□ read	□ write

Users
bradm
sheba
sponge

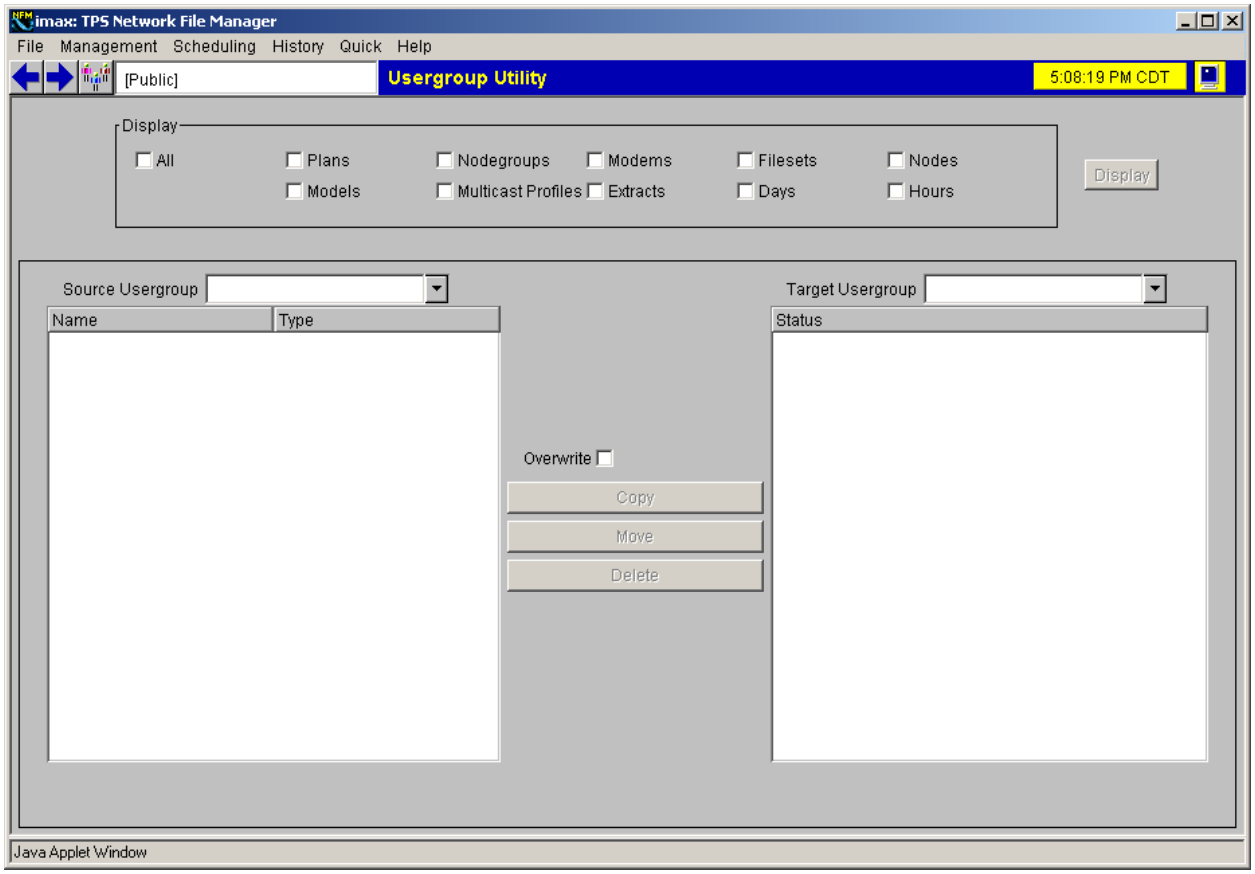
(See User NFM Permissions Tab section for description of fields)

Usergroup Utility

The Usergroup Utility screen is used to copy, move and delete items (plans, nodes, etc.) that exist within the various Usergroups.

Management | User ▸ | Usergroup Utility

(to access Usergroup Utility screen)



Field Name	Field Description
Display	A search criteria used for displaying items in the "Source Usergroup". Once you have selected the search criteria click on Display button.
Source Usergroup	Usergroup you want to copy, move or delete.
Target Usergroup	Usergroup you want to copy or move to.
Overwrite	This will overwrite an existing item on the Target Usergroup.

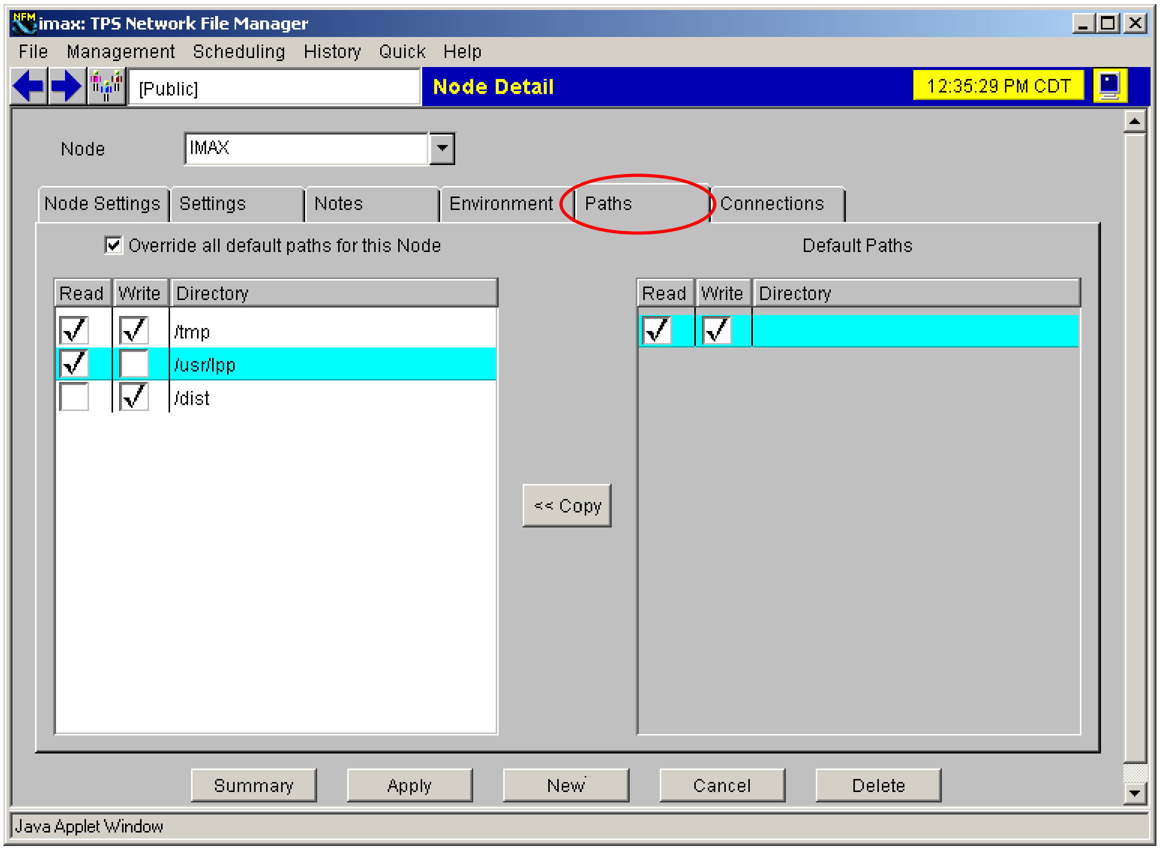
Path Permissions

As we have seen users may be granted or denied access to spefic nodes. In addition to this the NFM configuration allows for a node to be setup so individual directories (or paths) in the file system may be granted or denied access to them. Paths are located in the Model and Node screens under the Paths tab. The Paths tab maintains a list of path permissions associated with this node. Path permissions allow NFM to restrict or permit file transfer access to the specified directories on any nodes of this model type. If a directory is enabled or disabled it will affect any files within the directory tree. These entries can overlap to allow setting policy on a directory in one entry, and then changing policy for one of its subdirectories in a subsequent entry (see below). An empty string implies the root directory, and thus, dictates the default access if a filename does not match any subsequent entries.

If the "Override all default paths for this Node" is checked, then directories listed on the left side of the screen will override the same directory permissions on the right "Default Paths".

Paths Tab

The Paths tab allows for the defining of path permissions that take precedence over the ones defined in the node's model.



[Public]

12:35:29 PM CDT

Node: IMAX

☑ Override all default paths for this Node

Read	Write	Directory
✓	✓	/tmp
✓	☐	/usr/lpp
☐	✓	/dist

Default Paths

Read	Write	Directory
✓	✓	

(*The Paths tab is accessible from both the Node and Model screen*)

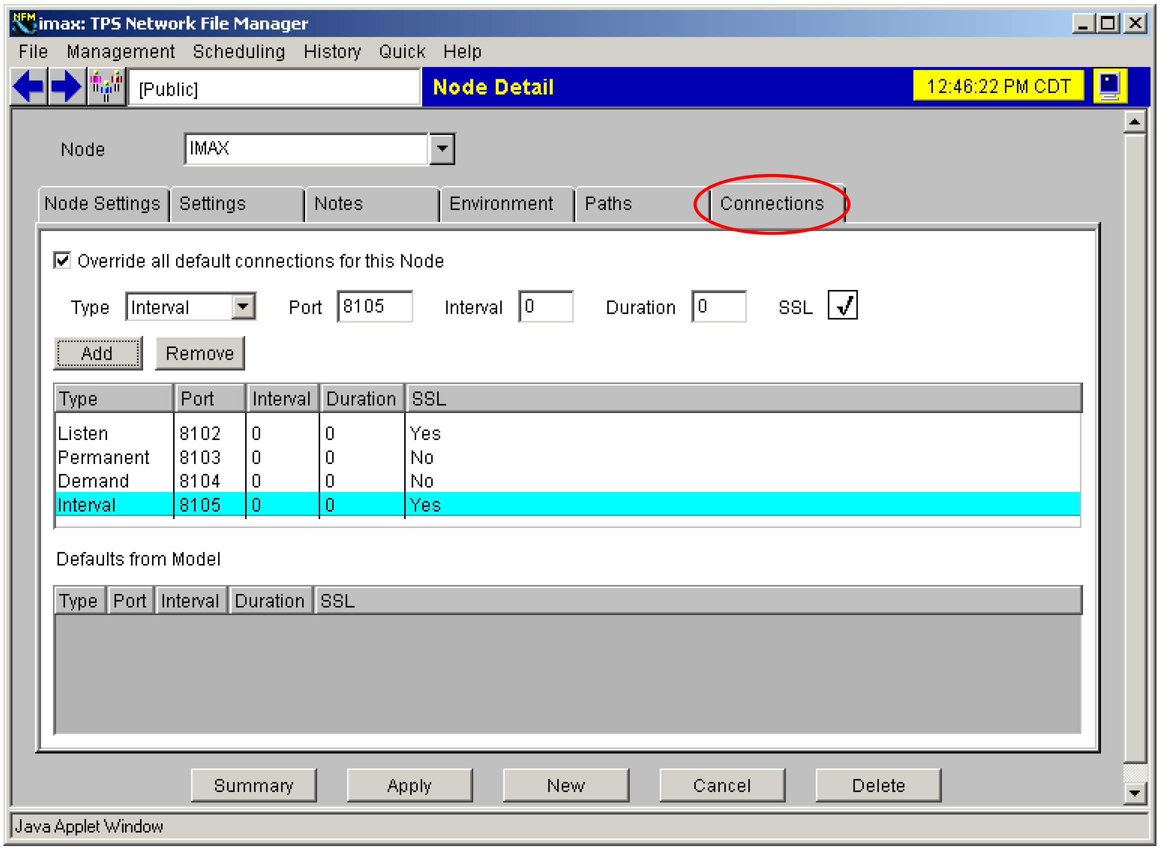
Encrypted Connections

NFM provides two forms of encryption for socket connections among NFM components, including NFM Server to NFM Client connections, and NFM Client to NFM Client (peer to peer) connections involving file transfers and program execution. The first type of encryption the NFM system uses a native type of encryption to protect sensitive data sent over the network (file transfer data, etc.). This is a Two-Fish Algorithm. The second type of encryption is the standard Secure Sockets Layer (SSL) encryption implements OpenSSL Version 2.3 and is more suitable for use over insecure networks like the Internet. Encryption is not available for non-NFM clients like FTP or NVDM nodes.

SSL Connections

Connections Tab

The Connections tab allows the user to customize socket type nodes (NFM Client).



Node: IMAX

☑ Override all default connections for this Node

Type: Interval   Port: 8105   Interval: 0   Duration: 0   SSL: ☑

Type	Port	Interval	Duration	SSL
Listen	8102	0	0	Yes
Permanent	8103	0	0	No
Demand	8104	0	0	No
Interval	8105	0	0	Yes

Defaults from Model

Type	Port	Interval	Duration	SSL

(The *Connections tab is accessible from the Node and Model screen*)

It is not usually necessary to add connection records. NFM uses default connections for sockets type nodes as follows:

A non-SSL type node listens for a socket connection on the default port 8008.
An SSL type node listens for an SSL encrypted socket connection on the default port 8009.

NFM Clients, by default use a connection 'Type' of Listen, meaning the connection is initiated from the NFM Server to the node. Sometimes, you might need to use another type of connection, a different port or use SSL. Theses connections are:

Field Name	Field Description
Type	Choices include: **Listen** The node will listen for a socket connection on the specified port (Used when you want to override the default 8008). **Permanent** The node will make a permanent socket connection to the NFM server on the specified port when the NFM client program is started. **Demand** ** Not currently implemented.** **Interval** ** Not currently implemented.**
Port	Specifies which TCP/IP port to listen or connect to.
Interval	** This field is currently not yet supported. **
Duration	** This field is currently not yet supported. **
SSL	Indicates an SSL type connection.

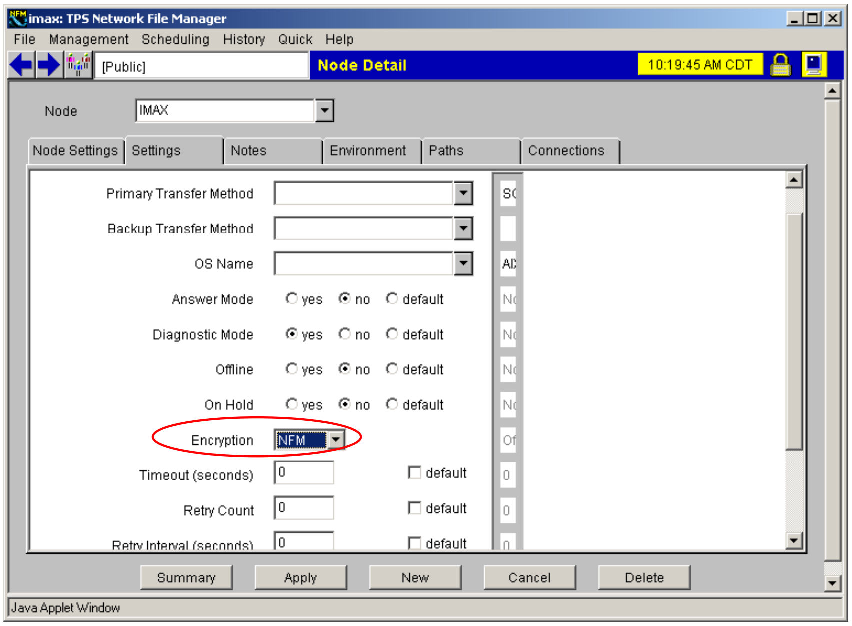
If the "Override all default connections for this Node" is checked, then connections listed on the top will override the same connections on the bottom "Default from Model".

If you wish to use, add, or edit a connections entry, you will also need to configure or change that node's configuration file to match. See NFM Client Configuration File for more information

Encryption

Encrypting File Transfers

This option is only used when transferring from an NFM Client to an NFM Client. NFM currently supports the following encryption types: NFM (a TwoFish algorithm) and SSL. This option can be set in the Node / Model screen or globally in the System Settings:



Node IMAX

Primary Transfer Method

Backup Transfer Method

OS Name

Answer Mode ○ yes ● no ○ default

Diagnostic Mode ● yes ○ no ○ default

Offline ○ yes ● no ○ default

On Hold ○ yes ● no ○ default

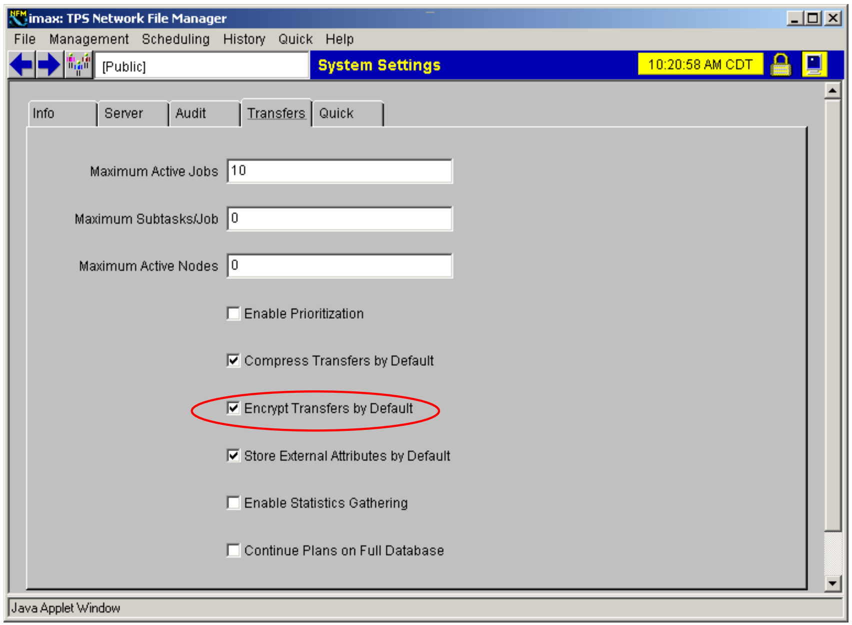
Encryption NFM

Timeout (seconds) 0 □ default

Retry Count 0 □ default

Retry Interval (seconds) 0 □ default

(*Encryption turned on from either Node or Model's Setting tab*)



Maximum Active Jobs 10

Maximum Subtasks/Job 0

Maximum Active Nodes 0

□ Enable Prioritization

☑ Compress Transfers by Default

☑ Encrypt Transfers by Default

☑ Store External Attributes by Default

□ Enable Statistics Gathering

□ Continue Plans on Full Database

(*Encryption turned on from the System Settings menu option*)

Encrypting the NFM login session

In addition to encryption of file transfers, NFM users have their NFM GUI session to the NFM Server encrypted as well.

User Exits

User exits are customer written programs that can be invoked by an NFM client to perform security checks or other customized tasks before or after doing particular actions (such as sending a file, receiving a file, etc). Information regarding the current action including an account user name and password are provided to the user exit. The user exit can then process the information as necessary, and if it chooses, disallow the action from occurring.

Program interface

The user exit program, if configured, is loaded once when the NFM client is started. It stays loaded and is called at the following key points by the NFM client:

1. Immediately after the client program is loaded. This will occur only once (initialization request).
2. Before a file is opened for sending (pre-read request).
3. After a file is sent (post read request).
4. Before a file is opened for receiving (pre write request).
5. After a file is received (post write request).
6. Before a program execution (pre exec request).
7. After a program execution (post exec request).

A separate area of memory (or control block) is associated with each request and is used to relay information back and forth between the NFM client and the user exit. The following section lists the available fields in the control block and describes their usage. For the exact layout of this control block, including field sizes and offsets, please refer to the `nfmcexit.h` file shipped with the NFM client.

User exit control block

(Note: Fields that are described as text are null terminated strings that may be empty.)

LENGTH (numeric, read only) The size in bytes of the control block. The size of the control block may increase in subsequent versions, however the size and offsets of the existing fields will not change.

ACTION (numeric, read only) A number representing which action is about to occur, or has already occurred. Set to one of the following:

0 Initialization. This is a single call made to the user exit after it is first loaded. It is at this point in the user exit that any one-time initialization should be performed.

1 Read request. A file open associated with sending a source file from this node.

2 Write request. A file open associated with receiving a target file to this node.

4 Exec request. Associated with executing a program on this node.

FINISHED (numeric, read only) Indicates whether or not the action has occurred. Set to one of the following:

0 Indicates that the action is about to occur. It is at this time that the exit may return a zero value to deny the action from occurring (see return code below).

1 Indicates that the action has already occurred.

USERID (text, read only) This field contains the optional Account User Name field from the Node Access Point (or NAP) record used to refer to this node in the NFM plan.

USERPASSWORD (text, read only) This field contains the optional Account Password field from the Node Access Point (or NAP) definition used to refer to this node in the NFM plan.

FILENAME (text, read only) This field contains a file name if the ACTION field is set to 1 (read) or 2 (write).

MESSAGE (text, modifiable) This field may be modified by the user exit to contain a readable error message that will be displayed in an audit message when the action that is about to be performed is denied. This only occurs when the exit returns a zero value.

REQUESTNUMBER (numeric, read only) This field contains a number unique to this request. This number will be the same for the before and after calls made from the same request.

NFMERROR (numeric, read only) This field may contain an NFM error number during an after call (FINISHED=1) from a specific action that failed.

USER AREA (undefined, 32 byte, modifiable) This area of the control block is available for the user exit to use for its own needs. Because the entire control block is unique per request, this area offers the user exit a place to pass information between the before and after calls for a given request. Additionally, this user area is copied from the user area of the initialization call to the user exit. This allows the user exit to perform one-time initialization (when ACTION is set to zero), and then set values into the user area that will be available for later calls to access.

User exit return value

During a call prior to a specific request (when `FINISHED=0`), the user exit may return with a zero to indicate that the request is to be denied. If this happens, the request will not be performed, and the NFM server will generate an audit error message indicating so. If a message has been copied to the MESSAGE field, it will be displayed as part of the audit error allowing the user exit to display a specific reason for denying the requested action..

The value returned from the user exit during the initialization call (`ACTION=0`), or during any of the post action calls (`FINISHED=1`) has no effect.

Restrictions

The operating environment of the user exit is restrictive. The user exit is called by the actual NFM client threads that are performing the operations. Because of this, great care must be taken to preserve the integrity of the calling process and threads. The following rules must be adhered to:

1. The user exit must be dynamically loadable, supporting the following loading conventions:
   `dlopen()` on UNIX
   `LoadLibrary()` on Windows
   `fetch()` on OS/390 z/OS

2. The program must be fully reentrant. It may be called simultaneously from multiple threads. It must perform any necessary locking/synchronization while accessing outside resources.

3. It must operate fast enough to prevent performance problems.

4. It must not have any “leaks” (memory, file handles, or other system resources).

5. It must not change the security context (effective user) or any other attribute of the calling thread/process. This could negatively effect or disable the NFM client.

6. It cannot cause/throw any exceptions that will cause thread/process termination. This WILL disable the NFM client.

Sample program

The following files are shipped with the NFM client and can be found in the products base directory:

`nfmcexit.h` This include file contains definitions suitable for writing a user exit program for C or C++. Included is a structure definition for the user interface control block described above along with some useful defines.

`nfmpqc.c` This is a sample user exit program written in C. It has been compiled and tested on the various platforms that the NFM client runs on. This particular program simply examines the account user name provided, and if it is equal to "lame," it denies each of the available actions described in the previous section. This program may be used as a starting point for developing a user exit.

Configuring the user exit

Once the user exit has been built and placed in the appropriate location for loading (this process will vary by platform), the following steps should be done to enable its use:

1. Edit the NFM client configuration file, to include a line similar to the following one: (See "Encryption and connection options" in Chapter 6, "Advanced Functions" for a general description of the client configuration file.)

```
USEREXIT=PERMISSION, LIB=libnfmpqc.o, PARM1=NFMPQC
```

The field descriptions are:

`USEREXIT=PERMISSION` This indicates a permission query type exit. This is the only valid setting at this time.

`LIB=[loadable object name]` This contains the name of the loadable program object. This will be different based on the platform as follows:

`libnfmpqc.o`	For UNIX this represents a shared object module in `/usr/lib` or elsewhere in the LIBPATH.
`nfmpqc.dll`	For Windows this refers to a DLL.
`nfmpqc`	For OS/390 this refers to a load module.

NOTE: At this time the 4690 and VOS implementations of the NFM client do not support user exits.

`PARM1=NFMPQC` This must match the actual name used for the entry point into the user exit.

In order to use the `USERID` and `USERPASSWORD` fields in the user exit, a Node Access Point (or NAP) must be configured on the NFM server, and subsequently used from within a plan to refer to this node for action. When this occurs, the NAP's Account User Name and Account Password values will be available in these two fields in the control block, when the user exit runs.

TPS®/NFM Training Guide

Section 4

ADMINISTRATIVE FUNCTIONS

NFM Server Components

Installing the NFM Server package (Part 1 of 2):

(see 'Installing the TPS® Command Service package' for Part 2 of 2)

The NFM Server component can run on several different platforms including AIX, Windows and Linux. Complete installation of the NFM Server requires two pieces (each shipped separately) be installed, the NFM Server and TPS® Command Service (`Cserv`). Installing TPS®/NFM Server & TPS® Command Service should be installed by '`root`' on UNIX platforms and '`Administrator`' on Windows.

OS	Install Command	Directories

AIX	`Installp –acFd` *`<filename>`* `all`	`/usr/lpp/tps/nfm/ (product directory)`
		`/var/tps/ (working directory)`

Linux	`tar xvf` *`<filename>`*
	`installp_tpsnfm`
		`/opt/tps/nfm/ (product directory)`
		`/var/tps/ (working directory)`

Windows	*`<Double clickable install executable>`*	`C:\Program Files\TPS Systems\NFMServer\`
		`(product & working directory)`

NFM Server files:

<product directory>

`bin`	NFM Server binaries / executables
`gui`	GUI files to be copied into the Web Server directory

`<`*`working directory`*`>/nfm/`

`attachments`	Custom FTP & NVDM scripts used by NFM
`commands`	NFM Remote Command Line scripts
`logs`	NFM Server logs
`ssl`	SSL related files
`tmp`	Temporary NFM files
`tracking`	Used by NFM to track and resume plans

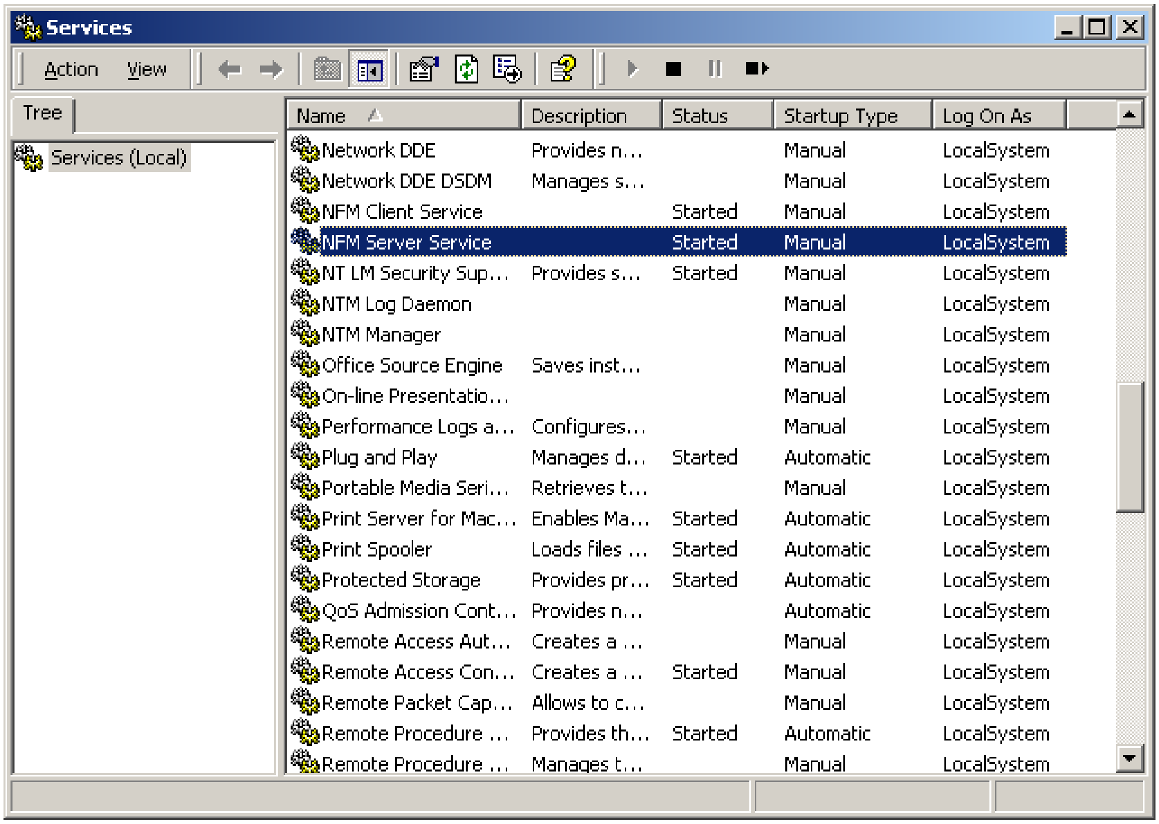
`<`*`working directory`*`>/dba/`

`data`	Stores all NFM databases
`init`	Stores common databases shared by various TPS® products

Line added to startup script / service:

AIX	`(/etc/inittab):`	`nfmserv:2:respawn:/usr/lpp/tps/nfm/bin/nfmserv`
Linux	`(/etc/inittab):`	`nfv:2345:respawn:/opt/tps/nfm/bin/nfmserv`

Windows



Name	Description	Status	Startup Type	Log On As
Network DDE	Provides n...		Manual	LocalSystem
Network DDE DSDM	Manages s...		Manual	LocalSystem
NFM Client Service		Started	Manual	LocalSystem
NFM Server Service		Started	Manual	LocalSystem
NT LM Security Sup...	Provides s...	Started	Manual	LocalSystem
NTM Log Daemon			Manual	LocalSystem
NTM Manager			Manual	LocalSystem
Office Source Engine	Saves inst...		Manual	LocalSystem
On-line Presentatio...			Manual	LocalSystem
Performance Logs a...	Configures...		Manual	LocalSystem
Plug and Play	Manages d...	Started	Automatic	LocalSystem
Portable Media Seri...	Retrieves t...		Manual	LocalSystem
Print Server for Mac...	Enables Ma...	Started	Automatic	LocalSystem
Print Spooler	Loads files ...	Started	Automatic	LocalSystem
Protected Storage	Provides pr...	Started	Automatic	LocalSystem
QoS Admission Cont...	Provides n...		Automatic	LocalSystem
Remote Access Aut...	Creates a ...		Manual	LocalSystem
Remote Access Con...	Creates a ...	Started	Manual	LocalSystem
Remote Packet Cap...	Allows to c...		Manual	LocalSystem
Remote Procedure ...	Provides th...	Started	Automatic	LocalSystem
Remote Procedure ...	Manages t...		Manual	LocalSystem

(The NFM Server Service is set to start as a Windows Service from the Windows Services Control Panel)

*** When NFM Server is installed, the 'Startup Type' is set to Manual. It is recommended that you change this to Automatic, so the Service will start whenever the OS is started.*

Installing the TPS® Command Service package (Part 2 of 2):

OS	Install Command	Directories

AIX	installp -acFd <*filename*> all	/usr/lpp/tps/cserv/ (product directory) /var/tps/cserv/ (working directory)
Linux	tar xvf <*filename*> installp_tpsnfm	/opt/tps/cserv/ (product directory) /var/tps/cserv/ (working directory)
Windows	<*Double click install executable*>	C:\Program Files\TPS Systems\CSERV\ (product & working directory)

Cserv files/directories:

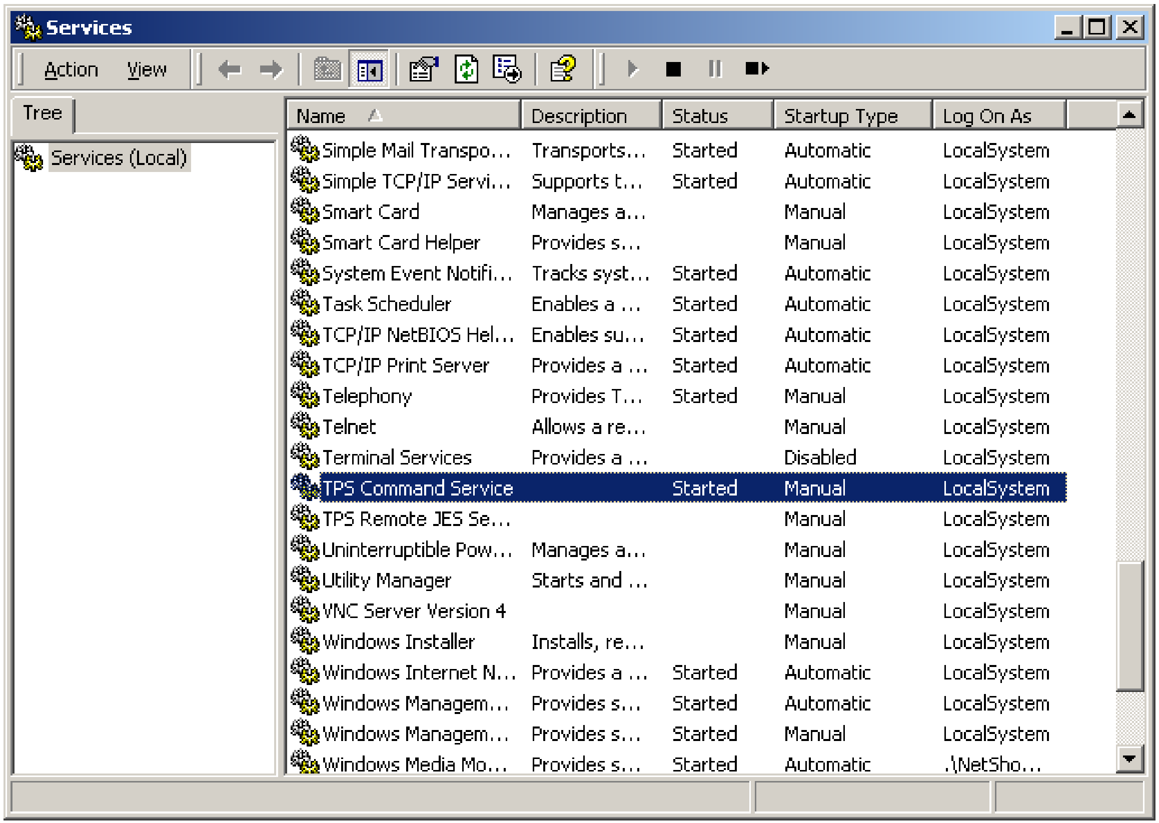
```
     <product directory>
          cserv          Cserv binaries / executables

     <working directory>
          apps           Scripts that point to applications used
                         by CServ
          conf           Configuration options
          /logs          Cserv logs
```

Line added to startup script / service:

AIX (/etc/inittab): tpscserv:2:respawn:/usr/lpp/tps/cserv/bin/tpscserv

Linux (/etc/inittab): tc:2345:respawn:/opt/tpscserv/bin/tpscserv

Windows



Name	Description	Status	Startup Type	Log On As
Simple Mail Transpo...	Transports...	Started	Automatic	LocalSystem
Simple TCP/IP Servi...	Supports t...	Started	Automatic	LocalSystem
Smart Card	Manages a...		Manual	LocalSystem
Smart Card Helper	Provides s...		Manual	LocalSystem
System Event Notifi...	Tracks syst...	Started	Automatic	LocalSystem
Task Scheduler	Enables a ...	Started	Automatic	LocalSystem
TCP/IP NetBIOS Hel...	Enables su...	Started	Automatic	LocalSystem
TCP/IP Print Server	Provides a ...	Started	Automatic	LocalSystem
Telephony	Provides T...	Started	Manual	LocalSystem
Telnet	Allows a re...		Manual	LocalSystem
Terminal Services	Provides a ...		Disabled	LocalSystem
TPS Command Service		Started	Manual	LocalSystem
TPS Remote JES Se...			Manual	LocalSystem
Uninterruptible Pow...	Manages a...		Manual	LocalSystem
Utility Manager	Starts and ...		Manual	LocalSystem
VNC Server Version 4			Manual	LocalSystem
Windows Installer	Installs, re...		Manual	LocalSystem
Windows Internet N...	Provides a ...	Started	Automatic	LocalSystem
Windows Managem...	Provides s...	Started	Automatic	LocalSystem
Windows Managem...	Provides s...	Started	Manual	LocalSystem
Windows Media Mo...	Provides s...	Started	Automatic	.\NetSho...

(The TPS® Command Service is set to start as a Windows Service from the Windows Services Control Panel)

** *When TPS® Command Service is installed, the '`Startup Type`' is set to `Manual`. It is recommended that you change this to `Automatic`, so the Service will start whenever the OS is started.*

Upgrading the NFM Server package

To upgrade the NFM Server you will need to install the NFM Server package executable. This will overwrite the binary files located in the NFM's product directory and reorganize the NFM database files, if necessary. Once you have successfully upgraded the NFM Server you will need to upgrade the NFM GUI Interface. See "*Coping over NFM files to the HTTP Server*" under NFM GUI Components for more information.

Removing the NFM Server package

To remove the NFM Server requires two pieces be removed, the NFM Server and TPS® Command Service (`Cserv`) packages. Removing the NFM Server & TPS® Command Service should be done by `root` on UNIX platforms and `Administrator` on Windows. Removal can be done with the following commands:

OS	Uninstall Command (NFM Server)	Uninstall Command (TPS® Command Service)
AIX	`/usr/lpp/tps/nfm/bin/uninstall *`	`/usr/lpp/tps/cserv ***`
Linux	`/opt/tps/nfm/bin/uninstall *`	`rm –r /opt/tps/cserv **/***`
Windows	Go to Windows Control panel, select 'Add Remove Programs', select 'TPS/NFM Server", press the "Remove" or "Uninstall" button. ****	Go to Windows Control panel, select 'Add Remove Programs', select 'TPS/Command Server", press the "Remove" or "Uninstall" button.

* The `uninstall` script will remove everything in `/opt/tps/nfm/` or `/usr/lpp/tps/nfm`; however, it will NOT delete anything in the `/var/tps/nfm`directories. If you want NFM Server removed completely you will have to remove everything in `/var/tps/nfm` manually.

**Unlike the NFM Server product, there is no `uninstall` script for TPS® Command Service. First you should stop the process. Then you can remove everything in:

*** If you have TPS®/Network Transaction Manager (NTM) installed on this system be aware that removing TPS® Command Service will prevent it from working.

**** `Uninstall` will remove everything in `C:/Program Files/TPS Systems/NFMServer` except the attachments, commands, `dba`, and log directories. If you want NFM Server removed completely you will have to remove these directories manually.

Creating NFM Server Backups

The following directories should be backed up regularly. These directories contain configuration files as well as database files for

UNIX:

```
/var/tps/nfm/*
/var/tps/cserv/*
```

Windows:

```
C:\Program Files\TPS Systems\NFMServer\*
```

NFM Server Processes

Below is a list of processes started by the NFM Server:

Process Name	Description
nfmserv	NFM main server program. One instance (of the process) should always exist (be loaded in mermory) while the NFM server is running which should be all the time. Controls the submitted plan queue. Responsible for starting plans at their designated time.
nfmservi **	NFM remote command server program. This program services the remote command interface utility program that can call in from the NFM client computers. One instance should always exist while the NFM server is running if remote commands are enabled.
nfmservp **	NFM proxy connection server. One instance should always exist if any nodes using connection type other than LISTEN are configured. This program services clients that are configured to permanently call in.
nfm	The nfm command. Used internally by the GUI and also as part of the local and remote command line interface. Any number of instances may be loaded simultaneously.
nfmjsrv	NFM plan instance server. One instance should be loaded for each ACTIVE plan instance (not FINISHED and not SCHEDULED).

** `nfmservp` and `nfmservi` processes are not started if the "System Settings" field "Enable remote commands" is checked off. **

TPS® Command Service Processes

Below is a list of processes started by the TPS® Command Service (part of NFM Server):

Process Name	Description
tpscserv	The main process that handles communication between the NFM Server and NFM GUI.

NFM GUI Components

Installing the GUI Interface:

There are two choices available for using the NFM GUI. These are the Windows Stand-alone GUI and the Browser Based GUI. Either GUI Interface requires JAVA be installed.

Installing the Windows Stand-alone GUI package (*Optional*):

For those choosing the stand-alone JAVA Windows program:

Advantages: Up in minutes (Do not have to setup an HTTP Server)

Disadvantages: Windows only application
GUI updates have to be distributed manually to each computer

Install command: <*Double click on the install executable*>

Directories created: C:/Program Files/TPS Systems/Network File Manager/

Setting up the Browser-Based GUI Interface (*Recommended Choice*):

For those choosing the browser based GUI Interface:

Advantages: GUI updates are automatically sent when the HTML page is loaded via the client's web browser. No need to reinstall client software.

Disadvantages: People not familiar with HTTP Server configuration might have to spend time learning about web site administration or setup a HTTP Server.

Configuring the HTTP Server

First, you will need to install or configure an HTTP Server. With hundreds of different HTTP servers out there it is impossible to go over every configuration and installation. Though we do not provide any support for any HTTP server we currently use Apache on numerous systems. Apache is available for download at http://www.apache.org. Documentation and support is also available through this site.

You can check if you have an HTTP server already running by simply trying to connect to it via your web browser.

Coping over NFM files to the HTTP Server

Once you have your HTTP Server installed and configured, you need to copy the NFM GUI directory to the HTTP Server's base directory (this step is also required if you are performing an upgrade). We recommend creating an 'NFM' directory off the HTTP Server's base directory then copy:

AIX*	`/usr/lpp/tps/nfm/gui`
Linux*	`/opt/tps/nfm/gui`
Windows	`/Program Files/TPS Systems/NFMServer/GUI`
	`to` *`<HTTP Server's base directory>`*`/NFM`

* On UNIX platforms, create a symbolic link between the `gui` directory and <*HTTP Server's base directory*>/NFM. That way if you receive an NFM Server update the GUI files will be automatically updated on the HTTP Server without the need for a copy.

Verifying Java is Installed

Finally, you will need to verify each browser has Sun Java Virtual Machine (JVM) version 1.4.0_01 or higher. We have provided a test script to test your JAVA version at http://www.tps.com/NFMTest or for Windows platforms you can access Windows Control Panel and select Java. If the browser doesn't have Sun JVM or this version of Sun JVM you can download it from http://sun.java.com/

NFM Client Component

Installing the NFM Client package:

The NFM Client runs on several different platforms including AIX, 4690, Stratus, Solaris, OS/390, SCO OpenServer 5, HP, Windows and Linux. Installing the NFM Client should be installed by `root` on UNIX platforms and `Administrator` on Windows.

OS	Install Command	Directories
AIX	`Installp –acFd <filename> all`	/usr/lpp/tps/nfmc/ (product directory) /var/tps/nfmc (working directory)

Linux	`tar xvf <filename>`
	`installp_tpsnfm`
		/opt/tps/nfmc/ (product directory)
		/var/tps/nfmc (working directory)

Solaris	`pkgadd -d <full path>/<filename>`	/opt/tps/nfmc/ (product directory) /var/tps/nfmc (working directory)
SCO OpenServer 5	`pkgadd -d <full path>/<filename>`	/opt/tps/nfmc/ (product directory) /var/tps/nfmc/ (working directory)
OS	**Install Command**	**Directories**

4690	Create a subdirectory for the product in any desired location.
	
	Example:
	`Mkdir C:/ADX_UPGM/NFMC`
	
	Copy the NFM client modules from the install medium to this directory.
	
	Example:
	`Copy a:/*.* C:/ADX_UPGM/NFMC`
	
	(This will include `NFMC.386` and `NFMI.386` at this time.)
	
	`NFMC.386` The primary NFM client load module.
	`NFMI.386` The NFM remote command interface module.
	
	From the Controller configuration menu, create a background application that starts `NFMC.386` automatically from the installed location.
	
	NOTE: The NFM client will automatically create needed sub-directories in the installed location.

Stratus	• Create a subdirectory for the product in any desired location, example:
	
	`create_directory (master_disk)>system>nfmc`
	
	• Transfer the `nfmclient.pm` and `nfmi.pm` modules to this directory.
	
	• Verify that both `nfmclient.pm` and `nfmi.pm` are fixed files with record length 4096 as required for VOS program modules.
	
	NOTE: `nfmclient.pm` will automatically create needed `>tmp` and `>logs` sub-directories in the current directory at startup.
	
	The NFM client must be started and left executing on Stratus for it to respond to incoming request from the NFM server. In most cases it will be useful to create a command macro `(.cm)` to start `nfmclient.pm` with the appropriate options:
	
	`!start_process (master_disk)>system>nfmc>nfmclient.pm &+`
	`-output_path (master_disk)>system>nfmc>nfmclient.out &+`
	`-process_name nfmclient &+  -privileged &+`
	`-current_dir (master_disk)>system>nfmc`

OS/390	Components are installed on OS/390 using the `TSO RECEIVE` command to install a product image file. This file should be placed in a temporary dataset on the target computer. Files: `PACKAGE.NFMC` Perform the following steps: • Transfer the `PACKAGE.NFMC` file to the host as an MVS dataset with the parameters - `SEQ, FB`, and record size 80. Make sure this is done as a binary transfer. The product will use a top-level qualifier of `TPS01`, however this is configurable. • From the TSO command line, `unpackage` the file as follows: `RECEIVE INDATASET('TPS01.PACKAGE.NFMC')` After successful execution of the RECEIVE command, the following NFM components are installed: `TPS01.NFMC.LOAD(NFMC)` The primary NFM client load module. `TPS01.NFMC.LOAD(NFMI)` The NFM remote command interface module.	
Windows	*<Double click install executable>*	C:/Program Files/TPS Systems/NFMClient/ (product & working directory)

Upgrading the NFM Client package

The NFM Client can be upgraded by simply running the executable just like a new install. All necessary files will be replaced with the new ones.

Automating NFM Windows Installs

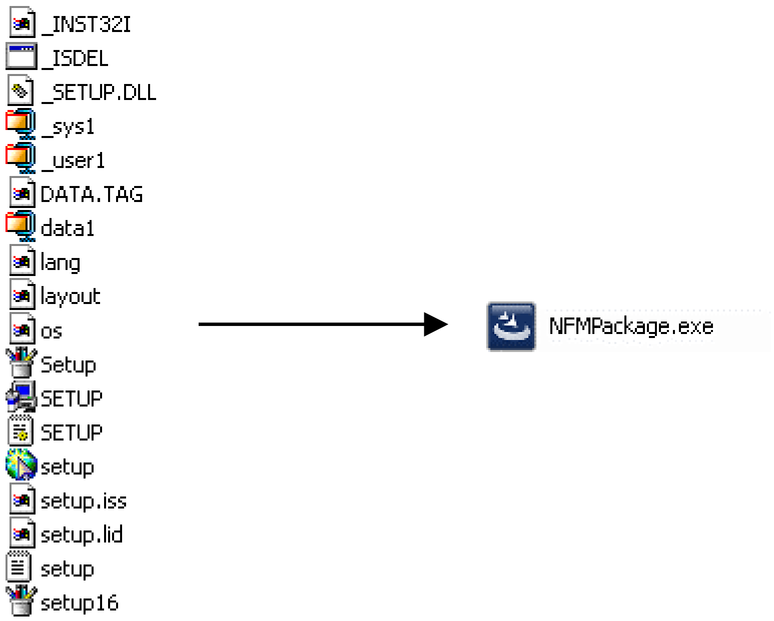
The NFM Client and Server install packages are GUI-based applications that require human interaction and prompting to install the product. However, through the use of a response file you can automate the install procedure.

How the install files are packaged

NFMPackage.exe

All NFM install packages (NFM Server, NFM Client, NFM Stand-alone GUI and TPS® Command Service) are built using Macromedia's InstallShield and Packaging for the Web programs. People familiar with InstallShield's self-extracting `EXE` and/or those that have automated the install of other InstallShield built packages will find our steps very similar.

NFM Install Files



Step 1: Making the response file

Setup.iss

The response file can be made in one of two ways. You can build this file automatically from the install or you can edit the response file that comes with the package.

First you'll need to extract the files by double clicking on the package (`NFMPackage.exe`) we shipped you. This will start the GUI-based NFM install screen. Once the Welcome screen comes up, click on the '`Next`' button. This will extract all the files and place them in your `%TEMP%` directory. This is usually `C:\Document and Settings\<current user>\Local Settings\Temp\`. To verify this run `echo %TEMP%` from the command line. In `%TEMP%` there will be a directory that starts with `pft*` this is where the NFM install files are located. Copy these files to another location and close the GUI install screen. ** Once the GUI install screen is closed the files in `%TEMP%\pft*` will be deleted. **

Now you should decide rather to edit the response file (`Setup.iss`) that came with the package or have `Setup.exe` automatically build one.

Edit the response file directly

Setup.iss

The NFM package comes with a response file called Setup.iss. The values or “`responses`” contained in this file are the default values used during install (for example: the default install `path/directory`). If you want to use the default install values you will not have to modify the file. However, if you want to make a change, the file is text and you may change these values with any text editor.

Automatically build the response file

Setup.exe

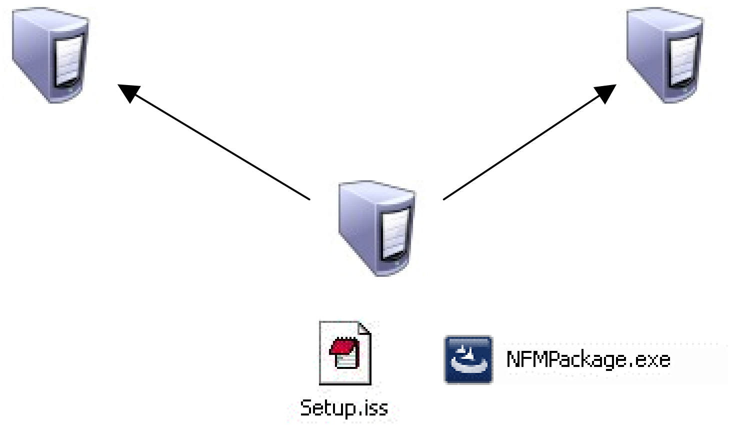
If you not comfortable or unfamiliar with changing values, InstallShield provides a “friendly” way to create the response file. From the new location start the command prompt and run:

```
SETUP.exe /r /f1”C:\setup.iss”
```

Where: `/r` – tells setup to record your answers
`/f1` – where to save the response file

The NFM install GUI screen will now appear. Follow each step and screen to complete installation. ** *This will install the software on this machine* ** Once completed, you will have a response file which will be used to automate the install process.

Step 2: Distributing the Software



Regardless of which method you choose to distribute the software (CD, network copy, etc.) be sure the original package (`NFMPackage.exe`) and the response file (`Setup.iss`) are distributed to the target system.

Step 3: Install the Software Package

Once both files (`NFMPackage.exe` and `Setup.iss`) are on the target system, you can install the NFM package with the following command:

```
NFMPackage.exe /s /a /s /SMS /f1"C:\<PATH>\Setup.iss"
               /f2"C:\<PATH>\Setup.log"
```

Where: /s – silent mode for package installation
/a – adds / passes remaining options to SETUP.exe
/s - silent mode for SETUP.exe
/SMS - waits for SETUP.exe to finish before exiting
/f1 – full path to the response file
/f2 – full path to the install log

After the package has been installed, the Setup.log will contain a "ResultCode" value that is the success or failure of the install. Here is a list of possible result codes:

0	Success.
-1	General error.
-2	Invalid mode.
-3	Required data not found in the Setup.iss file.
-4	Not enough memory available.
-5	File does not exist.
-6	Cannot write to the response file.
-7	Unable to write to the log file.
-8	Invalid path to the InstallShield Silent response file.
-9	Not a valid list type (string or number).
-10	Data type is invalid.
-11	Unknown error during setup.
-12	Dialog boxes are out of order.
-51	Cannot create the specified folder.
-52	Cannot access the specified file or folder.
-53	Invalid option selected.

** These values are not generated by NFM ** but instead by InstallShield.

For additional questions related to InstallShield's automated installs please consult www.installshield.com

NFM Client files / directories:

<product directory>

bin	NFM Client binaries / executables and config files

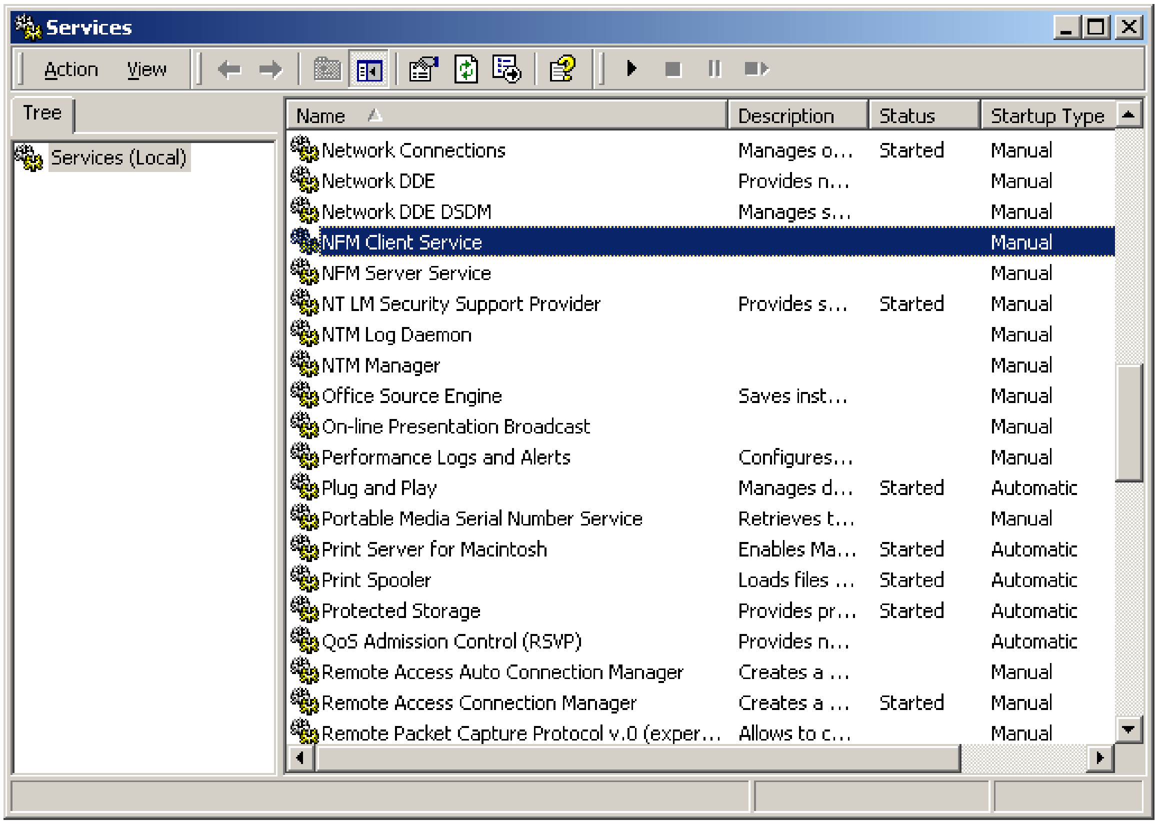
<working directory>

logs	NFM Client logs
ssl	SSL related files
tmp	Temporary NFM files used during the instance of the plan

Line added to startup script / service:

```
AIX    (/etc/inittab): nfmclient:2:respawn:/usr/lpp/tps/nfmc/bin/nfmclient
Linux  (/etc/inittab): nfc:2345:respawn:/opt/tps/nfmc/bin/nfmclient
Sun    (/etc/inittab):  nf:234:respawn:/opt/tps/nfmc/bin/nfmclient
SCO    (/etc/inittab): nf:234:respawn:/opt/tps/nfmc/bin/nfmclient
```

Windows



Name	Description	Status	Startup Type
Network Connections	Manages o...	Started	Manual
Network DDE	Provides n...		Manual
Network DDE DSDM	Manages s...		Manual
NFM Client Service			Manual
NFM Server Service			Manual
NT LM Security Support Provider	Provides s...	Started	Manual
NTM Log Daemon			Manual
NTM Manager			Manual
Office Source Engine	Saves inst...		Manual
On-line Presentation Broadcast			Manual
Performance Logs and Alerts	Configures...		Manual
Plug and Play	Manages d...	Started	Automatic
Portable Media Serial Number Service	Retrieves t...		Manual
Print Server for Macintosh	Enables Ma...	Started	Automatic
Print Spooler	Loads files ...	Started	Automatic
Protected Storage	Provides pr...	Started	Automatic
QoS Admission Control (RSVP)	Provides n...		Automatic
Remote Access Auto Connection Manager	Creates a ...		Manual
Remote Access Connection Manager	Creates a ...	Started	Manual
Remote Packet Capture Protocol v.0 (exper...	Allows to c...		Manual

(The NFM Client is set to start as a Windows Service from the Windows Services Control Panel)

*** When NFM Client is installed, the '`Startup Type`' is set to `Manual`. It is recommended that you change this to `Automatic`, so the Service will start whenever the OS is started.*

NFM Client Processes

Process Name	Description
`nfmclient`	NFM client program. One instance should always exist while the NFM client is running which should be all the time.

`nfmi`	NFM remote command line utility program. Loaded as needed.

Removing the NFM Client package:

Removal can be done with the following commands:

OS	Uninstall Command
AIX	`/usr/lpp/tps/nfm/bin/uninstall *`
Linux	`/opt/tps/nfm/bin/uninstall *`
Windows	Go to Windows Control panel, select '`Add Remove Programs`', select '`TPS/NFM Client`", press the "`Remove`" or "`Uninstall`" button. **

* The `uninstall` script will remove everything in `/opt/tps/nfmc/` or `/usr/lpp/tps/nfmc`; however it will NOT delete anything in the `/var/tps/nfmc` directories. If you want NFM Client removed completely you will have to remove everything in `/var/tps/nfmc` manually.

** `Uninstall` will remove everything in `C:/Program Files/TPS Systems/NFMClient` except the `log` directory. If you want NFM Client removed completely you will have to remove these directories manually.

Creating NFM Client Backups

The following directories should be backed up regularly. These directories contain configuration files for

UNIX:
```
/var/tps/nfmc/*
```

Windows:
```
C:\Program Files\TPS Systems\NFMClient\*
```

NFM Client Configuration File

NFM Client uses an optional configuration file that resides in the NFM Client's main install directory and is usually called `nfm.cfg`. Settings inside this configuration file will override the default NFM settings. Some reasons for using an NFM Client configuration file include:

Want the client to use a different TCP/IP port.
Want to use SSL type communications
Want the client to use multiple connections or sockets
Want the client to use a different connection type other than `Listen`

A configuration file looks something like this:

```
NFMSERVER=nfmserver
NFMNODE=unknown
CONNECT=LISTEN, PORT=8008, INTERVAL=, DURATION=, SSL=0, KEY=,
CERTIFICATE=,
```

The first part contains the following fields:

NFMSERVER=[address] This field should be set to the IP address or hostname that the NFM server is running on.

NFMNODE=[name] This field should be set to the NFM node name that represents this node.

(NOTE: These two fields are only required if a `CONNECT` record of type `PERMANENT` is defined below).

The remaining portion of the file contains one or more connection entries. These entries should equivalent to the connection entries defined in the NFM node or model definition for this node. The following fields are available:

CONNECT=[type] This field should be set to the required connection type. Currently supported values are `LISTEN` and `PERMANENT`.

PORT=[number] This field should be set to desired port for this connection.

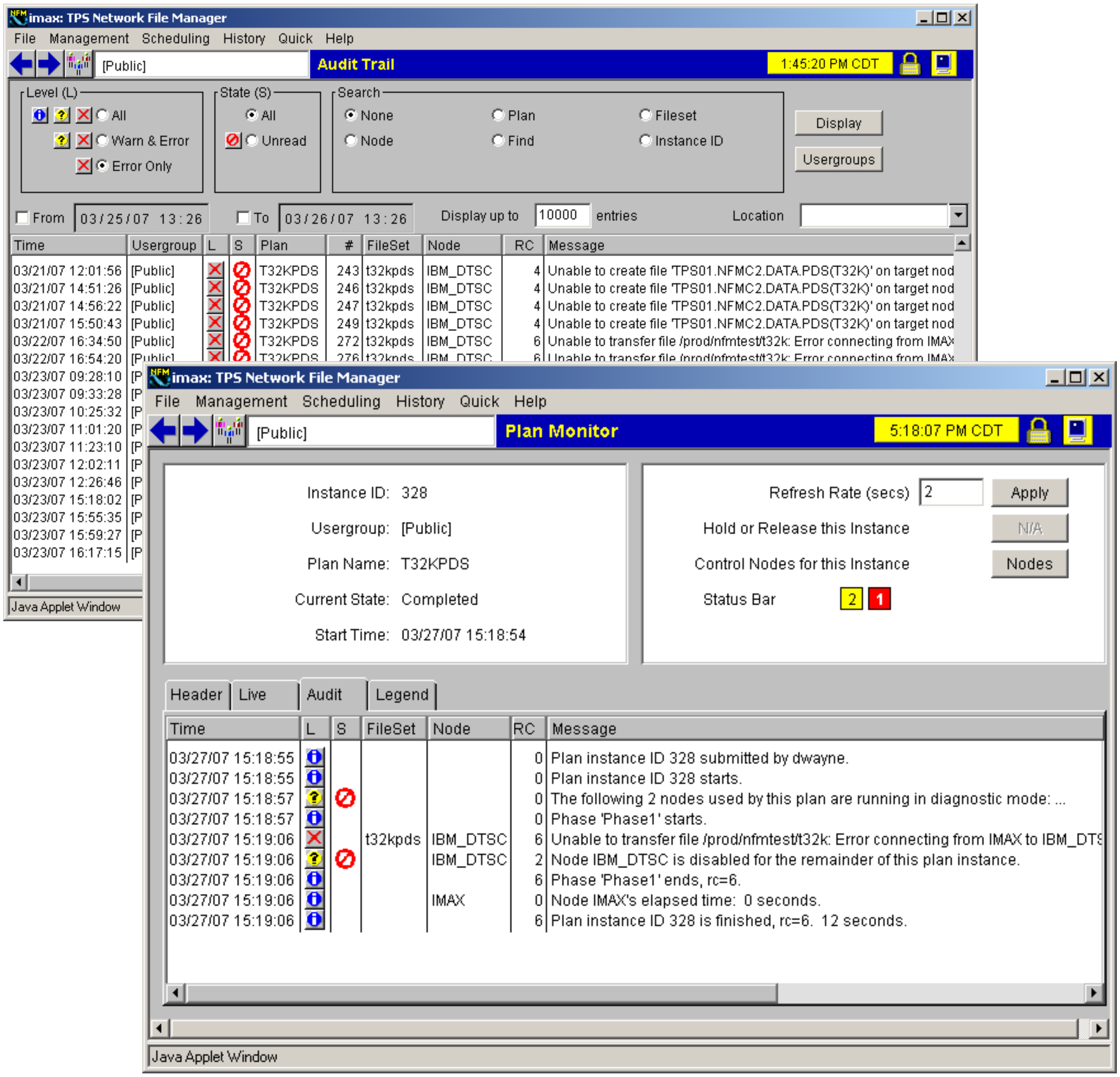
SSL=[Y/N] Indicates if this is an SSL connection.

Troubleshooting

NFM was designed to be very comprehensive with regards to capturing and reporting errors in a very user-friendly fashion. Most errors will result in either a popup window display (*Figure 3*) (if the error occurred during an interactive / GUI session) or an audit message (*Figure 1 & 2*) for those that occurred during plan activity or any other unattended operation.

Figure 1 (top): Displays Audits from the Entire System (Viewable by selecting '`History`' and '`Audit Trail`')

Figure 2 (bottom): Displays Audits Pertaining to that Plan Instance (Viewable by selecting '`Scheduling`' and '`Plan Activity`' and double clicking the plan)



□ From 03/25/07 13:26 □ To 03/26/07 13:26 Display up to 10000 entries Location

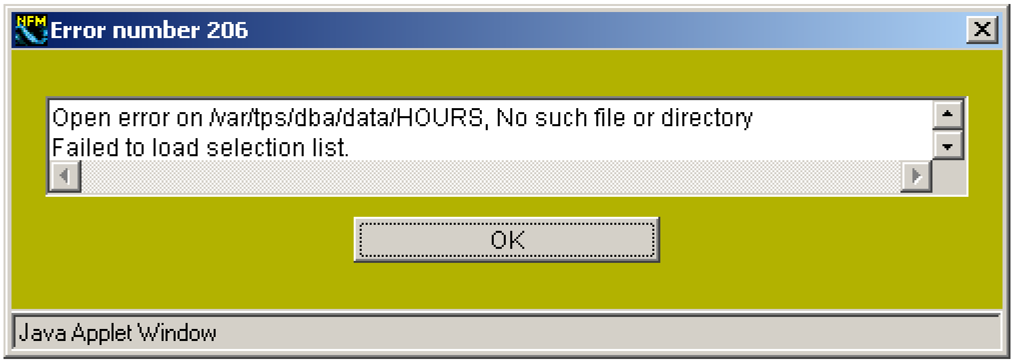
Time	Usergroup	L	S	Plan	#	FileSet	Node	RC	Message
03/21/07 12:01:56	[Public]	☒	⊘	T32KPDS	243	t32kpds	IBM_DTSC	4	Unable to create file 'TPS01.NFMC2.DATA.PDS(T32K)' on target nod
03/21/07 14:51:26	[Public]	☒	⊘	T32KPDS	246	t32kpds	IBM_DTSC	4	Unable to create file 'TPS01.NFMC2.DATA.PDS(T32K)' on target nod
03/21/07 14:56:22	[Public]	☒	⊘	T32KPDS	247	t32kpds	IBM_DTSC	4	Unable to create file 'TPS01.NFMC2.DATA.PDS(T32K)' on target nod
03/21/07 15:50:43	[Public]	☒	⊘	T32KPDS	249	t32kpds	IBM_DTSC	4	Unable to create file 'TPS01.NFMC2.DATA.PDS(T32K)' on target nod
03/22/07 16:34:50	[Public]	☒	⊘	T32KPDS	272	t32kpds	IBM_DTSC	6	Unable to transfer file /prod/nfmtest/t32k: Error connecting from IMAX
03/22/07 16:54:20	[Public]	☒	⊘	T32KPDS	276	t32kpds	IBM_DTSC	6	Unable to transfer file /prod/nfmtest/t32k: Error connecting from IMAX
03/23/07 09:28:10	[P								
03/23/07 09:33:28	[P								
03/23/07 10:25:32	[P								
03/23/07 11:01:20	[P								
03/23/07 11:23:10	[P								
03/23/07 12:02:11	[P								
03/23/07 12:26:46	[P								
03/23/07 15:18:02	[P								
03/23/07 15:55:35	[P								
03/23/07 15:59:27	[P								
03/23/07 16:17:15	[P								

Instance ID: 328
Usergroup: [Public]
Plan Name: T32KPDS
Current State: Completed
Start Time: 03/27/07 15:18:54

Refresh Rate (secs) 2
Hold or Release this Instance
Control Nodes for this Instance
Status Bar 2 1

Time	L	S	FileSet	Node	RC	Message
03/27/07 15:18:55	ⓘ				0	Plan instance ID 328 submitted by dwayne.
03/27/07 15:18:55	ⓘ				0	Plan instance ID 328 starts.
03/27/07 15:18:57	?	⊘			0	The following 2 nodes used by this plan are running in diagnostic mode: ...
03/27/07 15:18:57	ⓘ				0	Phase 'Phase1' starts.
03/27/07 15:19:06	☒		t32kpds	IBM_DTSC	6	Unable to transfer file /prod/nfmtest/t32k: Error connecting from IMAX to IBM_DTS
03/27/07 15:19:06	?	⊘		IBM_DTSC	2	Node IBM_DTSC is disabled for the remainder of this plan instance.
03/27/07 15:19:06	ⓘ				6	Phase 'Phase1' ends, rc=6.
03/27/07 15:19:06	ⓘ			IMAX	0	Node IMAX's elapsed time: 0 seconds.
03/27/07 15:19:06	ⓘ				6	Plan instance ID 328 is finished, rc=6. 12 seconds.

(Figure 3)



Open error on /var/tps/dba/data/HOURS, No such file or directory
Failed to load selection list.

This section focuses on the most common problems that may occur where the cause may not be obvious, and suggested procedures for diagnosing and making corrections. The problems are divided by category into the following sections:

- Problems logging on to NFM.
- Problems communicating with NFM nodes.
- Problems related to system date and time.
- Unexpected results from a plan.

Problems logging on to NFM

Browser option

Logging on to the NFM system using the browser option involves more steps occurring under the covers than with the stand-alone option. Because of this more things can go wrong. If you have an error such as "The page cannot be displayed," it is recommended that you try the stand-alone option first as this may give you back a more descriptive error if it is a communications problem. If the stand-alone option works but the browser does not, then the http server may not be configured correctly or may not be currently running. If you are able to successfully get to the sign on screen using the browser option, then you have successfully established an http session and there is no need to try the stand-alone option.

Stand-alone option

If you receive one of the following errors:

Unknown host name.
Unable to connect to host `host_name`: Operation timed out.
Unable to connect to host `host_name`*:* Host unreachable.

Or, you receive anything else that would seem to indicate a communications error, the IP name or address is probably configured wrong or the network is down. Try pinging the address from the command line, if that does not work there is a network problem. If you get the error:

Unable to connect to host `host_name:8100` Connection refused.

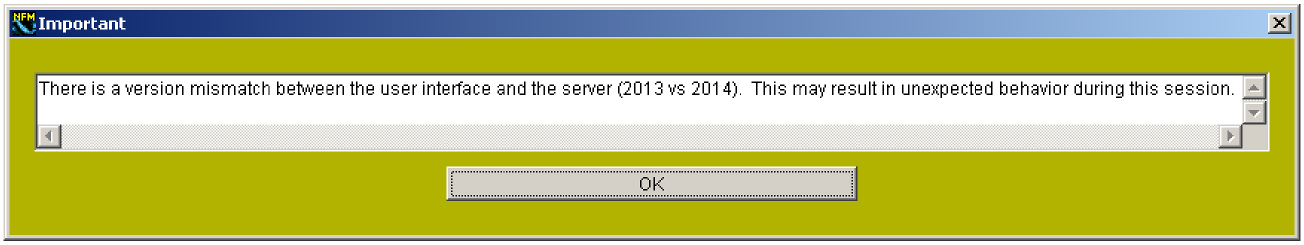
It is likely the TPS® Command server is not running, check to make sure it is installed and running.

If one of these problems still persists, you may need to check with the network administrator. Several of these problems could also occur (most likely operation timed out) if you are attempting to cross a firewall that is not configured to allow this type of socket connection. This could occur even if a ping works. NFM can be configured to use a different port than the default of `8100` if this port is prohibited by a firewall.

If during the logon attempt you get the following message:

Command rejected, failed to execute command `nfm`.

This is a probable indication that the TPS® Command server is working fine but that the NFM server product is not installed or running.



There is a version mismatch between the user interface and the server (2013 vs 2014). This may result in unexpected behavior during this session.

Version mismatch errors are common problems after an NFM Server upgrade. When logging into the system the NFM GUI checks with the NFM Server to make sure they are both running the same version. Running an older version of the NFM GUI with a newer version of the server can cause some unexpected behavior.

The most common mistake when upgrading the NFM Server is not copying the GUI directory to the web server directory. This is the most likely cause for this error.

Problems communicating with NFM nodes

NFM clients

Any of the errors listed above with regards to IP communications problems may show up as audit error messages during attempts by NFM to communicate with NFM nodes during file transfers or other network activity. Once again, try the ping command to see if it is a network problem. If the transfer is taking place between two remote nodes (neither one is the NFM server), make sure the nodes can ping each other.

If you are getting the connection refused message, the NFM client is probably not installed or not running on the node in question. A simple way to test connectivity to an NFM node is to perform the following command on the NFM server systems command line: `nfm,statnode,`*`node_name`*. This contacts the node and prints the version of the NFM client running (see the NFM command line interface in Chapter 7, "Tools").

If you are still getting network errors even though the pings work, consider the firewall problem as stated in the previous section. In the case of contacting NFM clients, the default port value is 8008. If desired, this can be changed, see the description of the communications name in node configuration.

NFM non-clients

If problems are encountered communicating with FTP type nodes, you should try to perform the FTP command manually. If this does not work there may be a problem with the way FTP is configured for either the FTP site or the computer attempting to make the FTP connection. You may need to reference the appropriate operating system documentation to determine the problem.

Problems related to system date & time

It is fairly important that all computers involved in NFM keep their system date and time relatively accurate. This is especially true between the NFM server and any computer running the user interface (browser or stand-alone). This is important because the NFM interface keeps its own clock synchronized with the one on the NFM server, it does however, adjust the display to show local time zone information. This is why the clock displayed on each NFM screen may be close but not exactly matches the time shown by the local operating system.

If the date and time are off by more than half an hour, you may see one of the following conditions occur when you try to schedule a plan:

The NFM system will warn you that you are attempting to schedule something in the past. This message is intended to keep someone from accidentally starting a plan immediately if it was not the intent.

You notice that a plan has not started yet, even though the start time has already passed looking at the NFM clock display.

Unexpected results from a plan

If a plan does not accomplish its intended goals or does something unexpected, run the plan with the `History` option set to `Detail` (defined in plan header or set from plan scheduler). This causes NFM to record in full, each operation a plan performed. For a file transfer, it records the exact source and target filename used for each node involved.

This will also record when events are being skipped due to termination logic that was configured. This will help identify items that may not be set up correctly. It would probably be a good idea to keep this option configured at all times, unless the resulting amount of data generated would be undesirable.

If a plan that has been running terminates and information that should have appeared in the audit trail does not, check to make sure the file system or disk drive containing NFM's databases has not filled up.

If environment variables are used but do not seem to contain the values that are expected, try executing a simple function `set` on any target node that is in question (you must check the `Use Shell` box). This will make the entire environment variable list in effect for that client to print out to an audit message (you must click on the audit entry with the trailing '…' to see the list). This is available because NFM sends all environment variables down to an execute function and they become locally defined.

The remaining section focuses on problems that may occur where the cause may not be obvious. For these kind of problems, gathering additional information might help you troubleshoot or be needed by TPS® Technical Support to troubleshoot your problem. If you need to contact TPS® Technical Support please provide the following information:

NFM System Logs
NFM Server Logs
NFM Client Logs
TPS® Command Service Logs
NFM Server Database Files
NFM Job Logs / Core Files
Gathering Platform Specific Information
GUI Screen Captures

** This information can be supplied to TPS® via email or contact us for FTP instructions. **

Gathering NFM System Logs

(For your review or TPS® Technical Support)

Some problems may require generating and examining log files. First, based on the nature of the problem, determine which NFM component(s) (NFM Server, NFM Client, TPS® Command Service or NFM GUI Interface) should have logging turned on.

NFM Server Logs

Section 1: Gathering NFM Server Logs

Why turn on NFM Server Logs?

The NFM Server will not start or abnormally exits

Due to the envolvement NFM Server plays in file or program execution (See Appendix B: Anatomy of a File Transfer), when troubleshooting a problem between NFM Clients or NFM Non-Client Nodes it is also a good idea to turn on NFM Server logs (in addition to NFM Client logging).

How to turn on NFM Server Logs?

Step 1: Make sure the `NFM Server Service` is stopped (Windows only)

Step 2: End any `nfmserv`, `nfmservi` and `nfmservp` processes that are running (using Task Manager, `End Task` button on Windows or `ps` command for UNIX).

Step 3: Remove old logs files from the NFM Server logs directory (archive any logs you might want to keep):

Platform	**Directory**
UNIX	`/var/tps/nfm/logs`
Windows	`C:/Program Files/TPS Systems/NFMServer/logs`

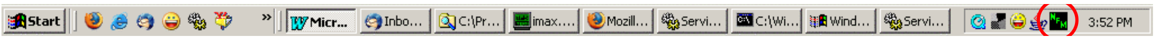
Step 4: Start the NFM Server. From a prompt start nfmserv.exe as follows:

Platform	**Directory**
UNIX	`nfmserv -tV`
Windows	`C:\Program  Files\TPS  Systems\NFMServer>nfmserv  -foreground -tV  **`

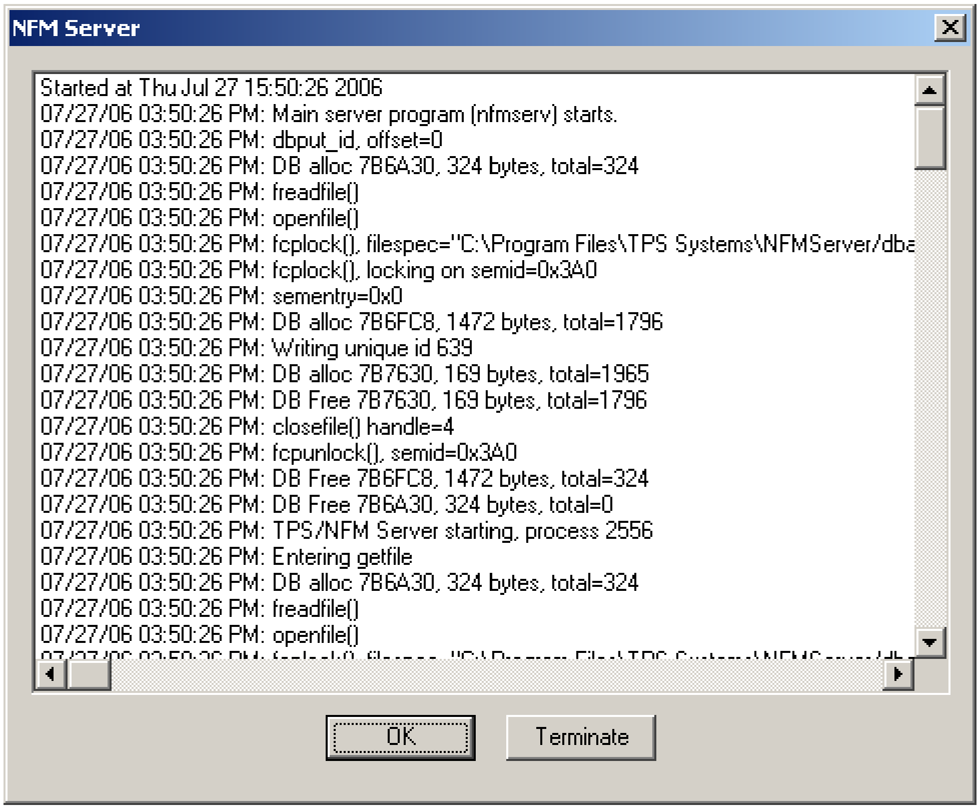
- Note anything that is printed to the screen.
- Note whether or not the `nfmserv` process is active (from Task Manager or the `ps  -deaf` command)

Step 5: Send everything in the NFM Server logs directory. This should include a `nfmserv` and `nfmservi` logs. All files will be text files.

** Starting the `nfmserv` program with the –foreground option will add an NFM icon in the System Tray.



Double-clicking on this icon will bring up a dialog box containing an NFM Server trace.



```
Started at Thu Jul 27 15:50:26 2006
07/27/06 03:50:26 PM: Main server program (nfmserv) starts.
07/27/06 03:50:26 PM: dbput_id, offset=0
07/27/06 03:50:26 PM: DB alloc 7B6A30, 324 bytes, total=324
07/27/06 03:50:26 PM: freadfile()
07/27/06 03:50:26 PM: openfile()
07/27/06 03:50:26 PM: fcplock(), filespec="C:\Program Files\TPS Systems\NFMServer/dba
07/27/06 03:50:26 PM: fcplock(), locking on semid=0x3A0
07/27/06 03:50:26 PM: sementry=0x0
07/27/06 03:50:26 PM: DB alloc 7B6FC8, 1472 bytes, total=1796
07/27/06 03:50:26 PM: Writing unique id 639
07/27/06 03:50:26 PM: DB alloc 7B7630, 169 bytes, total=1965
07/27/06 03:50:26 PM: DB Free 7B7630, 169 bytes, total=1796
07/27/06 03:50:26 PM: closefile() handle=4
07/27/06 03:50:26 PM: fcpunlock(), semid=0x3A0
07/27/06 03:50:26 PM: DB Free 7B6FC8, 1472 bytes, total=324
07/27/06 03:50:26 PM: DB Free 7B6A30, 324 bytes, total=0
07/27/06 03:50:26 PM: TPS/NFM Server starting, process 2556
07/27/06 03:50:26 PM: Entering getfile
07/27/06 03:50:26 PM: DB alloc 7B6A30, 324 bytes, total=324
07/27/06 03:50:26 PM: freadfile()
07/27/06 03:50:26 PM: openfile()
07/27/06 03:50:26 PM: fcplock(), filespec="C:\Program Files\TPS Systems\NFMServer/db
```

(*NFM Server trace window*)

Where are NFM Server Logs kept? What NFM Server Logs are created?

NFM Client Logs are kept in the following directory:

Platform	**Directory**
UNIX	`/var/tps/nfm/logs`
Windows	`C:/Program Files/TPS Systems/NFMServer/logs`

NFM Server Log files created (all files are text):

when NFM Server starts:

`nfmserv` *(NFM main server program)*
`nfmservi` *(if using NFM remote command program)*
`nfmservp` *(if using NFM proxy server)*

once the plan is run:

`job.`*`<PlanID>`*
`job.`*`<PlanID>`*`.`*`<Phase name>`*
`job.`*`<PlanID>`*`.`*`<Phase name>`*`.`*`<nodename>`*

Section 2: Examing the NFM Server Logs

What does the NFM Server Logs look like? What does it mean?

Now let's examine the different NFM Server Logs. From the above section several NFM Server Logs are generated.

Log file: `nfmserv`
Description: Log of the NFM main server program

Description	Log file
When process was terminated	Thread 0x00000001 fd 4 [nfmserv] 04/18/07 02:53:39.766 PM: nfmserv terminated by another process

Log file: `nfmseri`
Description: Log of the NFM remote command server program

Description	Log file
NFM listening on port	**Thread 0x00000001 fd 20** 04/13/07 04:04:20 PM: nfm_startthread(), fun=300298E4, s=2FF21908, ssize=4496, logdir=(NULL), logname=8018, pflag=0, childidp=0, childc=0, async=1 04/13/07 04:04:20 PM: about to start thread. 04/13/07 04:04:20 PM: pthread_create returns 0, errno=0. 04/13/07 04:04:20 PM: thread 0x102 started. 04/13/07 04:04:20 PM: nfm_startthread(), fun=300298E4, s=2FF21908, ssize=4496, logdir=(NULL), logname=8019, pflag=0, childidp=0, childc=0, async=1 04/13/07 04:04:20 PM: about to start thread. 04/13/07 04:04:20 PM: pthread_create returns 0, errno=0. 04/13/07 04:04:20 PM: thread 0x203 started. 04/13/07 04:04:20 PM: child 00000102 inherits log 04/13/07 04:04:20 PM: nfmslisten(), port=8018 04/13/07 04:04:20 PM: nfmservi 04/13/07 04:04:20 PM: listensock(). port=8018

	04/13/07 04:04:20 PM: SockListen(). 04/13/07 04:04:20 PM: SockListen(). I am imax. My address is 2 04/13/07 04:04:20 PM: SockListen(). calling socket(AF_INET,SOCK_STREAM, 0) 04/13/07 04:04:20 PM: SockListen(). sock_fd=22, backlog=1024 04/13/07 04:04:20 PM: child 00000203 inherits log 04/13/07 04:04:20 PM: nfmslisten(), port=8019 04/13/07 04:04:20 PM: nfmservi 04/13/07 04:04:20 PM: listensock(). port=8019 04/13/07 04:04:20 PM: SockListen(). 04/13/07 04:04:20 PM: SockListen(). I am imax. My address is 2 04/13/07 04:04:20 PM: SockListen(). calling socket(AF_INET,SOCK_STREAM, 0) 04/13/07 04:04:20 PM: SockListen(). sock_fd=24, backlog=1024 04/13/07 04:04:20 PM: SockListen(). bind() ok, listening on socket. 04/13/07 04:04:20 PM: SockListen(). returning OK 04/13/07 04:04:20 PM: answersock(). port=8019 04/13/07 04:04:20 PM: SockAnswer(). listening on socket. 04/13/07 04:04:20 PM: SockAnswer(). entering accept with sock_fd=24. 04/13/07 04:04:20 PM: SockListen(). bind() ok, listening on socket. 04/13/07 04:04:20 PM: SockListen(). returning OK 04/13/07 04:04:20 PM: answersock(). port=8018 04/13/07 04:04:20 PM: SockAnswer(). listening on socket. 04/13/07 04:04:20 PM: SOCKANSWER(). ENTERING ACCEPT WITH SOCK_FD=22.

Log file: `nfmserp`
Description: Log of the NFM proxy server program

Description	Log file
NFM listening on port	Thread 0x00000001 fd 20 04/13/07 04:04:20 PM: nfm_startthread(), fun=30013E30, s=2FF21718, ssize=5008, logdir=(NULL), logname=8028, pflag=0, childidp=0, childc=0, async=1 04/13/07 04:04:20 PM: about to start thread. 04/13/07 04:04:20 PM: child 00000102 inherits log

	04/13/07 04:04:20 PM: nfmplisten(), port=8028 04/13/07 04:04:20 PM: nfmservp 04/13/07 04:04:20 PM: listensock(). port=8028 04/13/07 04:04:20 PM: SockListen(). 04/13/07 04:04:20 PM: SockListen(). I am imax. My address is 2 04/13/07 04:04:20 PM: SockListen(). calling socket(AF_INET,SOCK_STREAM, 0) 04/13/07 04:04:20 PM: SockListen(). sock_fd=22, backlog=1024 04/13/07 04:04:20 PM: SockListen(). bind() ok, listening on socket. 04/13/07 04:04:20 PM: SockListen(). returning OK 04/13/07 04:04:20 PM: answersock(). port=8028 04/13/07 04:04:20 PM: SockAnswer(). listening on socket. 04/13/07 04:04:20 PM: SockAnswer(). entering accept with sock_fd=22. 04/13/07 04:04:20 PM: pthread_create returns 0, errno=0. 04/13/07 04:04:20 PM: THREAD 0X102 STARTED.

Log file: `job.<PlanID>`
Description: Log of the entire plan
(a snapshot of important / note worthy lines)

Description	Log file
When the nfmjsrv process was started	Thread 0x00000001 fd 5 [nfmjsrv] 04/19/07 03:47:17.920 PM: [nfmjsrv] 04/19/07 03:47:18.148 PM: Entering getfile [nfmjsrv] 04/19/07 03:47:18.149 PM: DB alloc 3004E220, 320 bytes, total=320 [nfmjsrv] 04/19/07 03:47:18.149 PM: freadfile() [nfmjsrv] 04/19/07 03:47:18.149 PM: openfile()
Fcplock() locks the table to prevent any writes	[nfmjsrv] 04/19/07 03:47:18.149 PM: fcplock(), filespec="/var/tps/dba/data/USERPERMS", flags=0 [nfmjsrv] 04/19/07 03:47:18.150 PM: fcplock(), locking on semid=0x1A [nfmjsrv] 04/19/07 03:47:18.150 PM: sementry=0x0 [nfmjsrv] 04/19/07 03:47:18.150 PM: freadfile, lseek took 0 seconds.
Reads the table into memory	[nfmjsrv] 04/19/07 03:47:18.150 PM: DB alloc 3004E390, 14569 bytes, total=14889 [nfmjsrv] 04/19/07 03:47:18.150 PM: freadfile, getmem took 0 seconds.
Fcpunlock() frees up the table	[nfmjsrv] 04/19/07 03:47:18.151 PM: freadfile, read took 0 seconds. [nfmjsrv] 04/19/07 03:47:18.151 PM: closefile() handle=6 [nfmjsrv] 04/19/07 03:47:18.151 PM: fcpunlock(), semid=0x1A [nfmjsrv] 04/19/07 03:47:18.151 PM: Leaving getfile, (0 seconds)

Log file continues to read in other tables Logging turned on (globally) Name of process ** Start of JOB information ** JOB name Plan name Plan ID Plan Start time ** Scheduled options ** History Level Default Source Node Default Target Node Plan History Level Submitted By == NODE record (IMAX)== ======================== = ENVIRONMENT VARIABLES =======================	[nfmjsrv] 04/19/07 03:47:18.151 PM: Entering scanrecs [nfmjsrv] 04/19/07 03:47:18.152 PM: DB alloc 30051CA0, 964 bytes, total=15853 [nfmjsrv] 04/19/07 03:47:18.152 PM: scanning 241 slots [nfmjsrv] 04/19/07 03:47:18.152 PM: Entering sort (5 items) [nfmjsrv] 04/19/07 03:47:18.152 PM: Leaving sort, (0 seconds) [nfmjsrv] 04/19/07 03:47:18.152 PM: DB Free 3004E220, 320 bytes, total=15533 [nfmjsrv] 04/19/07 03:47:18.152 PM: Leaving scanrecs at, (0 seconds) [nfmjsrv] 04/19/07 03:47:18.152 PM: DB alloc 3004E220, 240 bytes, total=15773 [nfmjsrv] 04/19/07 03:47:18.153 PM: DB Free 30051CA0, 964 bytes, total=14809 [nfmjsrv] 04/19/07 03:47:18.153 PM: DB Free 3004E390, 14569 bytes, total=240 <skipping redundant information> [nfmjsrv] 04/19/07 03:47:19.538 PM: global logging is on [nfmjsrv] 04/19/07 03:47:19.538 PM: nfmjsrv [nfmjsrv] 04/19/07 03:47:19.538 PM: Memory allocated = 91086,0,6452,0,0 [nfmjsrv] 04/19/07 03:47:19.538 PM: Memory maximum = 174806,0,6452,0,0 [nfmjsrv] 04/19/07 03:47:19.538 PM: Created shared memory, key=Private Buffer, id=1966099, size=3288 [nfmjsrv] 04/19/07 03:47:19.538 PM: Created gnodes array nodect=2, maxnodes=1, gnode=2784, nodes=3008, group=280, total=3288. [nfmjsrv] 04/19/07 03:47:19.539 PM: NAP array 0 entries. [nfmjsrv] 04/19/07 03:47:19.539 PM: Created shared memory, key=Private Buffer, id=1703961, size=116 [nfmjsrv] 04/19/07 03:47:19.539 PM: Alloc Free 3004F6C0, 116 bytes [nfmjsrv] 04/19/07 03:47:19.539 PM: JOB smallfile [nfmjsrv] 04/19/07 03:47:19.539 PM: Plan : smallfile [nfmjsrv] 04/19/07 03:47:19.539 PM: Id : 933 [nfmjsrv] 04/19/07 03:47:19.540 PM: Start time : Thu Apr 19 15:47:16 2007 [nfmjsrv] 04/19/07 03:47:19.540 PM: Recursion interval: 0 [nfmjsrv] 04/19/07 03:47:19.540 PM: Recursion count : 0 [nfmjsrv] 04/19/07 03:47:19.540 PM: Day table : [nfmjsrv] 04/19/07 03:47:19.540 PM: Hour table : [nfmjsrv] 04/19/07 03:47:19.540 PM: Initial state : 0 [nfmjsrv] 04/19/07 03:47:19.540 PM: Follow job id : 0 [nfmjsrv] 04/19/07 03:47:19.540 PM: History : 0 [nfmjsrv] 04/19/07 03:47:19.540 PM: Default src node : IMAX [nfmjsrv] 04/19/07 03:47:19.540 PM: Default trg node : HERBIE

===================== CONNECTIONS	[nfmjsrv] 04/19/07 03:47:19.540 PM: Plan history : 2
=====================	[nfmjsrv] 04/19/07 03:47:19.540 PM: Submitted by : root
	[nfmjsrv] 04/19/07 03:47:19.541 PM: ---
	[nfmjsrv] 04/19/07 03:47:19.541 PM: NODES
== NODE record	[nfmjsrv] 04/19/07 03:47:19.541 PM: ---

(HERBIE)==	[nfmjsrv] 04/19/07 03:47:19.541 PM: Number: 1
	Name=IMAX All index=52 Perms=F
	Model=AIX_SOCKETS Opsys=AIX

= ENVIRONMENT VARIABLES	[nfmjsrv] 04/19/07 03:47:19.541 PM: Environment variables:
======================= =	[nfmjsrv] 04/19/07 03:47:19.541 PM: [0] NTPHONEBOOK=N
	[nfmjsrv] 04/19/07 03:47:19.541 PM: [1] NFMEMPTY=
	[nfmjsrv] 04/19/07 03:47:19.541 PM: [2] NEW4=4
	[nfmjsrv] 04/19/07 03:47:19.541 PM: [3] NEW3=3
	[nfmjsrv] 04/19/07 03:47:19.541 PM: [4] NEW2=2
	[nfmjsrv] 04/19/07 03:47:19.541 PM: [5] NEW1=1
	[nfmjsrv] 04/19/07 03:47:19.541 PM: [6] CREATETMP=mkdir $(TMPDIR)
===================== CONNECTIONS	[nfmjsrv] 04/19/07 03:47:19.542 PM: [7] brad2=hello IMAX
=====================	[nfmjsrv] 04/19/07 03:47:19.542 PM: [8] _TEMPLOGS=5
	[nfmjsrv] 04/19/07 03:47:19.542 PM: [9] brad1=hello
	[nfmjsrv] 04/19/07 03:47:19.542 PM: [10] brad3=hello
	[nfmjsrv] 04/19/07 03:47:19.542 PM: [11] DEFDIR=/home/
	[nfmjsrv] 04/19/07 03:47:19.542 PM: Connections:
** Plan Information Starts **	[nfmjsrv] 04/19/07 03:47:19.542 PM: Def Type=0 Port=8008 Ssl=0
	[nfmjsrv] 04/19/07 03:47:19.542 PM: Def ssl Type=0 Port=8009 Ssl=1
Phase 1	[nfmjsrv] 04/19/07 03:47:19.542 PM: ---

	[nfmjsrv] 04/19/07 03:47:19.542 PM: Number: 2
	Name=HERBIE All index=41 Perms=F
	Model=AIX_SOCKETS Opsys=AIX

Fileset name	[nfmjsrv] 04/19/07 03:47:19.542 PM: Environment variables:
Files	[nfmjsrv] 04/19/07 03:47:19.542 PM: [0] WOOFIE=woofie9
Logging turned on for the nodes	[nfmjsrv] 04/19/07 03:47:19.543 PM: [1] TMPDIR=/home/brad/$(WOOFIE)
	[nfmjsrv] 04/19/07 03:47:19.543 PM: [2] CREATETMP=mkdir $(TMPDIR)
	[nfmjsrv] 04/19/07 03:47:19.543 PM: [3] brad1=hello
	[nfmjsrv] 04/19/07 03:47:19.543 PM: [4] brad2=hello
	[nfmjsrv] 04/19/07 03:47:19.543 PM: [5] brad3=hello
	[nfmjsrv] 04/19/07 03:47:19.543 PM: [6] DEFDIR=/home/
	[nfmjsrv] 04/19/07 03:47:19.543 PM: Connections:
Fcplock() locks the	[nfmjsrv] 04/19/07 03:47:19.543 PM: Def Type=0 Port=8008 Ssl=0
AUDITS table	[nfmjsrv] 04/19/07 03:47:19.543 PM: Def ssl Type=0 Port=8009 Ssl=1
	[nfmjsrv] 04/19/07 03:47:19.543 PM: ---
	[nfmjsrv] 04/19/07 03:47:19.545 PM: Plan phases & functions

	[nfmjsrv] 04/19/07 03:47:19.545 PM: --
	[nfmjsrv] 04/19/07 03:47:19.545 PM:
	[nfmjsrv] 04/19/07 03:47:19.545 PM: Phase "Phase1", 1 functions, statoff 0, maxnodes 1
	[nfmjsrv] 04/19/07 03:47:19.545 PM: Function #0, [Transfer] Syncpoint=yes
	[nfmjsrv] 04/19/07 03:47:19.545 PM: Source node(s): (nodeindex:napindex)
Fcplock() locks the AUDITS table (Julian Date)	[nfmjsrv] 04/19/07 03:47:19.545 PM: 1:0,
	[nfmjsrv] 04/19/07 03:47:19.546 PM: Target node(s): (nodeindex:napindex)
	[nfmjsrv] 04/19/07 03:47:19.546 PM: 2:0,
	[nfmjsrv] 04/19/07 03:47:19.546 PM: Fileset:smallfile
	[nfmjsrv] 04/19/07 03:47:19.546 PM: [0]file:woof.txt
	[nfmjsrv] 04/19/07 03:47:19.546 PM: NTPHONEBOOK=N.
	[nfmjsrv] 04/19/07 03:47:19.546 PM: The following 2 nodes used by this plan are running in diagnostic mode: IMAX HERBIE
	[nfmjsrv] 04/19/07 03:47:19.546 PM: dbput_id, offset=0
	[nfmjsrv] 04/19/07 03:47:19.546 PM: DB alloc 3004F7A0, 320 bytes, total=5648
	[nfmjsrv] 04/19/07 03:47:19.546 PM: freadfile()
	[nfmjsrv] 04/19/07 03:47:19.546 PM: openfile()
Fcplock() JOBS table	[nfmjsrv] 04/19/07 03:47:19.546 PM: fcplock(), filespec="/var/tps/dba/data/AUDITS", flags=0
	[nfmjsrv] 04/19/07 03:47:19.547 PM: fcplock(), locking on semid=0x9
	[nfmjsrv] 04/19/07 03:47:19.547 PM: sementry=0x0
	[nfmjsrv] 04/19/07 03:47:19.547 PM: DB alloc 30050DD0, 1472 bytes, total=7120
	[nfmjsrv] 04/19/07 03:47:19.547 PM: Writing unique id 32752
	[nfmjsrv] 04/19/07 03:47:19.548 PM: closefile() handle=6
	[nfmjsrv] 04/19/07 03:47:19.548 PM: fcpunlock(), semid=0x9
	[nfmjsrv] 04/19/07 03:47:19.548 PM: DB Free 30050DD0, 1472 bytes, total=5648
	[nfmjsrv] 04/19/07 03:47:19.548 PM: DB Free 3004F7A0, 320 bytes, total=5328
	[nfmjsrv] 04/19/07 03:47:19.548 PM: DB alloc 3004F7A0, 320 bytes, total=5648
	[nfmjsrv] 04/19/07 03:47:19.548 PM: freadfile()
	[nfmjsrv] 04/19/07 03:47:19.548 PM: openfile()
	[nfmjsrv] 04/19/07 03:47:19.548 PM: fcplock(), filespec="/var/tps/dba/data/AUDITS.2007.109", flags=0
	[nfmjsrv] 04/19/07 03:47:19.548 PM: fcplock(), locking on semid=0x3E
Restricts number of active nodes if 'Maximum Active Nodes' set.	[nfmjsrv] 04/19/07 03:47:19.548 PM: sementry=0x0
	[nfmjsrv] 04/19/07 03:47:19.549 PM: DB alloc 30050DD0, 1472 bytes, total=7120
	[nfmjsrv] 04/19/07 03:47:19.549 PM: DB alloc 3004F630, 177 bytes, total=7297
	[nfmjsrv] 04/19/07 03:47:19.549 PM: DB Free 3004F630, 177 bytes, total=7120
Separate thread started	[nfmjsrv] 04/19/07 03:47:19.561 PM: closefile() handle=6

for log job.<PlanID>.Phase1 Free up Database files	[nfmjsrv] 04/19/07 03:47:19.562 PM: fcpunlock(), semid=0x3E [nfmjsrv] 04/19/07 03:47:19.562 PM: DB Free 30050DD0, 1472 bytes, total=5648 [nfmjsrv] 04/19/07 03:47:19.562 PM: DB Free 3004F7A0, 320 bytes, total=5328 [nfmjsrv] 04/19/07 03:47:19.587 PM: DB alloc 3004F7A0, 320 bytes, total=5648 [nfmjsrv] 04/19/07 03:47:19.587 PM: freadfile() [nfmjsrv] 04/19/07 03:47:19.587 PM: openfile() [nfmjsrv] 04/19/07 03:47:19.587 PM: fcplock(), filespec="/var/tps/dba/data/JOBS", flags=0 [nfmjsrv] 04/19/07 03:47:19.587 PM: fcplock(), locking on semid=0x11 [nfmjsrv] 04/19/07 03:47:19.587 PM: sementry=0x0 [nfmjsrv] 04/19/07 03:47:19.588 PM: freadfile, lseek took 0 seconds. [nfmjsrv] 04/19/07 03:47:19.588 PM: DB alloc 30050DD0, 9067 bytes, total=14715 [nfmjsrv] 04/19/07 03:47:19.588 PM: freadfile, getmem took 0 seconds. [nfmjsrv] 04/19/07 03:47:19.588 PM: freadfile, read took 0 seconds. [nfmjsrv] 04/19/07 03:47:19.588 PM: Entering upindexes [nfmjsrv] 04/19/07 03:47:19.588 PM: DB alloc 3004F250, 120 bytes, total=14835 [nfmjsrv] 04/19/07 03:47:19.588 PM: Leaving upindexes, (0 seconds) [nfmjsrv] 04/19/07 03:47:19.589 PM: DB alloc 3004F910, 349 bytes, total=15184 [nfmjsrv] 04/19/07 03:47:19.589 PM: DB Free 3004F910, 349 bytes, total=14835 [nfmjsrv] 04/19/07 03:47:19.589 PM: closefile() handle=6 [nfmjsrv] 04/19/07 03:47:19.589 PM: fcpunlock(), semid=0x11 [nfmjsrv] 04/19/07 03:47:19.589 PM: DB Free 3004F250, 120 bytes, total=14715 [nfmjsrv] 04/19/07 03:47:19.589 PM: DB Free 30050DD0, 9067 bytes, total=5648 [nfmjsrv] 04/19/07 03:47:19.589 PM: DB Free 3004F7A0, 320 bytes, total=5328 [nfmjsrv] 04/19/07 03:47:19.589 PM: Initializing maximum node semaphores <skipping redundant information> [nfmjsrv] 04/19/07 03:47:19.817 PM: Alloc 3004F250, 20 bytes [nfmjsrv] 04/19/07 03:47:19.817 PM: childpipe created fd[0]=6 fd[1]=7 [nfmjsrv] 04/19/07 03:47:19.818 PM: nfm_startthread(), fun=30031C98, s=2FF21CE8, ssize=3680, logdir=/var/tps/nfm/logs, logname=job.00933.Phase1, pflag=0, childidp=3004F254, childc=3004F278, async=0 [nfmjsrv] 04/19/07 03:47:19.819 PM: about to start thread. [nfmjsrv] 04/19/07 03:47:19.820 PM: pthread_create returns 0, errno=0. [nfmjsrv] 04/19/07 03:47:19.820 PM: thread 0x102 started. [nfmjsrv] 04/19/07 03:47:19.820 PM: Waiting on children task [nfmjsrv] 04/19/07 03:47:19.821 PM: nfm_wait(),

```
waits=3004F250, waitct=1, childlist=2FF21534,
eventcp=2FF21538, timeout=-1, allflag=0
[nfmjsrv] 04/19/07 03:47:19.821 PM: (0)=102
[nfmjsrv] 04/19/07 03:47:19.821 PM:    selectflag   =
1
[nfmjsrv] 04/19/07 03:47:19.821 PM:    childrenflag =
1
[nfmjsrv] 04/19/07 03:47:19.821 PM:    fdhigh       =
7
[nfmjsrv] 04/19/07 03:47:19.821 PM:    selecting on
child pipe fd 6
[nfmjsrv] 04/19/07 03:47:19.821 PM:    select(size=7,
timevl=0)
[nfmjsrv] 04/19/07 03:47:20.479 PM:    select, rc=1,
errno=0
[nfmjsrv] 04/19/07 03:47:20.480 PM: childpipe read,
rc=8, errno=0
[nfmjsrv] 04/19/07 03:47:20.480 PM: process pipe,
size=8, read=102,0, first=0,0
[nfmjsrv] 04/19/07 03:47:20.480 PM: childpipe read,
rc=0, errno=0
[nfmjsrv] 04/19/07 03:47:20.480 PM: nfm_wait() exits,
eventct=1
[nfmjsrv] 04/19/07 03:47:20.480 PM: Waiting on
children task
[nfmjsrv] 04/19/07 03:47:20.480 PM: nfm_wait(),
waits=3004F250, waitct=1, childlist=2FF21534,
eventcp=2FF21538, timeout=-1, allflag=0
[nfmjsrv] 04/19/07 03:47:20.480 PM: (0)=102
[nfmjsrv] 04/19/07 03:47:20.480 PM:    selectflag   =
0
[nfmjsrv] 04/19/07 03:47:20.480 PM:    childrenflag =
0
[nfmjsrv] 04/19/07 03:47:20.481 PM:    fdhigh       =
1
[nfmjsrv] 04/19/07 03:47:20.481 PM: nfm_wait() exits,
eventct=0
[nfmjsrv] 04/19/07 03:47:20.481 PM: Waiting on
children task
[nfmjsrv] 04/19/07 03:47:20.481 PM: nfm_wait(),
waits=3004F250, waitct=1, childlist=2FF21534,
eventcp=0, timeout=-1, allflag=1
[nfmjsrv] 04/19/07 03:47:20.481 PM: (0)=102
[nfmjsrv] 04/19/07 03:47:20.481 PM:    selectflag   =
0
[nfmjsrv] 04/19/07 03:47:20.481 PM:    childrenflag =
0
[nfmjsrv] 04/19/07 03:47:20.481 PM:    fdhigh       =
1
[nfmjsrv] 04/19/07 03:47:20.481 PM: nfm_wait() exits,
eventct=0
[nfmjsrv] 04/19/07 03:47:20.511 PM: Node IMAX's
elapsed time:  1 seconds.  Bytes sent: 99
[nfmjsrv] 04/19/07 03:47:20.511 PM: dbput_id, offset=0
[nfmjsrv] 04/19/07 03:47:20.511 PM: DB alloc 30050DD0,
320 bytes, total=5888
[nfmjsrv] 04/19/07 03:47:20.511 PM: freadfile()
[nfmjsrv] 04/19/07 03:47:20.511 PM: openfile()
[nfmjsrv] 04/19/07 03:47:20.512 PM: fcplock(),
filespec="/var/tps/dba/data/AUDITS", flags=0
[nfmjsrv] 04/19/07 03:47:20.512 PM: fcplock(), locking
on semid=0x9
[nfmjsrv] 04/19/07 03:47:20.512 PM: sementry=0x0
[nfmjsrv] 04/19/07 03:47:20.512 PM: DB alloc 30050F40,
1472 bytes, total=7360
[nfmjsrv] 04/19/07 03:47:20.513 PM: Writing unique id
32756
[nfmjsrv] 04/19/07 03:47:20.513 PM: closefile()
handle=8
[nfmjsrv] 04/19/07 03:47:20.513 PM: fcpunlock(),
semid=0x9
[nfmjsrv] 04/19/07 03:47:20.513 PM: DB Free 30050F40,
1472 bytes, total=5888
[nfmjsrv] 04/19/07 03:47:20.513 PM: DB Free 30050DD0,
320 bytes, total=5568
[nfmjsrv] 04/19/07 03:47:20.513 PM: DB alloc 30050DD0,
320 bytes, total=5888
           <skipping redundant information>

 [nfmjsrv] 04/19/07 03:47:20.581 PM: DB Free 3004FB50,
120 bytes, total=14955
[nfmjsrv] 04/19/07 03:47:20.581 PM: DB Free 30050DD0,
9067 bytes, total=5888
[nfmjsrv] 04/19/07 03:47:20.581 PM: DB Free 3004F9E0,
320 bytes, total=5568
[nfmjsrv] 04/19/07 03:47:20.582 PM: Freech 305E6670,
280 bytes
[nfmjsrv] 04/19/07 03:47:20.582 PM: Freech 305E6070,
1204 bytes
[nfmjsrv] 04/19/07 03:47:20.582 PM: Freech 305E5CA0,
928 bytes
[nfmjsrv] 04/19/07 03:47:20.582 PM: Freech 305E5C60,
16 bytes
[nfmjsrv] 04/19/07 03:47:20.582 PM: Freech
305E57D0, 672 bytes
[nfmjsrv] 04/19/07 03:47:20.582 PM: Freech 305E5710,
144 bytes
[nfmjsrv] 04/19/07 03:47:20.582 PM: Freech 305E54E0,
512 bytes
[nfmjsrv] 04/19/07 03:47:20.582 PM: Freech 305E4E80,
1204 bytes
[nfmjsrv] 04/19/07 03:47:20.582 PM: Freech 305E4AB0,
928 bytes
[nfmjsrv] 04/19/07 03:47:20.582 PM: Freech 3004F760,
16 bytes
[nfmjsrv] 04/19/07 03:47:20.583 PM: Freech 305E48D0,
432 bytes
[nfmjsrv] 04/19/07 03:47:20.583 PM: Removed shared
memory, id = 1703961
[nfmjsrv] 04/19/07 03:47:20.583 PM: Removed shared
memory, id = 1966099
[nfmjsrv] 04/19/07 03:47:20.583 PM: Removed shared
memory, id = 1179674
[nfmjsrv] 04/19/07 03:47:20.583 PM: Server node IMAX:
TEMPLOGS=nothing
```

Log file: `job.<PlanID>.<Phase name>`
Description: Log of the plan's phase
(similar in layout to `job.<PlanID> logfile`)

Log file: `job.<PlanID>.<Phase name>.<nodename>`

Description: Log of the plan's phase related to an individual node
(a snapshot of important / note worthy lines)

Description	Log file
	Thread 0x00000203 fd 11
When the process was started	04/19/07 03:47:19.882 PM: Node process/thread starts, id=515 gid=24868
	04/19/07 03:47:19.903 PM: NFM Software version 2.3.0.0
	04/19/07 03:47:19.903 PM: runphase_node() funi=0
	04/19/07 03:47:19.903 PM: groupstat=0, snode=A00003A0, tnode=A0000740, lnode=0
	04/19/07 03:47:19.903 PM: runfunction(), rfuni=0, function=0, snode=A00003A0, tnode=A0000740, skip=0
Fileset name	04/19/07 03:47:19.904 PM: Transferring fileset smallfile from IMAX to HERBIE
	04/19/07 03:47:19.904 PM: holdpoint() snode=A00003A0, tnode=A0000740, busy=0
	04/19/07 03:47:19.904 PM: buildnfctls()
	04/19/07 03:47:19.905 PM: meminit()
	04/19/07 03:47:19.905 PM: Alloc 300B6EB0, 64 bytes
	04/19/07 03:47:19.905 PM: snode=A00003A0
	04/19/07 03:47:19.905 PM: Alloc 300B6F20, 24 bytes
	04/19/07 03:47:19.905 PM: 0: dirinfo src=woof.txt, trg=woof.txt
	04/19/07 03:47:19.905 PM: 0: srcfile=woof.txt
	04/19/07 03:47:19.905 PM: makeremotelogname 30093A24, 90000000 300942EC 300B1564
	04/19/07 03:47:19.906 PM: Alloc 300B6F60, 86 bytes
	04/19/07 03:47:19.906 PM: Alloc 300B6FE0, 110 bytes
	04/19/07 03:47:19.906 PM: nservosock() IMAX
	04/19/07 03:47:19.906 PM: opensock(). comname=204.62.234.86, port=8008, encrypt=0, priority=1, timeout=0, keepalive=0
	04/19/07 03:47:19.907 PM: SockOpen(). spec was x.x.x.x format. Addr=cc3eea56
	04/19/07 03:47:19.907 PM: SockOpen(). calling socket(AF_INET,SOCK_STREAM,0)
	04/19/07 03:47:19.907 PM: SockOpen(). sock_fd=12
	04/19/07 03:47:19.909 PM: SockOpen(). returning OK
=====================	04/19/07 03:47:19.909 PM: Outgoing (formatted):
NFM Header Information	04/19/07 03:47:19.909 PM: NFM message (formatted)
=====================	04/19/07 03:47:19.909 PM: NFM message (formatted)
	04/19/07 03:47:19.909 PM: HDR
	04/19/07 03:47:19.910 PM: signature : NFMC
	04/19/07 03:47:19.910 PM: old version : 1010
	04/19/07 03:47:19.910 PM: version : 1012
Type of message	04/19/07 03:47:19.910 PM: msglen : 110
	04/19/07 03:47:19.910 PM: hdrlen : 24
	04/19/07 03:47:19.910 PM: msgtype : Request
	04/19/07 03:47:19.910 PM: abilities1 : 0x3 PACING SSL
	04/19/07 03:47:19.910 PM: abilities2 : 0x0
	04/19/07 03:47:19.910 PM: mblknum : 0
Message Request	04/19/07 03:47:19.910 PM: acknum : 0
	04/19/07 03:47:19.911 PM: flags1 : 0x0
	04/19/07 03:47:19.911 PM: flags2 : 0x0
Request Type (Directory Info)	04/19/07 03:47:19.911 PM: REQUEST
	04/19/07 03:47:19.911 PM: hdrlen : 62
	04/19/07 03:47:19.911 PM: complen : 86
	04/19/07 03:47:19.911 PM: reqtype : Directory info
	04/19/07 03:47:19.911 PM: rflags1 : 0x28 TRACEON

```
                        TEMPFILES
                        04/19/07 03:47:19.911 PM:   rflags2  : 0x0
                        04/19/07 03:47:19.912 PM:   priority : 0x0
      Directory         04/19/07 03:47:19.912 PM:   rflags3  : 0x0
         File           04/19/07 03:47:19.912 PM:   rflags4  : 0x0
                        04/19/07 03:47:19.912 PM:   logprefix:
  Binary Header Info    smallfil.00933.000
                        04/19/07 03:47:19.912 PM:   arg0     : /prod/nfmtest/
                        04/19/07 03:47:19.912 PM:   arg1     : woof.txt
                        04/19/07 03:47:19.912 PM: Outgoing (raw):
                        04/19/07 03:47:19.912 PM: 0000:   4E46 4D43 03F2 0000
                        006E 0018 03F4 0300    |NFMC.....n......|
                        04/19/07 03:47:19.912 PM: 0010:   0001 0000 0000 0000
                        0001 003E 0000 0056    |...........>...V|
                        04/19/07 03:47:19.913 PM: 0020:   000F 2800 736D 616C
                        6C66 696C 2E30 3039    |..(.smallfil.009|
                        04/19/07 03:47:19.913 PM: 0030:   3333 2E30 3030 0000
                        0000 0000 0000 0000    |33.000..........|
                        04/19/07 03:47:19.913 PM: 0040:   0000 0000 00AF 477F
                        CE7F B4A9 7DB6 EDC3    |......G.....}...|
                        04/19/07 03:47:19.913 PM: 0050:   9CC9 6C50 1D00 2F70
                        726F 642F 6E66 6D74    |..lP../prod/nfmt|
                        04/19/07 03:47:19.913 PM: 0060:   6573 742F 0077 6F6F
                        662E 7478 7400         |est/.woof.txt.  |
                        04/19/07 03:47:19.913 PM: nfm_poll(fd=12, events=0x2,
                        to=100000)
                        04/19/07 03:47:19.913 PM: nfm_poll() before poll
                        04/19/07 03:47:19.914 PM: nfm_poll() after poll
                        04/19/07 03:47:19.914 PM: SockSnd(110 bytes).
                        04/19/07 03:47:19.914 PM: Entering SockRcv().
                        buflen=18, red=0, sock=12, ssl=0
                        04/19/07 03:47:19.914 PM: nfm_poll(fd=12, events=0x1,
                        to=100000)
                        04/19/07 03:47:19.918 PM: nfm_poll() before poll
                        04/19/07 03:47:19.949 PM: nfm_poll() after poll
                        04/19/07 03:47:19.949 PM: recv returned 18 bytes
                        04/19/07 03:47:19.949 PM: Alloc 300BB090, 96 bytes
                        04/19/07 03:47:19.949 PM: Entering SockRcv().
                        buflen=74, red=0, sock=12, ssl=0
  Binary Header Info    04/19/07 03:47:19.950 PM: nfm_poll(fd=12, events=0x1,
                        to=100000)
                        04/19/07 03:47:19.950 PM: nfm_poll() before poll
                        04/19/07 03:47:19.950 PM: nfm_poll() after poll
                        04/19/07 03:47:19.950 PM: recv returned 74 bytes
                        04/19/07 03:47:19.950 PM: In msgtype 2
                        04/19/07 03:47:19.951 PM: Incoming (raw):
                        04/19/07 03:47:19.951 PM: 0000:   4E46 4D43 03F2 0000
                        005C 0018 03F4 0300    |NFMC.....\......|
                        04/19/07 03:47:19.951 PM: 0010:   0002 0000 0000 0000
                        0001 003E 0000 0044    |...........>...D|
                        04/19/07 03:47:19.951 PM: 0020:   000F 0000 0000 0000
                        0000 0000 0000 0000    |................|
                        04/19/07 03:47:19.951 PM: 0030:   0000 0000 0000 0000
======================  0000 0000 0000 0000    |................|
NFM Header Information  04/19/07 03:47:19.951 PM: 0040:   0000 0000 0000 0000
======================  0000 0000 0000 0000    |................|
                        04/19/07 03:47:19.951 PM: 0050:   0000 0000 0000 3000
                        0039 3900              |......0..99.    |
                        04/19/07 03:47:19.951 PM: Incoming (formatted):
                        04/19/07 03:47:19.952 PM: NFM message (formatted)
     Message Type       04/19/07 03:47:19.952 PM: NFM message (formatted)
                        04/19/07 03:47:19.952 PM:  HDR
                        04/19/07 03:47:19.952 PM:   signature    : NFMC
                        04/19/07 03:47:19.952 PM:   old version  : 1010
```

	04/19/07 03:47:19.952 PM: version : 1012
	04/19/07 03:47:19.952 PM: msglen : 92
	04/19/07 03:47:19.952 PM: hdrlen : 24
Message Reply	04/19/07 03:47:19.952 PM: msgtype : Reply
	04/19/07 03:47:19.952 PM: abilities1 : 0x3 PACING SSL
Request Type (Directory Info)	04/19/07 03:47:19.953 PM: abilities2 : 0x0
	04/19/07 03:47:19.953 PM: mblknum : 0
	04/19/07 03:47:19.953 PM: acknum : 0
	04/19/07 03:47:19.953 PM: flags1 : 0x0
	04/19/07 03:47:19.953 PM: flags2 : 0x0
	04/19/07 03:47:19.953 PM: REPLY
	04/19/07 03:47:19.953 PM: hdrlen : 62
	04/19/07 03:47:19.953 PM: complen : 68
	04/19/07 03:47:19.953 PM: reqtype : Directory info
	04/19/07 03:47:19.954 PM: rflags1 : 0x0
	04/19/07 03:47:19.954 PM: rflags2 : 0x0
	04/19/07 03:47:19.954 PM: priority : 0x0
	04/19/07 03:47:19.954 PM: rflags3 : 0x0
	04/19/07 03:47:19.954 PM: rflags4 : 0x0
	04/19/07 03:47:19.954 PM: logprefix:
	04/19/07 03:47:19.954 PM: arg0 : 0
	04/19/07 03:47:19.954 PM: arg1 :
	04/19/07 03:47:19.954 PM: arg2 : 99
	04/19/07 03:47:19.955 PM: closesock(). handle=12
	04/19/07 03:47:19.955 PM: SockClose(sockh=12).
	04/19/07 03:47:19.958 PM: calling closesocket(12)
	04/19/07 03:47:19.959 PM: SockClose(). returning OK
	04/19/07 03:47:19.959 PM: Freech 300B6FE0, 110 bytes
	04/19/07 03:47:19.959 PM: Freech 300B6F60, 86 bytes
	04/19/07 03:47:19.959 PM: sclienterr, reqtype=15, wftp=0
	04/19/07 03:47:19.959 PM: getclienterr, jobrc=0, nfmrc=0, sysrc=0, error=NULL
	04/19/07 03:47:19.959 PM: Alloc Free 300B6F20, 24 bytes
	04/19/07 03:47:19.959 PM: dirinfo yielded 1 matching entries from 3 arguments
	04/19/07 03:47:19.960 PM: Alloc 300B6F20, 64 bytes
	04/19/07 03:47:19.960 PM: Alloc 300B6F90, 16 bytes
	04/19/07 03:47:19.960 PM: 0: index=0, name=, size=99.
Mark position of plan	04/19/07 03:47:19.960 PM: Alloc Free 300B6EB0, 64 bytes
in case of failure	04/19/07 03:47:19.960 PM: buildnfctls() exit
	04/19/07 03:47:19.960 PM: filefunction() node: HERBIE sfile: woof.txt tfile: woof.txt
	04/19/07 03:47:19.961 PM: holdpoint() snode=A00003A0, tnode=A0000740, busy=0
	04/19/07 03:47:19.961 PM: fun1
	04/19/07 03:47:19.961 PM: Save tracking, rc=0, filename=woof.txt, offset=0, complete=0
	04/19/07 03:47:19.961 PM: nfm open() filename=/var/tps/nfm/tracking/933.Phase1.0.HERBIE, oflag=0x102, mode=0x1B6
Logging turn on for remote client Remote log name	04/19/07 03:47:19.979 PM: nfm_open(), file=/var/tps/nfm/tracking/933.Phase1.0.HERBIE, fileid=12
	04/19/07 03:47:19.980 PM: nfm_write(), fileid=12, size=152, wsize=152
	04/19/07 03:47:19.980 PM: nfm_close(), fileid=0xC
	04/19/07 03:47:19.980 PM: fun2
	04/19/07 03:47:19.980 PM: fun3 0
	04/19/07 03:47:19.980 PM: Transfer file
	04/19/07 03:47:19.981 PM: Transfer file type=bin

```
                          04/19/07 03:47:19.981 PM:   Sockets transfer
                          /prod/nfmtest/woof.txt from IMAX to HERBIE
                          04/19/07 03:47:19.981 PM: makeremotelogname 30093A24,
                          90000000 300942EC 300B130C
                          04/19/07 03:47:19.981 PM: remote
                          logname=smallfil.00933.000
                          04/19/07 03:47:19.981 PM: Alloc 300B6FD0, 108 bytes
                          04/19/07 03:47:19.981 PM: Alloc 300BB120, 132 bytes
                          04/19/07 03:47:19.981 PM: Alloc 300BB1D0, 137 bytes
                          04/19/07 03:47:19.982 PM: append = 0, file=0, tot=1
                          04/19/07 03:47:19.982 PM: Alloc 300BB280, 145 bytes
   ===================    04/19/07 03:47:19.982 PM: Alloc 300BB340, 306 bytes
     NFM Header Info      04/19/07 03:47:19.982 PM: nservosock() HERBIE
   ===================    04/19/07 03:47:19.982 PM: opensock(). comname=herbie,
                          port=8008, encrypt=0, priority=1, timeout=0,
                          keepalive=0
                          04/19/07 03:47:19.982 PM: SockOpen(). spec was not
                          x.x.x.x format.
       Message Type       04/19/07 03:47:19.991 PM: SockOpen(). gethostbyname():
                          herbie found. Addr=300d0830
                          04/19/07 03:47:19.991 PM: SockOpen(). calling
                          socket(AF_INET,SOCK_STREAM,0)
                          04/19/07 03:47:19.991 PM: SockOpen(). sock_fd=12
                          04/19/07 03:47:19.993 PM: SockOpen(). returning OK
                          04/19/07 03:47:19.993 PM: Outgoing (formatted):
 Message Type (Request)   04/19/07 03:47:19.993 PM: NFM message (formatted)
                          04/19/07 03:47:19.993 PM: NFM message (formatted)
                          04/19/07 03:47:19.993 PM:  HDR
 Request Type (Transfer   04/19/07 03:47:19.994 PM:   signature     : NFMC
        Receive)          04/19/07 03:47:19.994 PM:   old version   : 1010
                          04/19/07 03:47:19.994 PM:   version       : 1012
                          04/19/07 03:47:19.994 PM:   msglen        : 306
                          04/19/07 03:47:19.994 PM:   hdrlen        : 24
                          04/19/07 03:47:19.994 PM:   msgtype       : Request
      Log file name       04/19/07 03:47:19.994 PM:   abilities1    : 0x3 PACING
      IP / Hostname       SSL
          File            04/19/07 03:47:19.994 PM:   abilities2    : 0x0
                          04/19/07 03:47:19.994 PM:   mblknum       : 0
                          04/19/07 03:47:19.995 PM:   acknum        : 0
       Block size         04/19/07 03:47:19.995 PM:   flags1        : 0x0
                          04/19/07 03:47:19.995 PM:   flags2        : 0x0
                          04/19/07 03:47:19.995 PM:  REQUEST
                          04/19/07 03:47:19.995 PM:   hdrlen   : 62
                          04/19/07 03:47:19.995 PM:   complen  : 137
      TCP/IP Port         04/19/07 03:47:19.995 PM:   reqtype  : Transfer
                          Retrieve
                          04/19/07 03:47:19.995 PM:   rflags1  : 0x28 TRACEON
                          TEMPFILES
                          04/19/07 03:47:19.996 PM:   rflags2  : 0x0
                          04/19/07 03:47:19.996 PM:   priority : 0x0
                          04/19/07 03:47:19.996 PM:   rflags3  : 0x40 EXTATTRS
                          04/19/07 03:47:19.996 PM:   rflags4  : 0x0
                          04/19/07 03:47:19.996 PM:   logprefix:
                          smallfil.00933.000
                          04/19/07 03:47:19.996 PM:   arg0     : 204.62.234.86
                          04/19/07 03:47:19.996 PM:   arg1     :
                          /prod/nfmtest/woof.txt
        File data         04/19/07 03:47:19.996 PM:   arg2     : 99
                          04/19/07 03:47:19.996 PM:   arg3     :          0
                          04/19/07 03:47:19.997 PM:   arg4     : 32000
                          04/19/07 03:47:19.997 PM:   arg5     :
                          04/19/07 03:47:19.997 PM:   arg6     : 0
                          04/19/07 03:47:19.997 PM:   arg7     : 0
                          04/19/07 03:47:19.997 PM:   arg8     : 32000
```

Incoming Response

===================

```
04/19/07 03:47:19.997 PM:   arg9      : 8008
04/19/07 03:47:19.997 PM:   arg10      :
04/19/07 03:47:19.997 PM:   arg11      :
04/19/07 03:47:19.997 PM:  ATTACHMENT
04/19/07 03:47:19.997 PM:   hdrlen     : 12
04/19/07 03:47:19.998 PM:   complen    : 145
04/19/07 03:47:19.998 PM:   type      : 3
04/19/07 03:47:19.998 PM:   filename :
04/19/07 03:47:19.998 PM:   size      : 132
04/19/07 03:47:19.998 PM:   flags1    : 0x0
04/19/07 03:47:19.998 PM: ----- Attachment Begins ----
-
04/19/07 03:47:19.998 PM: NFMC•ò
04/19/07 03:47:19.998 PM: ----- Attachment Ends ------
-
04/19/07 03:47:19.998 PM: Outgoing (raw):
04/19/07 03:47:19.999 PM: 0000:   4E46 4D43 03F2 0000
0132 0018 03F4 0300    |NFMC.....2......|
04/19/07 03:47:19.999 PM: 0010:   0001 0000 0000 0000
0001 003E 0000 0089    |...........>....|
04/19/07 03:47:19.999 PM: 0020:   000A 2800 736D 616C
6C66 696C 2E30 3039    |..(.smallfil.009|
04/19/07 03:47:19.999 PM: 0030:   3333 2E30 3030 0000
0000 0000 0000 0000    |33.000..........|
04/19/07 03:47:19.999 PM: 0040:   0000 0000 40AF 477F
CE7F B4A9 7DB6 EDC3    |....@.G.....}...|
04/19/07 03:47:19.999 PM: 0050:   9CC9 6C50 1D00 3230
342E 3632 2E32 3334    |..lP..204.62.234|
04/19/07 03:47:19.999 PM: 0060:   2E38 3600 2F70 726F
642F 6E66 6D74 6573    |.86./prod/nfmtes|
04/19/07 03:47:20.000 PM: 0070:   742F 776F 6F66 2E74
7874 0039 3900 2020    |t/woof.txt.99.  |
04/19/07 03:47:20.000 PM: 0080:   2020 2020 2020 2030
0033 3230 3030 0000    |       0.32000..|
04/19/07 03:47:20.000 PM: 0090:   3000 3000 3332 3030
3000 3830 3038 0000    |0.0.32000.8008..|
04/19/07 03:47:20.000 PM: 00a0:   0000 0300 0C00 0000
9100 0103 0000 4E46    |..............NF|
04/19/07 03:47:20.000 PM: 00b0:   4D43 03F2 0000 0084
0018 03F4 0300 0001    |MC..............|
04/19/07 03:47:20.000 PM: 00c0:   0000 0000 0000 0001
003E 0000 006C 000B    |.........>...l..|
04/19/07 03:47:20.000 PM: 00d0:   2800 736D 616C 6C66
696C 2E30 3039 3333    |(.smallfil.00933|
04/19/07 03:47:20.001 PM: 00e0:   2E30 3030 0000 0000
0000 0000 0000 0000    |.000............|
04/19/07 03:47:20.001 PM: 00f0:   0000 40AF 477F CE7F
B4A9 7DB6 EDC3 9CC9    |..@.G.....}.....|
04/19/07 03:47:20.001 PM: 0100:   6C50 1D00 2F70 726F
642F 6E66 6D74 6573    |lP../prod/nfmtes|
04/19/07 03:47:20.001 PM: 0110:   742F 776F 6F66 2E74
7874 0033 3230 3030    |t/woof.txt.32000|
04/19/07 03:47:20.001 PM: 0120:   0020 2020 2020 2020
2020 3000 3000 3000    |.         0.0.0.|
04/19/07 03:47:20.001 PM: 0130:   3000
|0.               |
04/19/07 03:47:20.001 PM: nfm_poll(fd=12, events=0x2,
to=100000)
04/19/07 03:47:20.001 PM: nfm_poll() before poll
04/19/07 03:47:20.002 PM: nfm_poll() after poll
04/19/07 03:47:20.002 PM: SockSnd(306 bytes).
04/19/07 03:47:20.002 PM: Entering SockRcv().
buflen=18, red=0, sock=12, ssl=0
04/19/07 03:47:20.002 PM: nfm_poll(fd=12, events=0x1,
   NFM Header Info       to=100000)
=====================    04/19/07 03:47:20.002 PM: nfm_poll() before poll
                         04/19/07 03:47:20.358 PM: nfm_poll() after poll
                         04/19/07 03:47:20.358 PM: recv returned 18 bytes
                         04/19/07 03:47:20.359 PM: Alloc 300D3A80, 88 bytes
                         04/19/07 03:47:20.359 PM: Entering SockRcv().
                         buflen=66, red=0, sock=12, ssl=0
                         04/19/07 03:47:20.359 PM: nfm_poll(fd=12, events=0x1,
     Message Type        to=100000)
                         04/19/07 03:47:20.359 PM: nfm_poll() before poll
                         04/19/07 03:47:20.360 PM: nfm_poll() after poll
                         04/19/07 03:47:20.360 PM: recv returned 66 bytes
                         04/19/07 03:47:20.360 PM: In msgtype 4
                         04/19/07 03:47:20.360 PM: Incoming (raw):
                         04/19/07 03:47:20.360 PM: 0000:   4E46 4D43 03F2 0000
Message Type (Status)    0054 0018 03F4 0300   |NFMC.....T......|
                         04/19/07 03:47:20.360 PM: 0010:   0004 0000 0000 0000
                         0005 003C 0000 003C   |...........<...<|
                         04/19/07 03:47:20.360 PM: 0020:   0000 0063 0000 022A
                         0100 0000 0000 0000   |...c...*........|
                         04/19/07 03:47:20.361 PM: 0030:   0000 0085 0000 0000
                         0000 0081 0000 0000   |................|
                         04/19/07 03:47:20.361 PM: 0040:   0000 086E 0000 062A
                         0000 0002 0000 0000   |...n...*........|
                         04/19/07 03:47:20.361 PM: 0050:   0000 0000
                         |....             |
                         04/19/07 03:47:20.361 PM: Incoming (formatted):
                         04/19/07 03:47:20.361 PM: NFM message (formatted)
                         04/19/07 03:47:20.361 PM: NFM message (formatted)
                         04/19/07 03:47:20.361 PM:  HDR
                         04/19/07 03:47:20.362 PM:   signature    : NFMC
                         04/19/07 03:47:20.362 PM:   old version  : 1010
                         04/19/07 03:47:20.362 PM:   version      : 1012
                         04/19/07 03:47:20.362 PM:   msglen       : 84
                         04/19/07 03:47:20.362 PM:   hdrlen       : 24
                         04/19/07 03:47:20.362 PM:   msgtype      : Status
  Binary Header Info     04/19/07 03:47:20.362 PM:   abilities1   : 0x3 PACING
                         SSL
                         04/19/07 03:47:20.362 PM:   abilities2   : 0x0
                         04/19/07 03:47:20.362 PM:   mblknum      : 0
                         04/19/07 03:47:20.362 PM:   acknum       : 0
                         04/19/07 03:47:20.363 PM:   flags1       : 0x0
                         04/19/07 03:47:20.363 PM:   flags2       : 0x0
                         04/19/07 03:47:20.363 PM:  STATUS
                         04/19/07 03:47:20.363 PM:   hdrlen             : 60
                         04/19/07 03:47:20.363 PM:   complen            : 60
                         04/19/07 03:47:20.363 PM:   file offset (low)  : 99
                         04/19/07 03:47:20.363 PM:   file offset (high) : 0
                         04/19/07 03:47:20.363 PM:   trans pid          : 554
                         04/19/07 03:47:20.363 PM:   flags1             : 0x1
                         STATUS
                         04/19/07 03:47:20.363 PM: Entering SockRcv().
                         buflen=18, red=0, sock=12, ssl=0
                         04/19/07 03:47:20.364 PM: nfm_poll(fd=12, events=0x1,
                         to=100000)
                         04/19/07 03:47:20.364 PM: nfm_poll() before poll
                         04/19/07 03:47:20.430 PM: nfm_poll() after poll
                         04/19/07 03:47:20.430 PM: recv returned 18 bytes
                         04/19/07 03:47:20.430 PM: Alloc 300D3B00, 165 bytes
 ===================     04/19/07 03:47:20.430 PM: Entering SockRcv().
   NFM Header Info       buflen=143, red=0, sock=12, ssl=0
 ===================     04/19/07 03:47:20.430 PM: nfm_poll(fd=12, events=0x1,
                         to=100000)
                         04/19/07 03:47:20.431 PM: nfm_poll() before poll
                         04/19/07 03:47:20.431 PM: nfm_poll() after poll
                         04/19/07 03:47:20.431 PM: recv returned 143 bytes
                         04/19/07 03:47:20.431 PM: In msgtype 2
      Message Type       04/19/07 03:47:20.431 PM: Incoming (raw):
                         04/19/07 03:47:20.431 PM: 0000:   4E46 4D43 03F2 0000
                         00A1 0018 03F4 0300   |NFMC............|
                         04/19/07 03:47:20.431 PM: 0010:   0002 0000 0000 0000
                         0001 003E 0000 0089   |..........>....|
                         04/19/07 03:47:20.432 PM: 0020:   000A 2800 736D 616C
                         6C66 696C 2E30 3039   |..(.smallfil.009|
  Message Type (Reply)   04/19/07 03:47:20.432 PM: 0030:   3333 2E30 3030 0000
                         0000 0000 0000 0000   |33.000.........|
                         04/19/07 03:47:20.432 PM: 0040:   0000 0000 40AF 477F
                         CE7F B4A9 7DB6 EDC3   |....@.G.....}...|
                         04/19/07 03:47:20.432 PM: 0050:   9CC9 6C50 1D00 3230
                         342E 3632 2E32 3334   |..lP..204.62.234|
                         04/19/07 03:47:20.432 PM: 0060:   2E38 3600 2F70 726F
                         642F 6E66 6D74 6573   |.86./prod/nfmtes|
                         04/19/07 03:47:20.432 PM: 0070:   742F 776F 6F66 2E74
                         7874 0039 3900 2020   |t/woof.txt.99.  |
                         04/19/07 03:47:20.432 PM: 0080:   2020 2020 2020 2030
                         0033 3230 3030 0000   |       0.32000..|
                         04/19/07 03:47:20.432 PM: 0090:   3000 3000 3332 3030
                         3000 3830 3038 0000   |0.0.32000.8008..|
                         04/19/07 03:47:20.433 PM: 00a0:   00
                         |.                |
                         04/19/07 03:47:20.433 PM: Incoming (formatted):
                         04/19/07 03:47:20.433 PM: NFM message (formatted)
                         04/19/07 03:47:20.433 PM: NFM message (formatted)
                         04/19/07 03:47:20.433 PM:  HDR
                         04/19/07 03:47:20.433 PM:   signature    : NFMC
                         04/19/07 03:47:20.433 PM:   old version  : 1010
                         04/19/07 03:47:20.433 PM:   version      : 1012
                         04/19/07 03:47:20.434 PM:   msglen       : 161
                         04/19/07 03:47:20.434 PM:   hdrlen       : 24
                         04/19/07 03:47:20.434 PM:   msgtype      : Reply
                         04/19/07 03:47:20.434 PM:   abilities1   : 0x3 PACING
                         SSL
                         04/19/07 03:47:20.434 PM:   abilities2   : 0x0
                         04/19/07 03:47:20.434 PM:   mblknum      : 0
                         04/19/07 03:47:20.434 PM:   acknum       : 0
                         04/19/07 03:47:20.434 PM:   flags1       : 0x0
                         04/19/07 03:47:20.434 PM:   flags2       : 0x0
                         04/19/07 03:47:20.435 PM:  REPLY
                         04/19/07 03:47:20.435 PM:   hdrlen   : 62
                         04/19/07 03:47:20.435 PM:   complen  : 137
                         04/19/07 03:47:20.435 PM:   reqtype  : Transfer
                         Retrieve
                         04/19/07 03:47:20.435 PM:   rflags1  : 0x28 TRACEON
                         TEMPFILES
                         04/19/07 03:47:20.435 PM:   rflags2  : 0x0
                         04/19/07 03:47:20.435 PM:   priority : 0x0
   Locks AUDITS file     04/19/07 03:47:20.435 PM:   rflags3  : 0x40 EXTATTRS
                         04/19/07 03:47:20.435 PM:   rflags4  : 0x0
                         04/19/07 03:47:20.436 PM:   logprefix:
                         smallfil.00933.000
                         04/19/07 03:47:20.436 PM:   arg0     : 204.62.234.86
                         04/19/07 03:47:20.436 PM:   arg1     :
                         /prod/nfmtest/woof.txt
                         04/19/07 03:47:20.436 PM:   arg2     : 99
                         04/19/07 03:47:20.436 PM:   arg3     :          0
                         04/19/07 03:47:20.436 PM:   arg4     : 32000
                         04/19/07 03:47:20.436 PM:   arg5     :
                         04/19/07 03:47:20.436 PM:   arg6     : 0
```

Fcplock() locks the AUDITS table (Julian Date)

```
04/19/07 03:47:20.436 PM:   arg7      : 0
04/19/07 03:47:20.437 PM:   arg8      : 32000
04/19/07 03:47:20.437 PM:   arg9      : 8008
04/19/07 03:47:20.437 PM:   arg10      :
04/19/07 03:47:20.437 PM:   arg11      :
04/19/07 03:47:20.437 PM: sclienterr, reqtype=10,
wftp=0
04/19/07 03:47:20.437 PM: clientcall: sclienterr
returned jobrc=0, rnode=0xA00003A0, rnodeflag=0
04/19/07 03:47:20.437 PM: Freech 300BB340, 306 bytes
04/19/07 03:47:20.437 PM: Freech 300BB280, 145 bytes
04/19/07 03:47:20.437 PM: Freech 300BB1D0, 137 bytes
04/19/07 03:47:20.438 PM: Freech 300BB120, 132 bytes
04/19/07 03:47:20.438 PM: Freech 300B6FD0, 108 bytes
04/19/07 03:47:20.438 PM: getpercomp() 0/99 = 0%
04/19/07 03:47:20.438 PM: sclienterr, reqtype=10,
wftp=0
04/19/07 03:47:20.438 PM: getclienterr, jobrc=0,
nfmrc=0, sysrc=0, error=NULL
04/19/07 03:47:20.438 PM: rc=0, inodename=HERBIE,
snode=A00003A0, tnode=A0000740
04/19/07 03:47:20.438 PM: Copied
IMAX:"/prod/nfmtest/woof.txt" to
HERBIE:"/prod/nfmtest/woof.txt".
04/19/07 03:47:20.438 PM: dbput_id, offset=0
```

Mark position of plan in case of failure

```
04/19/07 03:47:20.439 PM: DB alloc 300BB120, 320
bytes, total=320
04/19/07 03:47:20.439 PM: freadfile()
04/19/07 03:47:20.439 PM: openfile()
04/19/07 03:47:20.439 PM: fcplock(),
filespec="/var/tps/dba/data/AUDITS", flags=0
04/19/07 03:47:20.439 PM: fcplock(), locking on
semid=0x9
04/19/07 03:47:20.439 PM: sementry=0x0
04/19/07 03:47:20.440 PM: DB alloc 300D3BD0, 1472
bytes, total=1792
04/19/07 03:47:20.440 PM: Writing unique id 32754
04/19/07 03:47:20.440 PM: closefile() handle=13
04/19/07 03:47:20.440 PM: fcpunlock(), semid=0x9
04/19/07 03:47:20.440 PM: DB Free 300D3BD0, 1472
bytes, total=320
04/19/07 03:47:20.441 PM: DB Free 300BB120, 320 bytes,
total=0
04/19/07 03:47:20.441 PM: DB alloc 300BB120, 320
bytes, total=320
04/19/07 03:47:20.441 PM: freadfile()
04/19/07 03:47:20.441 PM: openfile()
04/19/07 03:47:20.441 PM: fcplock(),
filespec="/var/tps/dba/data/AUDITS.2007.109", flags=0
04/19/07 03:47:20.441 PM: fcplock(), locking on
semid=0x3E
04/19/07 03:47:20.442 PM: sementry=0x0
04/19/07 03:47:20.442 PM: DB alloc 300D3BD0, 1472
bytes, total=1792
04/19/07 03:47:20.442 PM: DB alloc 300BB290, 177
bytes, total=1969
04/19/07 03:47:20.443 PM: DB Free 300BB290, 177 bytes,
total=1792
04/19/07 03:47:20.453 PM: closefile() handle=13
04/19/07 03:47:20.453 PM: fcpunlock(), semid=0x3E
04/19/07 03:47:20.453 PM: DB Free 300D3BD0, 1472
bytes, total=320
04/19/07 03:47:20.453 PM: DB Free 300BB120, 320 bytes,
total=0
```

```
04/19/07 03:47:20.454 PM: fun6
04/19/07 03:47:20.454 PM: fun8
04/19/07 03:47:20.454 PM: fun9
04/19/07 03:47:20.454 PM: fun10 0
04/19/07 03:47:20.454 PM: closesock(). handle=12
04/19/07 03:47:20.454 PM: SockClose(sockh=12).
04/19/07 03:47:20.455 PM: calling closesocket(12)
04/19/07 03:47:20.455 PM: SockClose(). returning OK
04/19/07 03:47:20.455 PM: Alloc Free 300B6F20, 64
bytes
04/19/07 03:47:20.456 PM: Save tracking, rc=0,
filename=(null), offset=0, complete=1
04/19/07 03:47:20.456 PM: nfm open()
filename=/var/tps/nfm/tracking/933.Phase1.0.HERBIE,
oflag=0x102, mode=0x1B6
04/19/07 03:47:20.456 PM: nfm_open(),
file=/var/tps/nfm/tracking/933.Phase1.0.HERBIE,
fileid=12
04/19/07 03:47:20.456 PM: nfm_write(), fileid=12,
size=152, wsize=152
04/19/07 03:47:20.456 PM: nfm_close(), fileid=0xC
04/19/07 03:47:20.457 PM: Child 515 exits(0)
04/19/07 03:47:20.457 PM: thread cleanup id=203  rc=0,
childc=0x3004F738
04/19/07 03:47:20.458 PM: child pipe write rc=8
04/19/07 03:47:20.458 PM: Closing log handle
0xF0004D20, 11
```

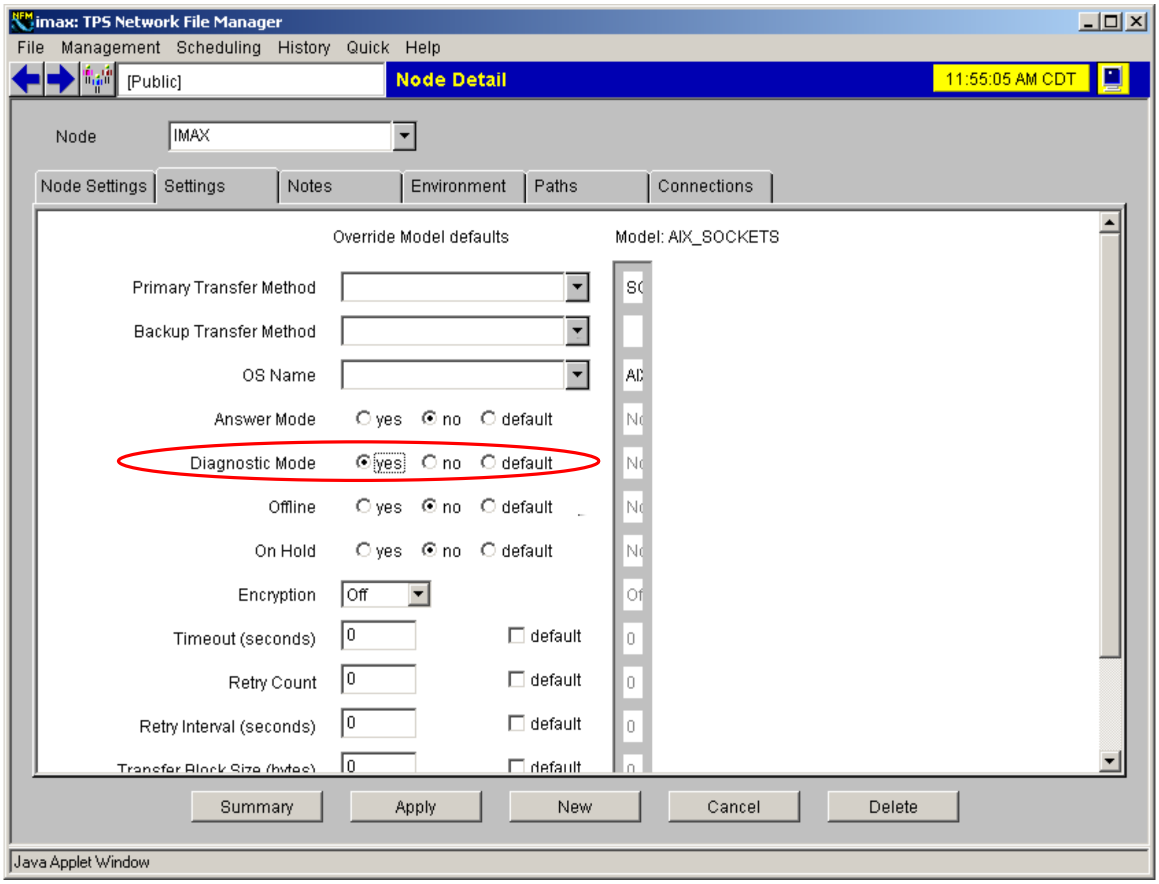
NFM Client Logs

Why turn on NFM Client Logs?

If you are troubleshooting a problem with a file transfer or an execute, an NFM Client log can help.

How to turn on NFM Client Logs?

NFM Client logging is turned on from the GUI in the Node or Model screen. Make sure to turn on logging for each node that is currently having a problem. This should include the target and the source nodes or node group.



[Public] Node Detail 11:55:05 AM CDT

Node: IMAX

Node Settings | Settings | Notes | Environment | Paths | Connections

	Override Model defaults		Model: AIX_SOCKETS
Primary Transfer Method			SC
Backup Transfer Method			
OS Name			AI
Answer Mode	○ yes ● no ○ default		No
Diagnostic Mode	● yes ○ no ○ default		No
Offline	○ yes ● no ○ default		No
On Hold	○ yes ● no ○ default		No
Encryption	Off		Of
Timeout (seconds)	0	□ default	0
Retry Count	0	□ default	0
Retry Interval (seconds)	0	□ default	0
Transfer Block Size (bytes)	0	□ default	0

(Turning on *NFM Client log*)

Where are NFM Client Logs kept? What NFM Client Logs are created?

NFM Client Logs are kept in the following directory:

Platform	**Directory**
UNIX	`/var/tps/nfmc/logs`
OS/390	`kept in the JES Job log`
Windows	`C:/Program Files/TPS Systems/NFMClient/logs`

NFM Client Log files created (Windows & UNIX):

`<plan name>.<plan id>.<number>`	The plan name is truncated to 8 characters to accommodate 8.3 file systems.

NFM Client Log files created (OS/390):

These entries are stored in the JES Job Log with the naming convention:

```
TPS01.NFMC.LOGS.A<4 digit job #><3 digit function #>
```

What does the NFM Client Log look like? What does it mean?

Now let’s examine the different NFM Client Log.

Log file: ***<plan name>.<plan id>.<number>***
Description: Log of the NFM Client

<table>
<tr><th>Description</th><th>Log file</th></tr>
<tr><td>Binary Header Info</td><td>Thread 0x00000A4C fd 3
06/04/07 04:01:30 PM: NFM processing request, version 2.2.2.0
06/04/07 04:01:30 PM: In msgtype 1
06/04/07 04:01:30 PM: Incoming (raw):
06/04/07 04:01:30 PM: 0000:   4E46 4D43 03F2 0000 011A 0018 03F4 0300 |NFMC............|
06/04/07 04:01:30 PM: 0010:   0001 0000 0000 0000 0001 003D 0000 007B |...........=...{|
06/04/07 04:01:30 PM: 0020:   000A 2821 4854 4D4C 2E30 3030 3037 2E30 |..(!HTML.00007.0|
06/04/07 04:01:30 PM: 0030:   3030 0000 0000 0000 0000 0000 0000 0000 |00..............|
06/04/07 04:01:30 PM: 0040:   0000 0000 00A1 C03C D0B8 B23D AAF6 9CFC |.......<...=....|
06/04/07 04:01:30 PM: 0050:   6F6A 8134 F772 6564 6861 7400 433A 5C74 |oj.4.redhat.C:\t|
06/04/07 04:01:30 PM: 0060:   656D 7064 6F65 7836 3638 3700 3931 3100 |empdoex6687.911.|
06/04/07 04:01:30 PM: 0070:   2020 2020 2020 2020 2030 0033 3230 3030 | 0.32000|
06/04/07 04:01:30 PM: 0080:   0000 3000 3000 3332 3030 3000 3830 3038 |..0.0.32000.8008|
06/04/07 04:01:30 PM: 0090:   0000 0000 0300 0C00 0000 8700 0103 0000 |................|
06/04/07 04:01:30 PM: 00a0:   4E46 4D43 03F2 0000 007A 0018 03F4 0300 |NFMC.....z......|
06/04/07 04:01:30 PM: 00b0:   0001 0000 0000 0000 0001 003D 0000 0062 |...........=...b|
06/04/07 04:01:30 PM: 00c0:   000B 0021 4854 4D4C 2E30 3030 3037 2E30 |...!HTML.00007.0|
06/04/07 04:01:30 PM: 00d0:   3030 0000 0000 0000 0000 0000 0000 0000 |00..............|
06/04/07 04:01:30 PM: 00e0:   0000 0000 00A1 C03C D0B8 B23D AAF6 9CFC |.......<...=....|
06/04/07 04:01:30 PM: 00f0:   6F6A 8134 F72F 746D 702F 646F 6578 3636 |oj.4./tmp/doex66|
06/04/07 04:01:30 PM: 0100:   3837 0033 3230 3030 0020 2020 2020 2020 |87.32000. |
06/04/07 04:01:30 PM: 0110:   2020 3000 3000 3000 3000 | 0.0.0.0. |</td></tr>
</table>

```
                             06/04/07 04:01:30 PM: Incoming (formatted):
    ==================       06/04/07 04:01:30 PM: NFM message (formatted)
      NFM Header Info        06/04/07 04:01:30 PM: NFM message (formatted)
                             06/04/07 04:01:30 PM:  HDR
    ==================       06/04/07 04:01:30 PM:   signature    : NFMC
                             06/04/07 04:01:30 PM:   old version  : 1010
                             06/04/07 04:01:30 PM:   version      : 1012
                             06/04/07 04:01:30 PM:   msglen       : 282
       Message Type          06/04/07 04:01:30 PM:   hdrlen       : 24
                             06/04/07 04:01:30 PM:   msgtype      : Request
                             06/04/07 04:01:30 PM:   abilities1   : 0x3 PACING SSL
                             06/04/07 04:01:30 PM:   abilities2   : 0x0
                             06/04/07 04:01:30 PM:   mblknum      : 0
                             06/04/07 04:01:30 PM:   acknum       : 0
                             06/04/07 04:01:30 PM:   flags1       : 0x0
  Message Type (Request)     06/04/07 04:01:30 PM:   flags2       : 0x0
                             06/04/07 04:01:30 PM:  REQUEST
                             06/04/07 04:01:30 PM:   hdrlen   : 61
  Request Type (Transfer     06/04/07 04:01:30 PM:   complen  : 123
         Receive)            06/04/07 04:01:30 PM:   reqtype  : Transfer Retrieve
                             06/04/07 04:01:30 PM:   rflags1  : 0x28 TRACEON
                             TEMPFILES
                             06/04/07 04:01:30 PM:   rflags2  : 0x21 PRIORITY_ON
                             06/04/07 04:01:30 PM:   priority : 0x1
                             06/04/07 04:01:30 PM:   rflags3  : 0x0
                             06/04/07 04:01:30 PM:   logprefix: HTML.00007.000
                             06/04/07 04:01:30 PM:   arg0     : redhat
                             06/04/07 04:01:30 PM:   arg1     : C:\tempdoex6687
        Block size           06/04/07 04:01:30 PM:   arg2     : 911
                             06/04/07 04:01:30 PM:   arg3     :         0
                             06/04/07 04:01:30 PM:   arg4     : 32000
                             06/04/07 04:01:30 PM:   arg5     :
                             06/04/07 04:01:30 PM:   arg6     : 0
      Listening port         06/04/07 04:01:30 PM:   arg7     : 0
                             06/04/07 04:01:30 PM:   arg8     : 32000
                             06/04/07 04:01:30 PM:   arg9     : 8008
                             06/04/07 04:01:30 PM:   arg10     :
                             06/04/07 04:01:30 PM:   arg11     :
                             06/04/07 04:01:30 PM:  ATTACHMENT
                             06/04/07 04:01:30 PM:   hdrlen     : 12
                             06/04/07 04:01:30 PM:   complen    : 135
                             06/04/07 04:01:30 PM:   type      : 3
                             06/04/07 04:01:30 PM:   filename :
                             06/04/07 04:01:30 PM:   size      : 122
                             06/04/07 04:01:30 PM:   flags1    : 0x0
                             06/04/07 04:01:30 PM: ----- Attachment Begins -----
                             06/04/07 04:01:30 PM: NFMC•ò
                             06/04/07 04:01:30 PM: ----- Attachment Ends -------
                             06/04/07 04:01:30 PM: nfmctransr()
                             06/04/07 04:01:30 PM: nfmctransr() vroot=""
                             06/04/07 04:01:30 PM:              trgf
                             ="C:\tempdoex6687"
                             06/04/07 04:01:30 PM:              tmpf
                             ="C:\tempdoex6687"
                             06/04/07 04:01:30 PM:    rflags1=0x28, rflags2=0x21,
                             rcomname=redhat, trgfile=C:\tempdoex6687, trc-
                             >trgfsize=911,
                             06/04/07 04:01:30 PM:    foffset=0, statusi=32000,
                             ackcount=0, trc->confirmi=32000, ressockh=0
                             06/04/07 04:01:30 PM: nfm_fileexist on C:\tempdoex6687
                             returns 0.
                             06/04/07 04:01:30 PM: nfmctransr() reserved_sock = ""
                             06/04/07 04:01:30 PM: opensock(). comname=redhat,
                             port=8008, encrypt=0, priority=1, timeout=0,
```

	keepalive=0
	06/04/07 04:01:30 PM: SockOpen(). spec=redhat,
	service=8008
	06/04/07 04:01:30 PM: SockOpen(). spec was not x.x.x.x
	format.
	06/04/07 04:01:30 PM: SockOpen(). gethostbyname():
	redhat found. Addr=13f054
	06/04/07 04:01:30 PM: SockOpen(). calling
	socket(AF_INET,SOCK_STREAM,0)
	06/04/07 04:01:30 PM: SockOpen(). sock_fd=364
	06/04/07 04:01:30 PM: SockOpen(). returning OK
	06/04/07 04:01:30 PM: pmregister() local IP
	addr=204.62.234.34
	06/04/07 04:01:30 PM: pm_init(204.62.234.34)
	06/04/07 04:01:30 PM: pm_getsem(nfmpm_204.62.234.34,
	1, 1)
	06/04/07 04:01:30 PM: Open/Create Semaphore on
	nfmpm_204.62.234.34.1 gets 368, returns 0
	06/04/07 04:01:30 PM: pm_getsem(nfmpm_204.62.234.34,
	2, 1)
=================	06/04/07 04:01:30 PM: Open/Create Semaphore on
NFM Header Info	nfmpm_204.62.234.34.2 gets 376, returns 0
=================	06/04/07 04:01:30 PM: pm_getsem(nfmpm_204.62.234.34,
	3, 1)
	06/04/07 04:01:30 PM: Open/Create Semaphore on
	nfmpm_204.62.234.34.3 gets 372, returns 0
	06/04/07 04:01:30 PM: pm_getsem(nfmpm_204.62.234.34,
	4, 1)
Message Type	06/04/07 04:01:30 PM: Open/Create Semaphore on
	nfmpm_204.62.234.34.4 gets 380, returns 0
	06/04/07 04:01:30 PM: pm_getsem(nfmpm_204.62.234.34,
	5, 10000)
	06/04/07 04:01:30 PM: Open/Create Semaphore on
	nfmpm_204.62.234.34.5 gets 0, returns 0
	06/04/07 04:01:30 PM: Outgoing (formatted):
Message Type (Request)	06/04/07 04:01:30 PM: NFM message (formatted)
	06/04/07 04:01:30 PM: NFM message (formatted)
	06/04/07 04:01:30 PM: HDR
Request Type (Transfer	06/04/07 04:01:30 PM: signature : NFMC
Send)	06/04/07 04:01:30 PM: old version : 1010
	06/04/07 04:01:30 PM: version : 1012
	06/04/07 04:01:30 PM: msglen : 122
	06/04/07 04:01:30 PM: hdrlen : 24
	06/04/07 04:01:30 PM: msgtype : Request
	06/04/07 04:01:30 PM: abilities1 : 0x3 PACING SSL
	06/04/07 04:01:30 PM: abilities2 : 0x0
	06/04/07 04:01:30 PM: mblknum : 0
	06/04/07 04:01:30 PM: acknum : 0
	06/04/07 04:01:30 PM: flags1 : 0x0
	06/04/07 04:01:30 PM: flags2 : 0x0
	06/04/07 04:01:30 PM: REQUEST
Binary Message data	06/04/07 04:01:30 PM: hdrlen : 61
	06/04/07 04:01:30 PM: complen : 98
	06/04/07 04:01:30 PM: reqtype : Transfer Send
	06/04/07 04:01:30 PM: rflags1 : 0x0
	06/04/07 04:01:30 PM: rflags2 : 0x21 PRIORITY_ON
	06/04/07 04:01:30 PM: priority : 0x1
	06/04/07 04:01:30 PM: rflags3 : 0x0
	06/04/07 04:01:30 PM: logprefix: HTML.00007.000
	06/04/07 04:01:30 PM: arg0 : /tmp/doex6687
	06/04/07 04:01:30 PM: arg1 : 32000
	06/04/07 04:01:30 PM: arg2 : 0
	06/04/07 04:01:30 PM: arg3 : 0
	06/04/07 04:01:30 PM: arg4 : 0

```
06/04/07 04:01:30 PM:   arg5     : 0
06/04/07 04:01:30 PM: Outgoing (raw):
06/04/07 04:01:30 PM: 0000:   4E46 4D43 03F2 0000 007A
0018 03F4 0300   |NFMC.....z......|
06/04/07 04:01:30 PM: 0010:   0001 0000 0000 0000 0001
003D 0000 0062   |...........=...b|
06/04/07 04:01:30 PM: 0020:   000B 0021 4854 4D4C 2E30
3030 3037 2E30   |...!HTML.00007.0|
06/04/07 04:01:30 PM: 0030:   3030 0000 0000 0000 0000
0000 0000 0000   |00..............|
06/04/07 04:01:30 PM: 0040:   0000 0000 00A1 C03C D0B8
B23D AAF6 9CFC   |.......<...=....|
06/04/07 04:01:30 PM: 0050:   6F6A 8134 F72F 746D 702F
646F 6578 3636   |oj.4./tmp/doex66|
06/04/07 04:01:30 PM: 0060:   3837 0033 3230 3030 0020
2020 2020 2020   |87.32000.       |
06/04/07 04:01:30 PM: 0070:   2020 3000 3000 3000 3000
|  0.0.0.0.      |
06/04/07 04:01:30 PM: Priority enque level 1
06/04/07 04:01:30 PM: pm_enque(), sockhndl=364,
user=6354F8, priority=1
06/04/07 04:01:30 PM: pm_locksem(376,1)
06/04/07 04:01:30 PM: pm_locksem() windows, rc=0
06/04/07 04:01:31 PM: pm_wouldlocksem(0, 10000)
06/04/07 04:01:31 PM: pm_locksem() windows rc=6539232,
rc=0
06/04/07 04:01:31 PM: pm_wouldlocksem(380, 1)
06/04/07 04:01:31 PM: pm_locksem() windows rc=0, rc=0
06/04/07 04:01:31 PM: pm_wouldlocksem(372, 1)
06/04/07 04:01:31 PM: pm_locksem() windows rc=0, rc=0
06/04/07 04:01:31 PM: Priority deque level 1
06/04/07 04:01:31 PM: pm_deque()
06/04/07 04:01:31 PM: pm_unlocksem(376,1)
06/04/07 04:01:31 PM: pm_unlocksem() rc=0
06/04/07 04:01:31 PM: SockSnd(122 bytes).
06/04/07 04:01:31 PM: Alloc 6357A8, 84 bytes
06/04/07 04:01:31 PM: Alloc F80060, 32264 bytes
06/04/07 04:01:31 PM: Alloc 63C8C8, 532 bytes
06/04/07 04:01:31 PM:  transr loop
06/04/07 04:01:31 PM:    Memory allocated =
286,32880,0,0,0
06/04/07 04:01:31 PM:    Memory maximum   =
286,32880,0,0,0
06/04/07 04:01:31 PM: Entering SockRcv(). buflen=18,
red=0, sock=364
06/04/07 04:01:31 PM: nfm_select(rd=364, wr=-1, ex=-1,
to=-1)
06/04/07 04:01:31 PM: nfm_select() before select
06/04/07 04:01:31 PM: nfm_select() after select
06/04/07 04:01:31 PM: returns (0)
06/04/07 04:01:31 PM: recv returned 18 bytes
06/04/07 04:01:31 PM: Entering SockRcv(). buflen=998,
red=0, sock=364
06/04/07 04:01:31 PM: nfm_select(rd=364, wr=-1, ex=-1,
to=-1)
06/04/07 04:01:31 PM: nfm_select() before select
06/04/07 04:01:31 PM: nfm_select() after select
06/04/07 04:01:31 PM: returns (0)
06/04/07 04:01:31 PM: recv returned 998 bytes
06/04/07 04:01:31 PM: In msgtype 3
06/04/07 04:01:31 PM: Incoming (raw):
06/04/07 04:01:31 PM: 0000:   4E46 4D43 03F2 0000 03F8
0018 03F4 0300   |NFMC............|
06/04/07 04:01:31 PM: 0010:   0003 0000 0000 0000 0004
```

```
                       0028 0000 03E0    |...........(....|
                       06/04/07 04:01:31 PM: 0020:   0000 0000 0000 0000 0100
                       0000 0000 0000    |................|
                       06/04/07 04:01:31 PM: 0030:   0000 0000 0000 0000 0000
                       0000 0000 0000    |................|
                       06/04/07 04:01:31 PM: 0040:   0029 0000 0000 01B0 6E65
                       7773 0000 0000    |.)......news....|
                       06/04/07 04:01:31 PM: 0050:   006E 6577 7300 0000 0000
                       0000 6C02 0000    |.news.......l...|
                       06/04/07 04:01:31 PM: 0060:   0000 0000 0000 0000 0053
 =================     6572 7665 7220    |.........Server |
  NFM Header Info      06/04/07 04:01:31 PM: 0070:   7374 6174 7573 3A0A 5365
 =================     7276 6572 2074    |status:.Server t|
                       06/04/07 04:01:31 PM: 0080:   6872 6F74 746C 6564 204E
                       6F20 7370 6163    |hrottled No spac|
                       06/04/07 04:01:31 PM: 0090:   6520 2873 706F 6F6C 2920
                       5B69 6E6E 7761    |e (spool) [innwa|
    Message Type       06/04/07 04:01:31 PM: 00a0:   7463 683A 3332 5D20 3020
                       6C74 2038 3030    |tch:32] 0 lt 800|
                       06/04/07 04:01:31 PM: 00b0:   300A 416C 6C6F 7769 6E67
                       2072 656D 6F74    |0.Allowing remot|
                       06/04/07 04:01:31 PM: 00c0:   6520 636F 6E6E 6563 7469
                       6F6E 730A 5061    |e connections.Pa|
                       06/04/07 04:01:31 PM: 00d0:   7261 6D65 7465 7273 2063
Message Type (Data)    2031 3420 6920    |rameters c 14 i |
                       06/04/07 04:01:31 PM: 00e0:   3020 2830 2920 6C20 3735
                       3030 3020 6F20    |0 (0) l 75000 o |
                       06/04/07 04:01:31 PM: 00f0:   3234 3320 7420 3330 3020
                       4820 3220 5420    |243 t 300 H 2 T |
                       06/04/07 04:01:31 PM: ... additional 760 bytes not
                       displayed
                       06/04/07 04:01:31 PM: Incoming (formatted):
                       06/04/07 04:01:31 PM: NFM message (formatted)
                       06/04/07 04:01:31 PM: NFM message (formatted)
                       06/04/07 04:01:31 PM:  HDR
                       06/04/07 04:01:31 PM:   signature    : NFMC
                       06/04/07 04:01:31 PM:   old version  : 1010
                       06/04/07 04:01:31 PM:   version      : 1012
                       06/04/07 04:01:31 PM:   msglen       : 1016
                       06/04/07 04:01:31 PM:   hdrlen       : 24
                       06/04/07 04:01:31 PM:   msgtype      : Data
                       06/04/07 04:01:31 PM:   abilities1   : 0x3 PACING SSL
                       06/04/07 04:01:31 PM:   abilities2   : 0x0
                       06/04/07 04:01:31 PM:   mblknum      : 0
                       06/04/07 04:01:31 PM:   acknum       : 0
                       06/04/07 04:01:31 PM:   flags1       : 0x0
                       06/04/07 04:01:31 PM:   flags2       : 0x0
                       06/04/07 04:01:31 PM:  DATA
                       06/04/07 04:01:31 PM:   hdrlen             : 40
                       06/04/07 04:01:31 PM:   complen            : 992
                       06/04/07 04:01:31 PM:   blocknum           : 0
                       06/04/07 04:01:31 PM:   file offset (low)  : 0
                       06/04/07 04:01:31 PM:   file offset (high) : 0
                       06/04/07 04:01:31 PM:   flags1    : 0x1 FIRSTBLOCK
                       06/04/07 04:01:31 PM:   mode      : 0x1B0
                       06/04/07 04:01:31 PM:   user      : news
                       06/04/07 04:01:31 PM:   group     : news
                       06/04/07 04:01:31 PM:   fda       : 0
                       06/04/07 04:01:31 PM:   dos attrs : 0x0
                       06/04/07 04:01:31 PM:   win attrs : 0x0
Binary Message Data    06/04/07 04:01:31 PM:   timestamp : 0x0
                       06/04/07 04:01:31 PM: nfmctransr() data block
                       received.
                       06/04/07 04:01:31 PM: Creating file C:\tempdoex6687
```

	for 911 bytes, offset 0
	06/04/07 04:01:31 PM: nfmwin_open()
	name=C:\tempdoex6687, oflag=0x8301, mode=0x180
	06/04/07 04:01:31 PM: nfm_open(),
	file=C:\tempdoex6687, rc=0, fileid=384
	06/04/07 04:01:31 PM: nfm_write(), fileid=384,
	size=911, wsize=911
	06/04/07 04:01:31 PM: transr loop
	06/04/07 04:01:31 PM: Memory allocated =
	286,32880,0,0,0
	06/04/07 04:01:31 PM: Memory maximum =
	286,32880,0,0,0
	06/04/07 04:01:31 PM: Entering SockRcv(). buflen=18,
	red=0, sock=364
===================	06/04/07 04:01:31 PM: nfm_select(rd=364, wr=-1, ex=-1,
NFM Header Info	to=-1)
===================	06/04/07 04:01:31 PM: nfm_select() before select
	06/04/07 04:01:31 PM: nfm_select() after select
	06/04/07 04:01:31 PM: returns (0)
	06/04/07 04:01:31 PM: recv returned 18 bytes
	06/04/07 04:01:31 PM: Entering SockRcv(). buflen=104,
	red=0, sock=364
Message Type	06/04/07 04:01:31 PM: nfm_select(rd=364, wr=-1, ex=-1,
	to=-1)
	06/04/07 04:01:31 PM: nfm_select() before select
	06/04/07 04:01:31 PM: nfm_select() after select
	06/04/07 04:01:31 PM: returns (0)
	06/04/07 04:01:31 PM: recv returned 104 bytes
	06/04/07 04:01:31 PM: In msgtype 2
Message Type (Reply)	06/04/07 04:01:31 PM: Incoming (raw):
	06/04/07 04:01:31 PM: 0000: 4E46 4D43 03F2 0000 007A
	0018 03F4 0300 |NFMC.....z......|
Reply Type (Transfer	06/04/07 04:01:31 PM: 0010: 0002 0000 0000 0000 0001
Send)	003D 0000 0062 |...........=...b|
	06/04/07 04:01:31 PM: 0020: 000B 0021 4854 4D4C 2E30
	3030 3037 2E30 |...!HTML.00007.0|
	06/04/07 04:01:31 PM: 0030: 3030 0000 0000 0000 0000
	0000 0000 0000 |00..............|
	06/04/07 04:01:31 PM: 0040: 0000 0000 00A1 C03C D0B8
	B23D AAF6 9CFC |.......<...=....|
	06/04/07 04:01:31 PM: 0050: 6F6A 8134 F72F 746D 702F
	646F 6578 3636 |oj.4./tmp/doex66|
	06/04/07 04:01:31 PM: 0060: 3837 0033 3230 3030 0020
	2020 2020 2020 |87.32000. |
	06/04/07 04:01:31 PM: 0070: 2020 3000 3000 3000 3000
	| 0.0.0.0. |
	06/04/07 04:01:31 PM: Incoming (formatted):
	06/04/07 04:01:31 PM: NFM message (formatted)
	06/04/07 04:01:31 PM: NFM message (formatted)
	06/04/07 04:01:31 PM: HDR
	06/04/07 04:01:31 PM: signature : NFMC
	06/04/07 04:01:31 PM: old version : 1010
	06/04/07 04:01:31 PM: version : 1012
	06/04/07 04:01:31 PM: msglen : 122
	06/04/07 04:01:31 PM: hdrlen : 24
	06/04/07 04:01:31 PM: msgtype : Reply
	06/04/07 04:01:31 PM: abilities1 : 0x3 PACING SSL
=====================	06/04/07 04:01:31 PM: abilities2 : 0x0
NFM Header Info	06/04/07 04:01:31 PM: mblknum : 0
=====================	06/04/07 04:01:31 PM: acknum : 0
	06/04/07 04:01:31 PM: flags1 : 0x0
	06/04/07 04:01:31 PM: flags2 : 0x0
	06/04/07 04:01:31 PM: REPLY
	06/04/07 04:01:31 PM: hdrlen : 61

Message Type	06/04/07 04:01:31 PM: complen : 98
	06/04/07 04:01:31 PM: reqtype : Transfer Send
	06/04/07 04:01:31 PM: rflags1 : 0x0
	06/04/07 04:01:31 PM: rflags2 : 0x21 PRIORITY_ON
	06/04/07 04:01:31 PM: priority : 0x1
	06/04/07 04:01:31 PM: rflags3 : 0x0
	06/04/07 04:01:31 PM: logprefix: HTML.00007.000
Message Type (Status)	06/04/07 04:01:31 PM: arg0 : /tmp/doex6687
	06/04/07 04:01:31 PM: arg1 : 32000
	06/04/07 04:01:31 PM: arg2 : 0
	06/04/07 04:01:31 PM: arg3 : 0
	06/04/07 04:01:31 PM: arg4 : 0
	06/04/07 04:01:31 PM: arg5 : 0
	06/04/07 04:01:31 PM: nfmctransr() reply received.
Binary Message Data	06/04/07 04:01:31 PM: nfm_close(), fileid=0x180, rc=0
	06/04/07 04:01:31 PM: close returns 0,
	06/04/07 04:01:31 PM: setfattrs() set=38, modtime=0
	06/04/07 04:01:31 PM: nfmwin_open()
	name=C:\tempdoex6687:NFM_attributes, oflag=0x8302,
	mode=0x180
	06/04/07 04:01:31 PM: nfm_open(),
	file=C:\tempdoex6687:NFM_attributes, rc=0, fileid=384
	06/04/07 04:01:31 PM: nfm_write(), fileid=384,
	size=30, wsize=30
	06/04/07 04:01:31 PM: nfm_close(), fileid=0x180, rc=0
	06/04/07 04:01:31 PM: Outgoing (formatted):
	06/04/07 04:01:31 PM: NFM message (formatted)
	06/04/07 04:01:31 PM: NFM message (formatted)
	06/04/07 04:01:31 PM: HDR
	06/04/07 04:01:31 PM: signature : NFMC
	06/04/07 04:01:31 PM: old version : 1010
	06/04/07 04:01:31 PM: version : 1012
	06/04/07 04:01:31 PM: msglen : 84
	06/04/07 04:01:31 PM: hdrlen : 24
===================	06/04/07 04:01:31 PM: msgtype : Status
NFM Header Info	06/04/07 04:01:31 PM: abilities1 : 0x3 PACING SSL
===================	06/04/07 04:01:31 PM: abilities2 : 0x0
	06/04/07 04:01:31 PM: mblknum : 0
	06/04/07 04:01:31 PM: acknum : 0
	06/04/07 04:01:31 PM: flags1 : 0x0
	06/04/07 04:01:31 PM: flags2 : 0x0
Message Type	06/04/07 04:01:31 PM: STATUS
	06/04/07 04:01:31 PM: hdrlen : 60
	06/04/07 04:01:31 PM: complen : 60
	06/04/07 04:01:31 PM: file offset (low) : 911
	06/04/07 04:01:31 PM: file offset (high) : 0
	06/04/07 04:01:31 PM: trans pid : 27650
	06/04/07 04:01:31 PM: flags1 : 0x1
Message Type (Reply)	STATUS
	06/04/07 04:01:31 PM: Outgoing (raw):
	06/04/07 04:01:31 PM: 0000: 4E46 4D43 03F2 0000 0054
Reply Type (Transfer	0018 03F4 0300 |NFMC.....T......|
Retrieve)	06/04/07 04:01:31 PM: 0010: 0004 0000 0000 0000 0005
	003C 0000 003C |...........<...<|
	06/04/07 04:01:31 PM: 0020: 0000 038F 0000 6C02 0100
	0000 0000 0000 |......l.........|
	06/04/07 04:01:31 PM: 0030: 0000 0000 0000 0000 0000
	0000 0000 0000 |................|
	06/04/07 04:01:31 PM: 0040: 0000 0B18 0000 0B0E 0000
	0001 0000 0000 |................|
	06/04/07 04:01:31 PM: 0050: 0000 0000
	|.... |
	06/04/07 04:01:31 PM: SockSnd(84 bytes).
	06/04/07 04:01:31 PM: okreply()

Binary Message Data

```
06/04/07 04:01:31 PM: Alloc 635820, 147 bytes
06/04/07 04:01:31 PM: nfmcreply()
06/04/07 04:01:31 PM: nfmcreplys()
06/04/07 04:01:31 PM: nfmcreply() port=8008
06/04/07 04:01:31 PM: Outgoing (formatted):
06/04/07 04:01:31 PM: NFM message (formatted)
06/04/07 04:01:31 PM: NFM message (formatted)
06/04/07 04:01:31 PM:  HDR
06/04/07 04:01:31 PM:   signature    : NFMC
06/04/07 04:01:31 PM:   old version  : 1010
06/04/07 04:01:31 PM:   version      : 1012
06/04/07 04:01:31 PM:   msglen       : 147
06/04/07 04:01:31 PM:   hdrlen       : 24
06/04/07 04:01:31 PM:   msgtype      : Reply
06/04/07 04:01:31 PM:   abilities1   : 0x3 PACING SSL
06/04/07 04:01:31 PM:   abilities2   : 0x0
06/04/07 04:01:31 PM:   mblknum      : 0
06/04/07 04:01:31 PM:   acknum       : 0
06/04/07 04:01:31 PM:   flags1       : 0x0
06/04/07 04:01:31 PM:   flags2       : 0x0
06/04/07 04:01:31 PM:  REPLY
06/04/07 04:01:31 PM:   hdrlen   : 61
06/04/07 04:01:31 PM:   complen  : 123
06/04/07 04:01:31 PM:   reqtype  : Transfer Retrieve
06/04/07 04:01:31 PM:   rflags1  : 0x28 TRACEON
TEMPFILES
06/04/07 04:01:31 PM:   rflags2  : 0x21 PRIORITY_ON
06/04/07 04:01:31 PM:   priority : 0x1
06/04/07 04:01:31 PM:   rflags3  : 0x0
06/04/07 04:01:31 PM:   logprefix: HTML.00007.000
06/04/07 04:01:31 PM:   arg0     : redhat
06/04/07 04:01:31 PM:   arg1     : C:\tempdoex6687
06/04/07 04:01:31 PM:   arg2     : 911
06/04/07 04:01:31 PM:   arg3     :          0
06/04/07 04:01:31 PM:   arg4     : 32000
06/04/07 04:01:31 PM:   arg5     :
06/04/07 04:01:31 PM:   arg6     : 0
06/04/07 04:01:31 PM:   arg7     : 0
06/04/07 04:01:31 PM:   arg8     : 32000
06/04/07 04:01:31 PM:   arg9     : 8008
06/04/07 04:01:31 PM:   arg10     :
06/04/07 04:01:31 PM:   arg11     :
06/04/07 04:01:31 PM: Outgoing (raw):
06/04/07 04:01:31 PM: 0000:   4E46 4D43 03F2 0000 0093
0018 03F4 0300   |NFMC............|
06/04/07 04:01:31 PM: 0010:   0002 0000 0000 0000 0001
003D 0000 007B   |...........=...{|
06/04/07 04:01:31 PM: 0020:   000A 2821 4854 4D4C 2E30
3030 3037 2E30   |..(!HTML.00007.0|
06/04/07 04:01:31 PM: 0030:   3030 0000 0000 0000 0000
0000 0000 0000   |00..............|
06/04/07 04:01:31 PM: 0040:   0000 0000 00A1 C03C D0B8
B23D AAF6 9CFC   |.......<...=....|
06/04/07 04:01:31 PM: 0050:   6F6A 8134 F772 6564 6861
7400 433A 5C74   |oj.4.redhat.C:\t|
06/04/07 04:01:31 PM: 0060:   656D 7064 6F65 7836 3638
3700 3931 3100   |empdoex6687.911.|
06/04/07 04:01:31 PM: 0070:   2020 2020 2020 2020 2030
0033 3230 3030   |         0.32000|
06/04/07 04:01:31 PM: 0080:   0000 3000 3000 3332 3030
3000 3830 3038   |..0.0.32000.8008|
06/04/07 04:01:31 PM: 0090:   0000 00
|...             |
06/04/07 04:01:31 PM: SockSnd(147 bytes).
```

```
06/04/07 04:01:31 PM: Alloc Free 635820, 147 bytes
06/04/07 04:01:31 PM: Alloc Free 63C8C8, 532 bytes
06/04/07 04:01:31 PM: Alloc Free F80060, 32264 bytes
06/04/07 04:01:31 PM: Alloc Free 6357A8, 84 bytes
06/04/07 04:01:31 PM: Alloc Free 638960, 286 bytes
06/04/07 04:01:31 PM: Entering SockRcv(). buflen=18,
red=0, sock=312
06/04/07 04:01:31 PM: nfm_select(rd=312, wr=-1, ex=-1,
to=100000)
06/04/07 04:01:31 PM: nfm_select() before select
06/04/07 04:01:31 PM: nfm_select() after select
06/04/07 04:01:31 PM: returns (0)
06/04/07 04:01:31 PM: closesock(). handle=312
06/04/07 04:01:31 PM: closesock().
setsockopt(SO_LINGER) fd=312, timeout=5, returned
0,6518360
06/04/07 04:01:31 PM: SockClose(312).
06/04/07 04:01:31 PM: SockClose(). calling
closesocket(312)
06/04/07 04:01:31 PM: SockClose(). returning OK
06/04/07 04:01:31 PM: pm wait count for level 0, 0
06/04/07 04:01:31 PM: pm wait count for level 1, 0
06/04/07 04:01:31 PM: pm wait count for level 2, 0
06/04/07 04:01:31 PM: pm wait count for level 3, 0
06/04/07 04:01:31 PM: pm wait count for level 4, 0
06/04/07 04:01:31 PM: closesock(). handle=364
06/04/07 04:01:31 PM: closesock().
setsockopt(SO_LINGER) fd=364, timeout=5, returned
0,6518360
06/04/07 04:01:31 PM: SockClose(364).
06/04/07 04:01:31 PM: SockClose(). calling
closesocket(364)
06/04/07 04:01:31 PM: SockClose(). returning OK
06/04/07 04:01:31 PM: pm wait count for level 0, 0
06/04/07 04:01:31 PM: pm wait count for level 1, 0
06/04/07 04:01:31 PM: pm wait count for level 2, 0
06/04/07 04:01:31 PM: pm wait count for level 3, 0
06/04/07 04:01:31 PM: pm wait count for level 4, 0
06/04/07 04:01:31 PM: pm_unregister()
06/04/07 04:01:31 PM:  leaving sockmsgsi
06/04/07 04:01:31 PM:    Memory allocated = 0,0,0,0,0
06/04/07 04:01:31 PM:    Memory maximum   =
286,33027,0,0,0
06/04/07 04:01:31 PM: Closing log handle
0x7803A6F0, 3
```

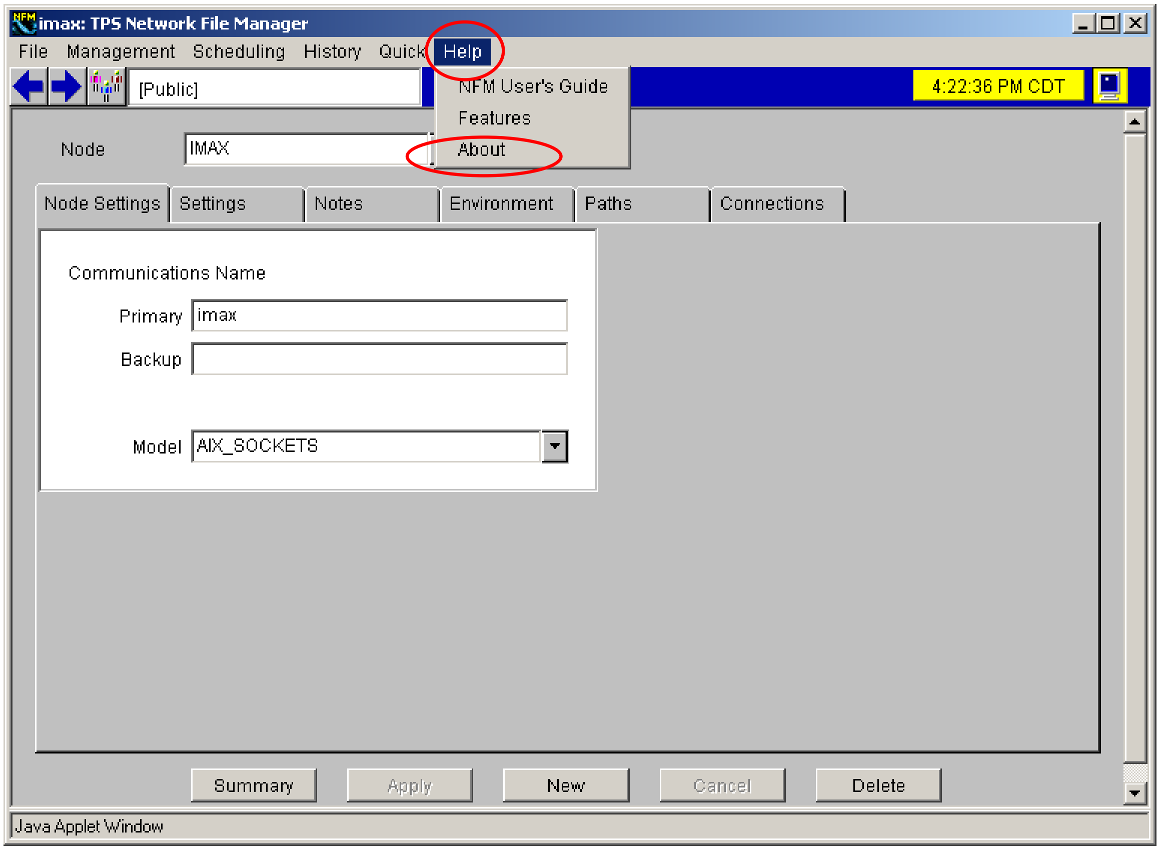
NFM GUI Interface Trace

Why turn on NFM GUI Trace?

The NFM GUI Trace shows all data from the NFM GUI to the NFM Server and vice versa. This is helpful if you want to see exactly what is being and sent and received from the NFM Server.

How to turn on NFM GUI Trace?

Step 1: Log into the GUI Interface
Step 2: Select `Help | About` (do this 4 times)



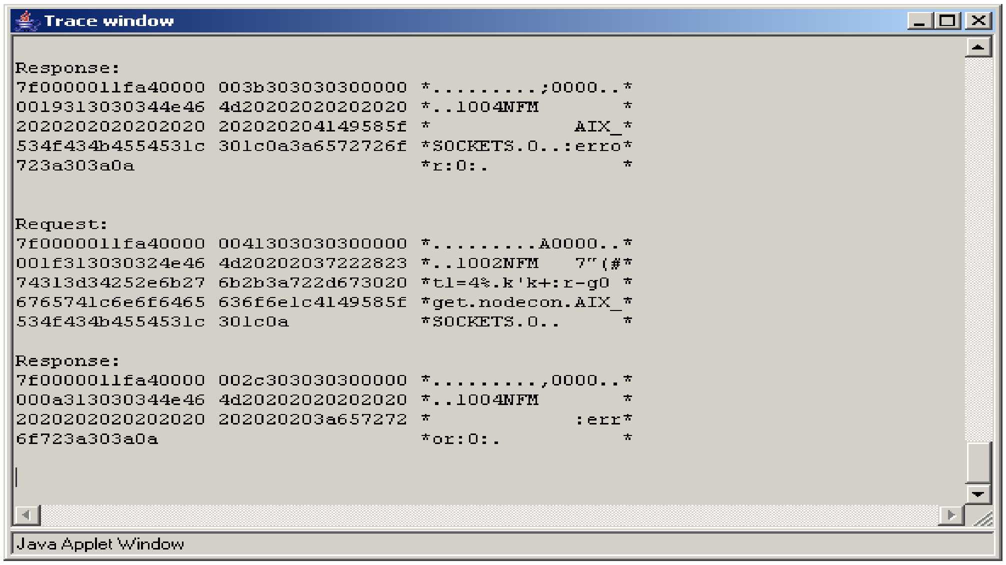
Node IMAX

Communications Name

Primary imax

Backup

Model AIX_SOCKETS



```
Response:
7f000001lfa40000 003b303030300000 *.........;0000..*
0019313030344e46 4d20202020202020 *..1004NFM       *
2020202020202020 202020204149585f *            AIX_*
534f434b4554531c 301c0a3a6572726f *SOCKETS.0..:erro*
723a303a0a                        *r:0:.           *


Request:
7f000001lfa40000 0041303030300000 *.........A0000..*
001f313030324e46 4d20202037222823 *..1002NFM   7"(#*
74313d34252e6b27 6b2b3a722d673020 *tl=4%.k'k+:r-g0 *
676574lc6e6f6465 636f6elc4149585f *get.nodecon.AIX_*
534f434b4554531c 301c0a           *SOCKETS.0..     *

Response:
7f000001lfa40000 002c303030300000 *.........,0000..*
000a313030344e46 4d20202020202020 *..1004NFM       *
2020202020202020 202020203a657272 *            :err*
6f723a303a0a                      *or:0:.          *
```

(*GUI trace window*)

The NFM GUI Trace is not saved to a log file and should be used for your benefit only. Use the 'TPS® Command Service Logs' instructions below if you need to send the data between the NFM GUI and NFM Server to TPS® Technical Support.

TPS® Command Service Logs

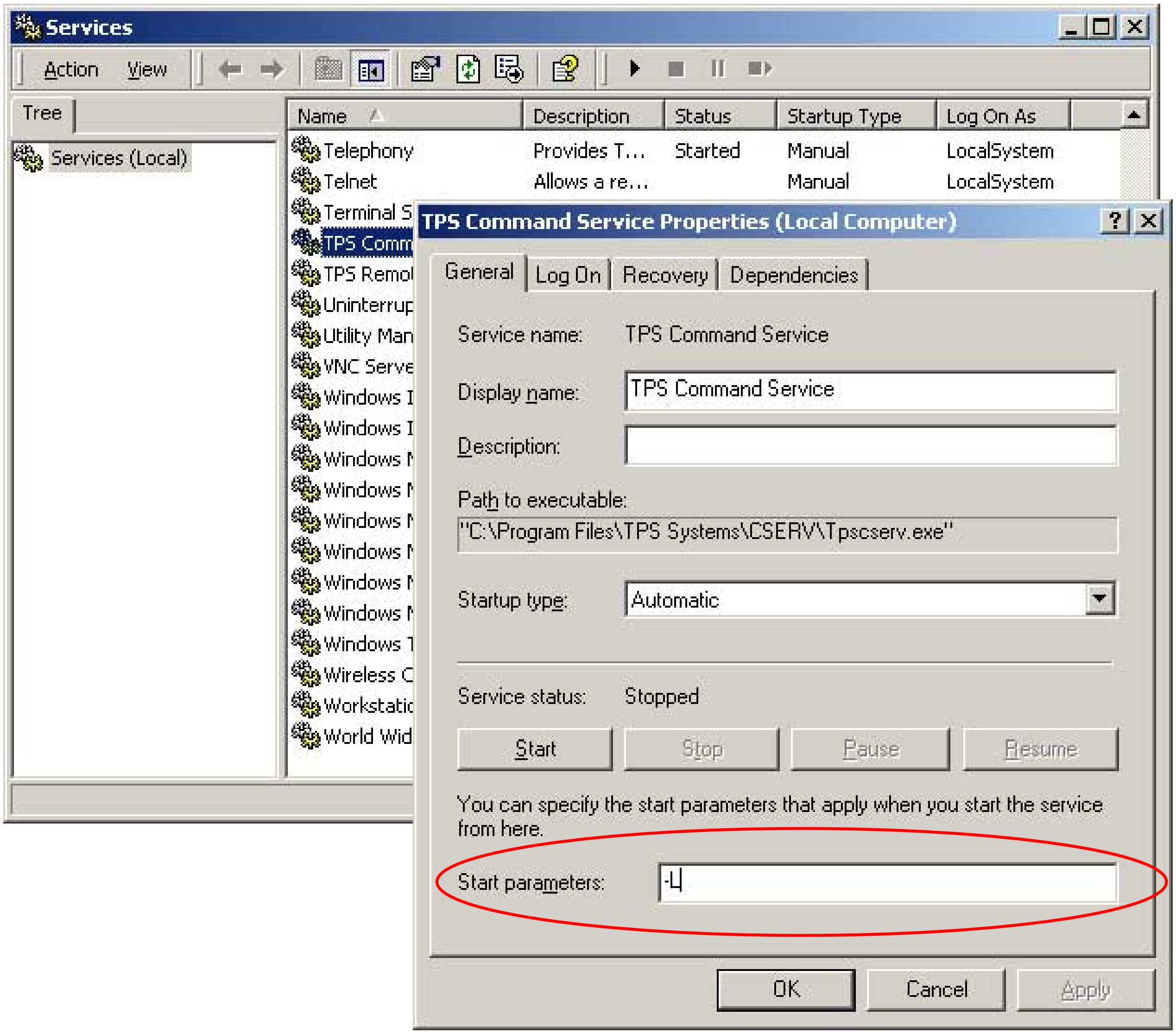
Why turn on TPS® Command Service Logs?

TPS® Command Service Logs log exactly what is being and sent and received from the NFM Server (similar to NFM GUI Trace). This is helpful when you want to see what is being sent or received from the GUI or through the NFM Local Command Line feature. TPS® Command Service provides the ability to capture communication in a log file, unlike the NFM GUI Trace (listed above).

How to turn on TPS® Command Service Logs?

For Windows:

Step 1: Open the `Services` control panel
Step 2: Double-click on the`TPS Command Service` line
Step 3: `Stop` the Service
Step 3: Under the `General` tab, in the `Startup parameters` field, add `-L`
Step 4: `Start` the Service



Name	Description	Status	Startup Type	Log On As
Telephony	Provides T...	Started	Manual	LocalSystem
Telnet	Allows a re...		Manual	LocalSystem
Terminal S				
TPS Comm				
TPS Remot				
Uninterrup				
Utility Man				
VNC Serve				
Windows I				
Windows I				
Windows M				
Windows M				
Windows M				
Windows M				
Windows M				
Windows M				
Windows T				
Wireless C				
Workstatic				
World Wid				

TPS Command Service Properties (Local Computer)

Service name: TPS Command Service

Display name: TPS Command Service

Description:

Path to executable:
"C:\Program Files\TPS Systems\CSERV\Tpscserv.exe"

Startup type: Automatic

Service status: Stopped

You can specify the start parameters that apply when you start the service from here.

Start parameters: -L

For UNIX Platforms:

Step 1: Stop the tpscserv process using the ‘`kill`’ command. (Note: `tpscserv` respawns itself from the `initab`. You might have to remove the respawn option from the `initab` before turning on logging.)
Step 2: Start TPS® Command Service with `tpscserv L –q`.

Where are TPS® Command Service Logs kept? What TPS® Command Service Logs are created?

TPS® Command Service Logs are keep in the following log directory:

Platform	Directory
UNIX	`/var/tps/cserv/logs`
Windows	`C:/Program Files/TPS Systems/CSERV/logs`

TPS® Command Service Logs files created:

<port number> Beginning socket type messages like listen, open, close. Also contains Socket Options passed in.

`errors` All error messages encountered

<nodename>:<port number> One log for each node communicating with the NFM Server. This log contains all traffic from that node to the NFM Server.

NFM Server Database Files

(For TPS® Technical Support)

In order to help reconstruct your NFM environment, TPS® Technical Support might need your NFM Server Database Files. This could give us some insight into or to help recreate your environment. Bundle up all files in the following directory (See 'NFM Server Database Components' for more information):

The NFM Server Database is located in the following directory:

Operating System	Directory
Windows	`C:\Program Files\TPS Sytems\NFMServer\dba\data`
UNIX	`/var/tps/dba/data`

** Files in the NFM Server Database directory are binary format.**

Core Files / Job Logs

(For TPS® Technical Support)

In the event of a NFM Server, NFM Client or TPS® Command Service crash, you should provide TPS® Technical Support with the appropriate Core File (UNIX) / Job Logs (Mainframe).

NFM Client Logs are recorded in the Job Log on the mainframe (see 'NFM Client Logs').

Gathering Platform Specific Information

(For TPS® Technical Support)

To start troubleshooting any problem, TPS® Technical Support needs certain platform specific information. This includes, by is not limited to:

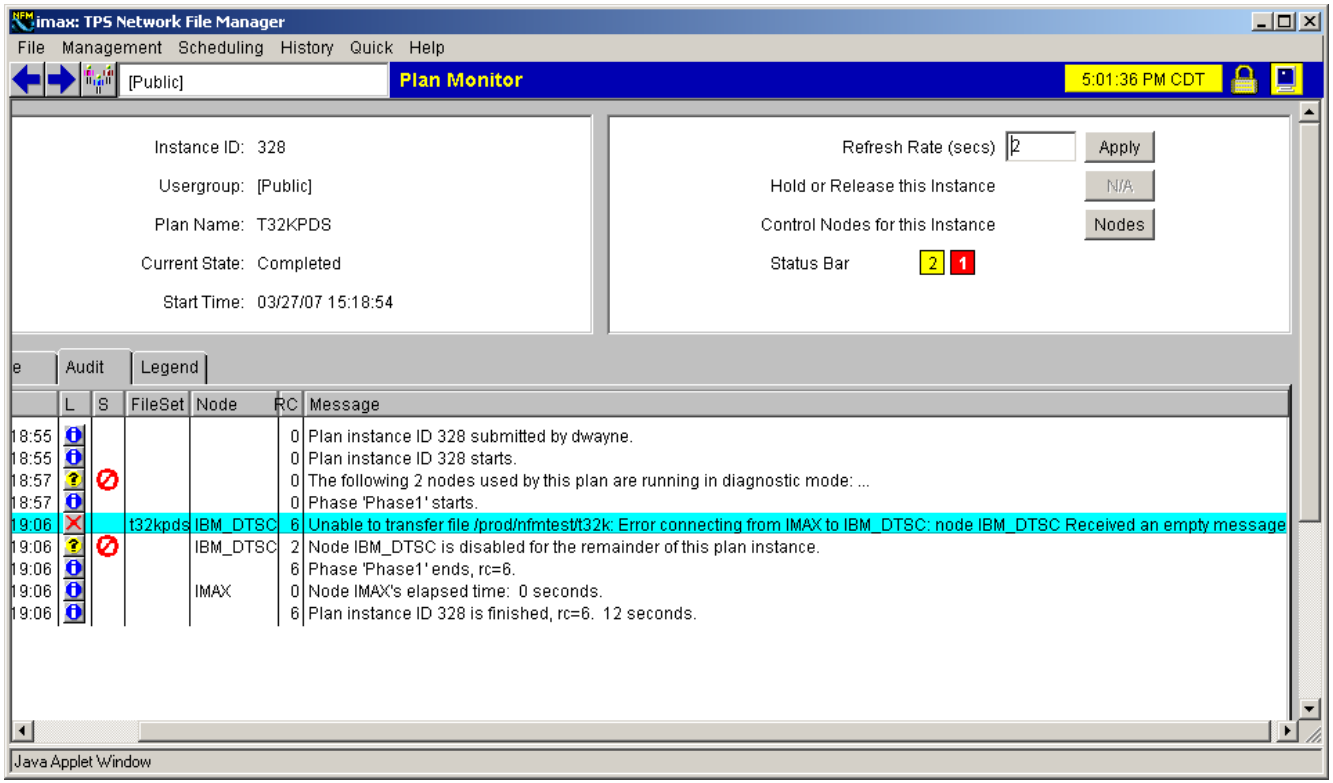
Operating System Level and Platform (including patch level)
NFM Server and NFM Client Version
Any Operating System Error Logs / Reports
File Permissions
`Inittab` or `Startup` Scripts for NFM

GUI Screen Captures

(For TPS® Technical Support)

A screen shot is sometimes the best way to show certain problems (especially problems displayed on the NFM GUI screen). We recommend sending screen captures of anything related to the problem. For example: If I see an error in the plan audit or audit screen I might send a screen capture of:

1) the audit screen containing the error *(Figure 4)*
2) the audit's plan *(Figure 5)*
3) the plan's fileset *(Figure 6)*

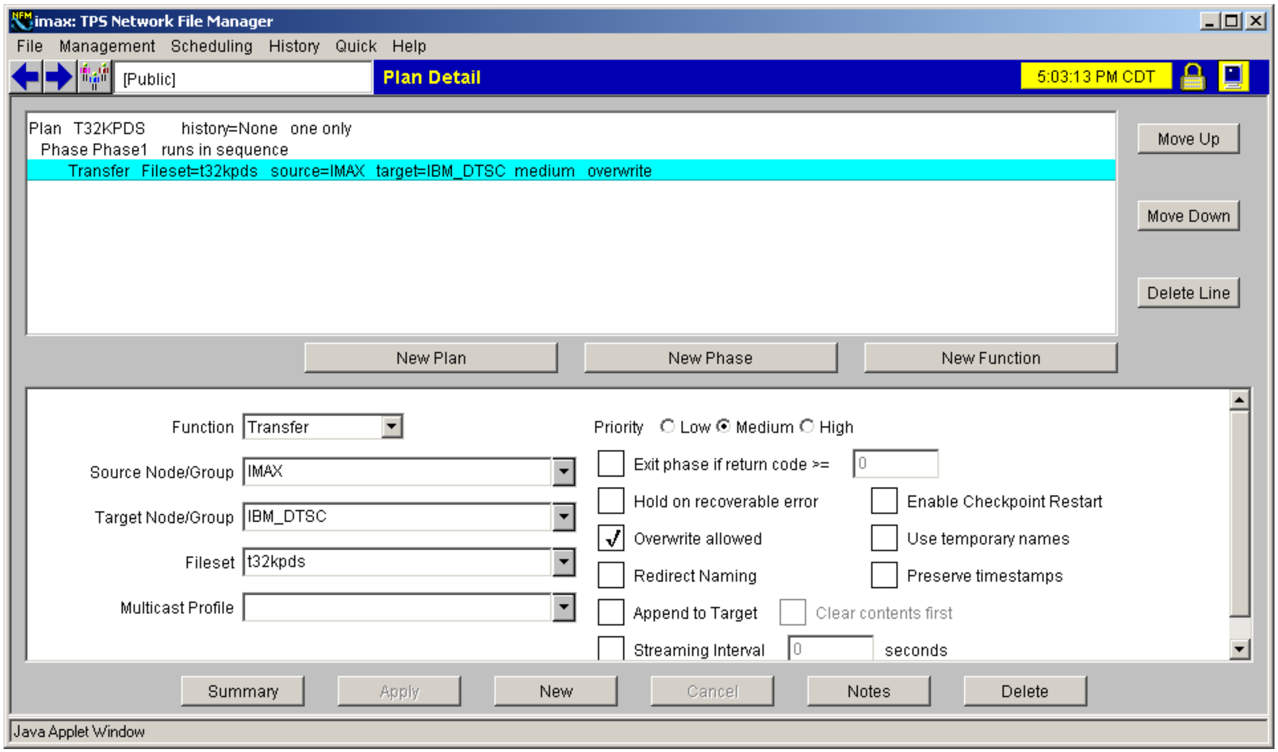


Instance ID: 328
Usergroup: [Public]
Plan Name: T32KPDS
Current State: Completed
Start Time: 03/27/07 15:18:54

Refresh Rate (secs) 2
Hold or Release this Instance
Control Nodes for this Instance
Status Bar 2 1

	L	S	FileSet	Node	RC	Message
18:55	ℹ				0	Plan instance ID 328 submitted by dwayne.
18:55	ℹ				0	Plan instance ID 328 starts.
18:57	?	⊘			0	The following 2 nodes used by this plan are running in diagnostic mode: ...
18:57	ℹ				0	Phase 'Phase1' starts.
19:06	✕		t32kpds	IBM_DTSC	6	Unable to transfer file /prod/nfmtest/t32k: Error connecting from IMAX to IBM_DTSC: node IBM_DTSC Received an empty message
19:06	?	⊘		IBM_DTSC	2	Node IBM_DTSC is disabled for the remainder of this plan instance.
19:06	ℹ				6	Phase 'Phase1' ends, rc=6.
19:06	ℹ			IMAX	0	Node IMAX's elapsed time: 0 seconds.
19:06	ℹ				6	Plan instance ID 328 is finished, rc=6. 12 seconds.

(Figure 4)



```
Plan  T32KPDS      history=None  one only
  Phase Phase1   runs in sequence
      Transfer  Fileset=t32kpds  source=IMAX  target=IBM_DTSC  medium  overwrite
```

Function: Transfer

Source Node/Group: IMAX

Target Node/Group: IBM_DTSC

Fileset: t32kpds

Multicast Profile:

Priority ○ Low ● Medium ○ High

□ Exit phase if return code >= 0

□ Hold on recoverable error

□ Enable Checkpoint Restart

☑ Overwrite allowed

□ Use temporary names

□ Redirect Naming

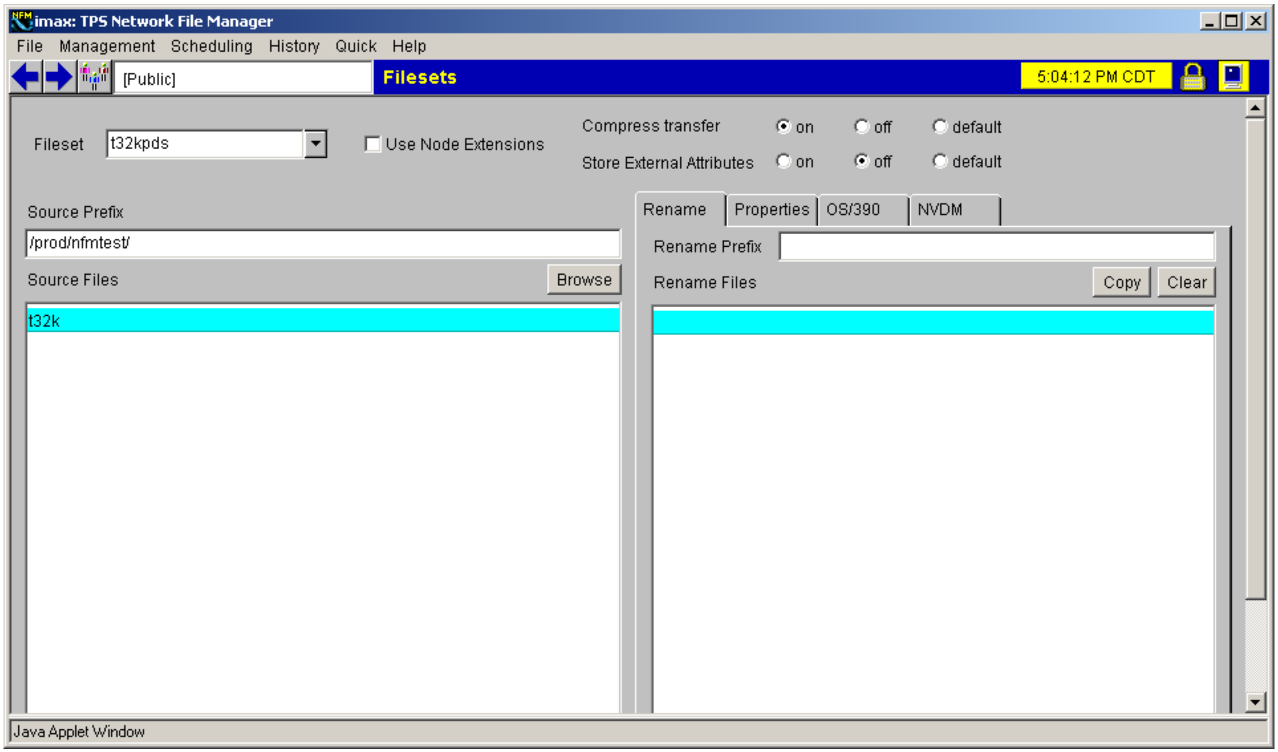
□ Preserve timestamps

□ Append to Target

□ Clear contents first

□ Streaming Interval 0 seconds

(Figure 5)



Fileset: t32kpds

□ Use Node Extensions

Compress transfer ● on ○ off ○ default

Store External Attributes ○ on ● off ○ default

Source Prefix: /prod/nfmtest/

Source Files: t32k

Rename Prefix:

Rename Files:

(Figure 6)

APPENDIX

Appendix A: Plan Return Codes

The plan return code (or `rc`) is set to the highest severity level of any error that occurred during a plans execution. By setting options in the plans phases and functions, this return code can be used to alter or terminate flow of execution. The values listed here will result from the functions other than type Execute. Execution functions will exit with `2` if the program could not be executed, otherwise, the function will exit with the exit code of the program itself.

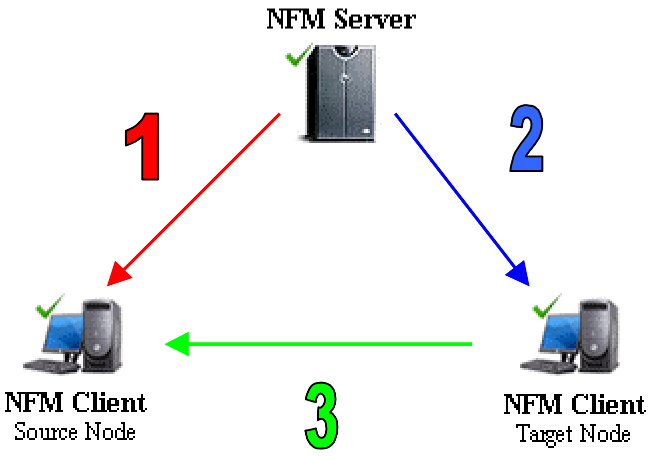
`0`	Successful.
`2`	Warning generated, or unexpected output from a non-NFM program during a file copy (such as FTP).
`4`	Individual file action has failed.
`6`	A node error occurred and will probably disable any further action with the node.
`8`	An error occurred on the NFM server, but the plan executing may be able to abort.
`10`	An error occurred on the NFM server severe enough to cause the plan executing to abort.
`20`	An error occurred severe enough to disable the NFM server.

Appendix B: Anatomy of a NFM File Transfer

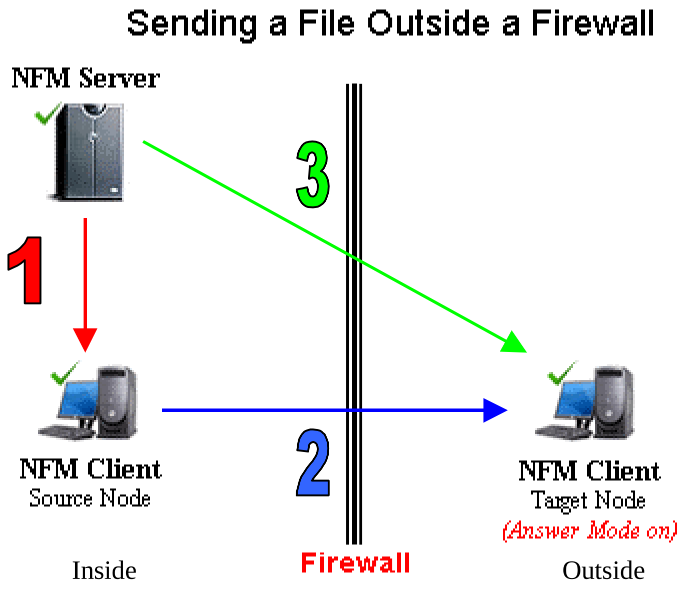
To understand the process of a file transfer within NFM, it is important to visualize the way NFM components interact with each other. When performing a file transfer several different file transfer methods are possible:

Transferring file(s) Inside a Firewall
Sending a file(s) Outside a Firewall
Receiving a file(s) from Outside a Firewall

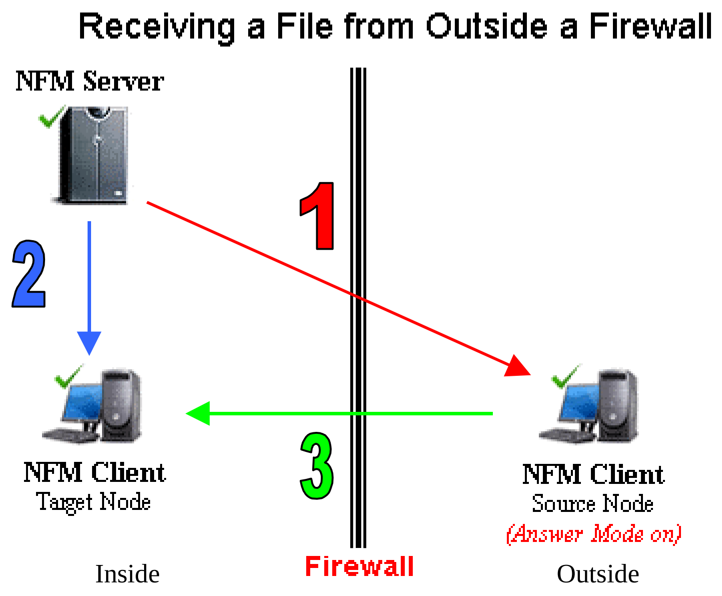
Transferring File Inside a Firewall



1. The NFM Server contacts the NFM Client Source Node on the port specified in the Node record (default `8008` (non-SSL) or `8009` (SSL)). It then retrieves directory and file information for each file in the Fileset.
2. The NFM Server contacts the NFM Client Target Node on the port specified in the Node record (default `8008` (non-SSL) or `8009` (SSL)). It then tells the Target Node to contact the Source Node.
3. Target Node contacts the Souce Node and starts retrieving the data. The target node reports back to the NFM Server status messages so the NFM Server can track the progress of the transfer.



1. The NFM Server contacts the NFM Client Source Node on the port specified in the Node record (default 8008 (non-SSL) or 8009 (SSL)). It then retrieves directory and file information for each file in the Fileset.
2. Using the same socket to communicate to the NFM Client (in Step 1), the NFM Server instructs the NFM Client Source Node to contact the NFM Client Target Node.
3. The NFM Server contacts the NFM Client Target Node and instructs him to use the socket in step 2 to retrieve the data. The target node reports back to the NFM Server status messages so the NFM Server can track the progress of the transfer.



(This is very similar to the "Transferring File Inside a Firewall" example)

1. The NFM Server contacts the NFM Client Source Node on the port specified in the Node record (default `8008` (non-SSL) or `8009` (SSL)). It then retrieves directory and file information for each file in the Fileset.
2. The NFM Server contacts the NFM Client Target Node on the port specified in the Node record (default `8008` (non-SSL) or `8009` (SSL)). It then tells the Target Node to contact the Source Node.
3. Target Node contacts the Souce Node and starts retrieving the data. The target node reports back to the NFM Server status messages so the NFM Server can track the progress of the transfer.

Appendix C: NFM in an High Availability Environment

The NFM application is designed to accommodate a High Availability environment. The functionality that supports this environment is inherent to the design because the system is effectively state driven by its database, and is largely tolerant to network interruptions and system restarts.

The NFM server controls all scheduling aspects of the NFM application. It also directs all client activity through a series of simple command requests that are issued as needed during plan activity. Because of this, a discussion of how NFM works with regards to High Availability is generally limited to how the NFM Server works. Any NFM clients that are available to handle requests from the NFM Server are assumed to still be available at the same designated IP addresses after a failover event. This may of course be a result of one or more of the NFM clients being part of a failover system themselves.

How things currently work

By examining certain aspects of how NFM functions, we will have a greater understanding of the behavior that can be expected in a High Availability environment.

Most NFM activity originates from running plans. When a plan is submitted from the plan-scheduling screen, a 'plan instance' is created in the NFM database. The plan instance will have a starting time (usually in the future) when it is scheduled to begin processing. All plan instances are in one of the following states:

SCHEDULED The plan is set to run in the future.

ACTIVE The plan is currently running.

FINISHED The plan is done running

CANCELLED The plan was deliberately halted prematurely (while it was active) by a user or program issuing a cancel request. A cancelled plan may be resumed.

INTERRUPTED The plan was prematurely halted (again while active) because the NFM server was halted and restarted. An interrupted plan may also be resumed.

It is this last state INTERRUPTED which is of primary concern when discussing the effects of a High Availability failover event. When a plan is ACTIVE (and only during this time), a process is loaded in memory that is responsible for performing the tasks associated with the plan instance. The database is altered during the running of the plan to record the progress of these tasks. This recorded progress facilitates the ability for a plan to be 'resumed' at a particular point, even after being either cancelled or interrupted.

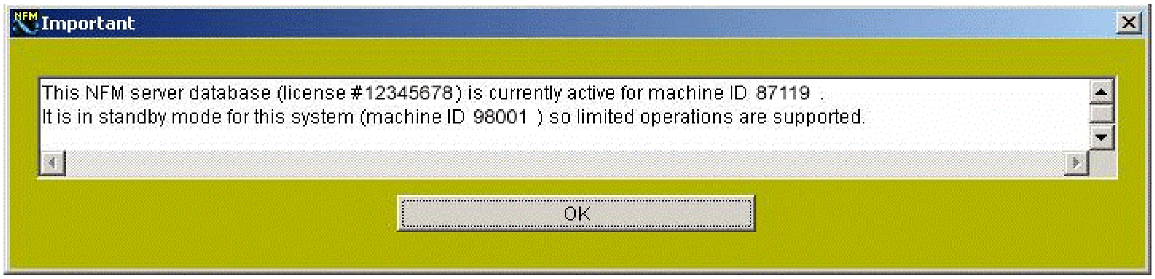
If the NFM server is halted (during a reboot for example), any active plan instances will still be marked as active in the database even though the process that was performing the work no longer exists. If the NFM server is subsequently restarted, it will find these ACTIVE plan instances and mark them as INTERRUPTED. The system may be configured to automatically resume plan instances that are INTERRUPTED after the NFM server is restarted (see System Settings / Transfers tab / Automatically Resume Interrupted Plans field).

Failing over to a backup NFM server

Stopping and starting the NFM server would naturally occur during a reboot of the NFM server computer, or a power outage for example. Similarly, taking the database from say a crashed NFM server computer and presenting it to a different NFM server would behave almost identically. This is the basis for a High Availability failover event involving the NFM server. The field “Automatically Resume Interrupted Plans” should always be checked on for a High Availability environment. This will cause the backup NFM server to automatically resume the interrupted plan instances when it is started up and activated.

The process of activating an NFM server against a particular database is a safeguard mechanism to prevent accidentally having two different NFM servers acting on the same NFM database at the same time (a real possibility if the database is being mirrored or network mounted). After a basic NFM server install, the database records the unique machine ID of the currently active computer (this occurs automatically). When an existing database is presented to a different computer running the NFM server, that NFM server will go into `standby` mode. It will not automatically go active until the command to activate it is issued. This can be done manually or more likely through a script that is run as part of a High Availability failover event. (The syntax for this command is simply: `nfm activate`.)

Failure to run the `nfm active` command and using the NFM GUI to login to the backup NFM Server will result in the following message:



```
This NFM server database (license #12345678) is currently active for machine ID 87119 .
It is in standby mode for this system (machine ID 98001 ) so limited operations are supported.
```

Since the backup NFM server is performing the exact same processing as the primary NFM server, it is imperative that the backup server provides an identical environment for processing as the primary server did. All network connections must of course be available using the same IP addresses or DNS names that are stored in the database. The NFM server node should reference the service IP address of the primary system, which should be taken over by the backup server.

The means by which the NFM database is made available to the backup NFM server is left up to the customer. The following points should be taken into consideration:

When the backup NFM server is started. The database, or a copy of it, must exist in the required location with the appropriate permissions set. Symbolic links may be used to accomplish this.

When the backup NFM server is started, the primary NFM server must not be allowed to access and modify the database. Although the activation process will help control this, it is still possible to have two different NFM servers access the same database if one is started while the other is still active.

File locations

The NFM server keeps all of its data in the `/var/tps` directory tree. Duplicating this directory tree in its entirety will result in a valid copy of the NFM server's database. This directory breaks down more specifically into the following components:

`/var/tps/cserv`	(TPS Command server files)
`/var/tps/dba`	(TPS Shared database files)
`/var/tps/nfm`	(NFM server specific files)
`/var/tps/nfmc`	(NFM client specific files)

In order to maintain a copy of data specific to the NFM server, you may include just the `/var/tps/dba` and `/var/tps/nfm` directories.

Limitations of resumed plans

As discussed previously, the NFM server keeps all plan progress recorded in its database. This information, besides being useful for monitoring activity is what makes the resuming of cancelled plans possible.

A resumed plan will attempt to continue processing at the exact point that it was previously interrupted (or cancelled). This is only possible with certain functions and then only to a certain degree. In general resuming a cancelled plan will allow the following:

> A file transfer operation will resume where it left off. If several files were being copied it will resume on the one it was in the middle of sending. If checkpoint restart (`CPR`) is enabled, it will resume in the middle of the file on the failed block.
>
> A delete operation will resume on the file it was processing previously.
>
> All other plan functions that were interrupted will be redone from the beginning. An execute function for example will be re-executed. It is important that the user setting up the plan take this into account, if they want to allow this plan to be resumed.

Appendix D: NFM Commands & Parameters

NFM Server Commands:

==

Command: `nfmfile`

Description: Tool used to interface with an NFM database. Used by an advanced NFM user or when they prefer the command line for extracting or updating NFM databases. This can also be used to extract / import NFM database into / from a text file.

Usage: `nfmfile` command [options]

Commands:

`Add`	: Add record to a file
`Daily`	: Toggle daily option for file
`Delete`	: Delete record from a file
`Deletem`	: Delete multiple records from a file
`dump`	: Dump file (formatted)
`export`	: Export database records to a flat file
`fields`	: List field components
`get`	: Get a specific record
`import`	: Import database records from a flat file
`list`	List file entries
`put`	: Add or update record
`table`	: Show possible values for a field
`tags`	: List tags for a file
`testfile`	: Create a test file (use -f and -z)
`update`	: Update a record
`memberm`	: List member members of a group file
`memberg`	: List member groups of a group group file
`nonmemberm`	: List non member members of a group file
`nonmemberg`	: List non member groups of a group file

Options:

`-a fieldtag=value`	: Specify primary key (`get`)
	: Specify all fields to set (`add`,`update`)
`-A ascii_file`	: Ascii file for import/export.
`-c maxcount`	: For list, maximum number of entries
`-d`	: Do not show duplicates (list).
`-D`	: Debug trace on.
`-e`	: Don't show error if 0
`-f filename`	: Filename to act on (most commands).
`-F string`	: Specify new field separator string

`-g groupname`	: Group name (for nonmembers only
`-l`	: Use line output instead of column
`-p datapath`	: Specify alternate path to file
`-t fieldtag`	: Specify which fields and in which order to display (list)
`-r`	: Remote mode
`-R string`	: Specify new record separator string
`-s sorttag`	: Tag name to sort list by
`-S`	: Sort by descending order
`-v value`	: Positionally dependent field value
`-T`	: Print tags on output
`-V`	: Verbose output
`-z size`	: Size

==

Command: **`nfm`**

Description: Used as part of the NFM Command Line Interface. Allows user to bypass the NFM GUI and interact directly with the NFM Server. This can include tasks such as scheduling plans, displaying or parsing through `AUDITS`, or any subcommand listed below.

Input Usage: `nfm subcommand,parm1,parm2, parm3 . . .`

Subcommands (`nfm help` displays all subcommands, `nfm help,<subcommand>` displays parameters):

`export`	Export records from NFM database to text file.
`exportx`	Export records using wildcard specifiers.
`import`	Import records from text file into NFM database.
`browsedir`	Display directory listing from a node.
`browseroot`	Display root directory from a node.
`callplan`	Run a plan and wait for its completion.
`canceljob`	Cancel a running plan instance.
`cancelplan`	Cancel all plan instances of a particular plan name.
`delay`	Wait for specified number of seconds.
`deleteasource, deleteday, deletefileset, deletehour, deletejob, deletemodel, deletenode, deletenodegroup, deleteplan, deleteuser`	Delete specified record items.
`enablejobnode`	Continue a held node in a plan instance**.**
`excludejobnode`	Exclude a particular node in a plan instance.
`getaudit`	Print a particular audit.
`getday, getfileset, gethour, getjobstatus, getmodel, getnode, getplan`	Display specified record items.
`holdjob`	Hold a plan instance.
`holdjobnode`	Hold a particular node in a plan instance.

listaudits Print a series of audits.

listclients, listdays, listfilesets, listhours, listjobs, listjobsx, listmodels, listnodes, listnodegroups, listnodemembers, listplans, listusers
List all the items of the specified record type.

putasource, putaudit, putday, putfileset, puthour, putmodel, putnode, putplan
Add or update specified record items.

resumejob Continue a held or canceled plan instance.

retryjob Retry failed operations in a finished plan instance.

setjobenv Set an environment variable for subsequent use within a plan.

statnode Contact a node and retrieve NFM client version number.

submitplan Submit a plan (create a plan instance).

Output

The output from the `nfm` command will vary based on the subcommand. It may be multiple lines but the final line will always be the error message line and will look like this:

```
:error:#:error_message
```

The `:error` indicates this to be the error message line. The :# indicates an NFM type error code, or zero for success. The `error_message` will contain an error message if the return code is non-zero. The following examples show the failure and success of deleting a user as well as a command with multiple line output:

```
in>  nfm deleteuser,skippie
out> :error:200:Record not found

in>  nfm deleteuser,doug
out> :error:0:

in>  nfm listusers
out> christine
     jack
     sam
     :error:0:
```

Unless otherwise indicated in the following section, all subcommands will generate a single line of output as listed in the first two examples above.

==

Command: `nfmreorg`

Description: Used to reorganize or restructure the NFM databases. This is sometimes required when upgrading / downgrading NFM Server.

Usage: `nfmreorg` [options]

Options:

`-f filename`	: Specify which file (otherwise all)
`-p pathname`	: Specify alternate path to files
`-i`	: Initialize file
`-r`	: Reorganize (otherwise just show)
`-t`	: Trace on
`-T`	: Verbose trace on

==

Command: `nfmcom`

Description: Used to communicate with a NFM Client to get directory listings, execute a command, or shutdown a client node.

Usage: `nfmcom` comand [options]

Commands:

`Browse`	: Browse a node
`Dirinfo`	: Dirinfo a node (for testing only)
`Echo`	: Echo a node
`Rsh`	: Remote shell to a client
`Shutnode`	: Shutdown a node client

Options:

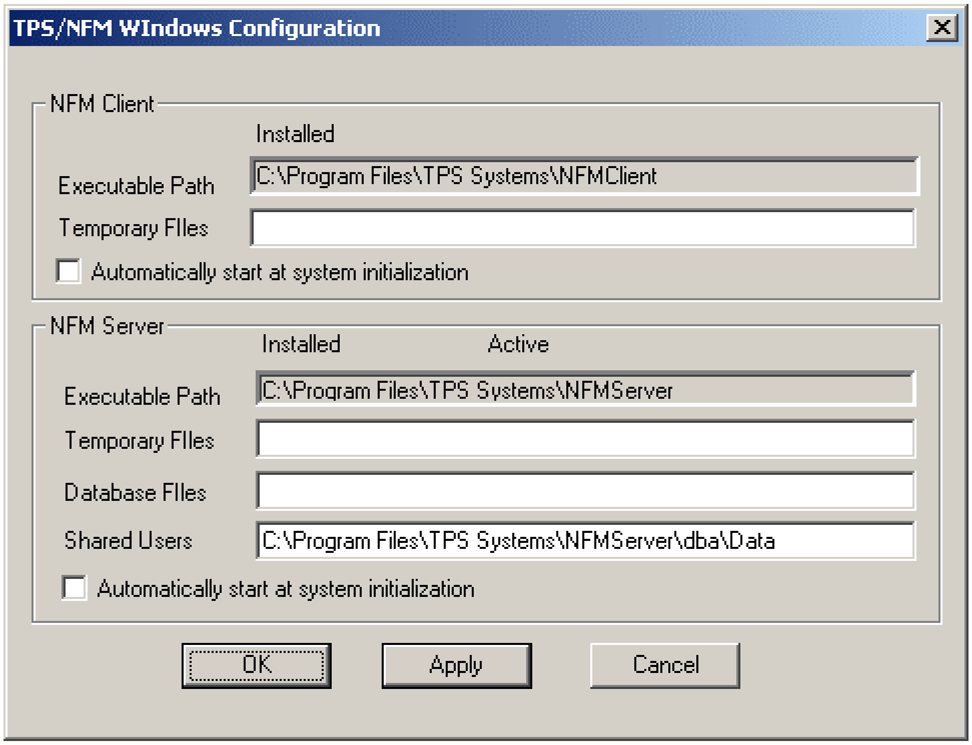
`-c command`	: Command (`rsh`)
`-d director`	: Directory (`browse`)
`-e`	: Extended tree (`browse`)
`-i`	: Usergroup ID #
`-n nodename`	: Node name (`browse`, `echo`, `rsh`, `shutdown`)
`-m #`	: Maximum count # (`browse`)
`-s #`	: Starting entry # (`browse`)
`-t [GV]`	: Debug trace `on`
`-R`	: Get `root` directory

==

Command: `nfmconfig` (Windows Only)

Description: Changes the default registry settings and startup options.

Usage: `nfmconfig`



TPS/NFM WIndows Configuration

NFM Client

Installed

Executable Path: C:\Program Files\TPS Systems\NFMClient

Temporary FIles:

☐ Automatically start at system initialization

NFM Server

Installed Active

Executable Path: C:\Program Files\TPS Systems\NFMServer

Temporary FIles:

Database FIles:

Shared Users: C:\Program Files\TPS Systems\NFMServer\dba\Data

☐ Automatically start at system initialization

==

Command: `nfmparse`

Description: A stand-alone program that can separate records from a single data file into multiple files that may be destined for multiple nodes. The `nfmparse` program also performs the reverse operation of combining files into a single data file (See NFM Parse for more information).

Usage: `nfmparse <options>`

where options are `<name>` or `<name>=<value> pairs`:

`parse (no value)`	: Parse action (default)
`combine (no value)`	: Combine action
`options`	: name of a text file containing more options
`file`	: input file name (default is `stdin`)
`delimited`	: input records are delimited by this string
`fixed`	: input records are fixed length, this many bytes
`pad`	:outpad character for fixed length (default ASCII space) (combine only)
`nodesubstitution`	: printf style format string to convert IDs to node names

nodetable : name of a file containing <ID> space <nodename>
filetable : name of a file containing <ID> space <filename>
memberdirectory : directory for member files (default is current directory)
maxrecord : maximum input record length + 1 (default is 2048)
verbose : output verbose messages to stdout per-header options
header : file header sentinel string
headerposition : offset of header sentinel within header record
headerinit : initial data string for header (combine only)
membername : output file name for this header type
storeposition : offset of ID within header record
storelength : length of ID field within header
filesizeposition : offset of 4 byte data size field within header record
filenameposition : offset of filename within header record
filenamelength : length of filename within header
dateposition : offset of date field within header record
dateformat : format of date field within header (default MM/DD/YY)
memberlength : output record length
memberpad : outpad character for fixed length (default ASCII space)
memberdelimiter : output record delimiter string
membereof : output record to end of file
copyheader : include header records in output files if copyheader=1

==

NFM Client Commands:

Command: **nfmi**

Description: Used as part of the NFM Remote Line Interface. Allows the NFM Client to bypass the NFM GUI and interact directly with the NFM Server. This can include tasks such as remotely scheduling plans.

Usage: `nfmi [options] parm=value parm=value ...`

Options:

d : Go directly to diagnostics
-i : Run interactively

`-s port_num`	: Specify an alternate port #
`-t`	: Turn on trace
`-v`	: Display version info
`-V`	: Run in verbose mode
`-x`	: Shutdown local nfm client
`-H directory`	: Specify alternate home directory
`-p`	: Specify a node password on the client

Command: `nfmi -id`

Description: Runs the `nfmi` command in interactive mode with diagnostics turned on. This can be used to test communications between two nodes.

Diagnostic commands are:

`connect hostname[:portnum]`	: Connect to a node
`create`	: Create big file
`dump`	: Hex dump a file
`dumpf`	: Hex dump a file
`echo`	: Echo node
`encrypt`	: Toggle encrypt
`help`	: Help
`fetch`	: Fetch file from client
`fread`	: Fetch file directly
`flush`	: Flush
`pass`	: Set a node password
`pause`	: Pause for debug
`quit`	: Quit diag mode
`rtrace`	: Toggle remote trace
`status`	: Display settings
`trace`	: Toggle local trace
`diag`	: Diagnostics client test
`4690`	: Test seek on file

==

Command: **`nfmclient`**

Description: Starts the NFM Client.

Usage: `nfmclient [options]`

Options:

`-b`	: Run background
`-B`	: Enable blocking I/O
`-d`	: Do not delete files
`-e (env variable)`	: Define environment variable
`-E`	: Trigger memory trace on error (used with `-M`)
`-f`	: Run foreground (WINDOWS only)
`-G[1,2]`	: Ghost reading and writing
`-H directory`	: Specify alternate home directory
`-i`	: Memory integrity test
`-I resource_name`	: Show status of priority resource
`-l #`	: Log maximum size (`default=100000`)
`-J #`	: Create log on memory jump of (#) (4690 diagnostic)
`-L logname`	: Specify alternate log base name
`-M`	: Trace to memory buffer
`-m #`	: Maximum connection backlog (`default=1024`)
`-N`	: No prioritization
`-n`	: Do not allow non-blocking socket operations
`-p`	: Change the node password (prompts for password)
`-q workers`	: Thread pooling maximum worker threads (0 turns off thread pooling)
`-P password`	: Change the node password (with the supplied password)
`-R`	: Read separately (don't use `send_file`)
`-r (directory)`	: Establish virtual `root` directory
`-s (port #)`	: Specify alternate socket #, (`default=8008`)
`-S`	: Debug trace log file only, no stdout
`-t[GV]`	: Debug trace on
`-T`	: Debug trace stdout only, no log files
`-u`	: Do not send user and group info
`-v`	: Display compatability version #
`-w directory`	: Establish working directory
`-x`	: Export translate tables
`-Q`	: Pause children for debug
`-Z`	: Pause for debug
`-z`	: Time in milleseconds to sleep between file open retries, `default=1000`. (Windows only)
`-1`	: Use single thread model (testing)
`--nofiles=#`	: Set maximum number of open files for client process

Appendix E: Email Notification with SMTP Client

The NFM Server comes with an SMTP Client executable that, if coded correctly, can notify individuals via email when a Plan Phase, Plan Function or an entire Plan fails. The SMTP Client is a command line driven program that can run on both UNIX and Windows. The SMTP program can be found in `C:\Program Files\TPS Systems\NFMServer\` (Windows) or `/usr/lpp/tps/nfm/bin` (UNIX).

The following shows the command line options for the SMTP program:

Usage: `smtpclient [options] [recipients ...]`

Message Header Options:

`-s SUBJECT`	subject line of message (`Subject`:)
`-t EMAIL`	address of the recipient (`To`:)
`-f EMAIL`	address of the sender (`From`:)
`-r EMAIL`	address for replies (`Reply-to`:)
`-e EMAIL`	address for errors (`Errors-to`:)
`-c EMAIL`	address for copies (`Cc`:)
`-n EMAIL`	address of the organization (`Organization`:)

Message Body Options:

`-B #`	number of blank lines for message body
`-T TEXT`	line of text for message body
`-F FILE`	text file name to insert in message body
`-A FILE`	file to send as attachment
`-M`	use `MIME`-style translation to quoted-printable

Message Options:

`-p #`	set priority (`High:1-Normal:3-Low:5`)
`-R`	request email read notification

Processing Options:

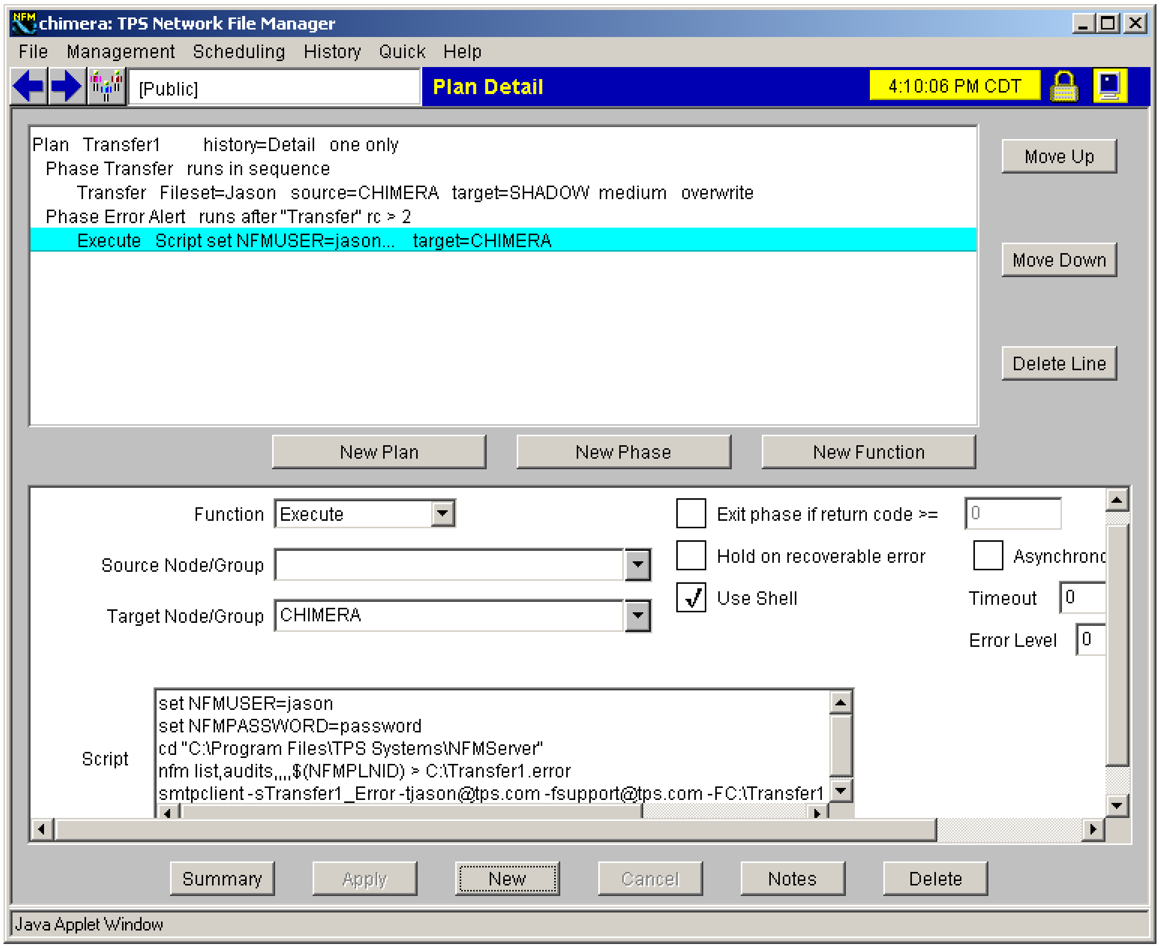
`-S SERVER`	TCP/IP host where SMTP server can be contacted
`-P PORT`	TCP/IP port where SMTP server can be contacted

Giving Feedback:

`-q`	disable normal output to terminal
`-v`	enable verbose logging messages

`-V`	display version string
`-?`	display this page

Example 1: Uses the SMTP Client program to notify someone, via email, when a plan fails:



```
Plan  Transfer1       history=Detail  one only
  Phase Transfer  runs in sequence
      Transfer  Fileset=Jason  source=CHIMERA  target=SHADOW  medium  overwrite
  Phase Error Alert  runs after "Transfer" rc > 2
      Execute  Script set NFMUSER=jason...   target=CHIMERA
```

Function: Execute

Source Node/Group:

Target Node/Group: CHIMERA

□ Exit phase if return code >= 0

□ Hold on recoverable error

□ Asynchrono

☑ Use Shell

Timeout: 0

Error Level: 0

Script:

```
set NFMUSER=jason
set NFMPASSWORD=password
cd "C:\Program Files\TPS Systems\NFMServer"
nfm list,audits,,,,$(NFMPLNID) > C:\Transfer1.error
smtpclient -sTransfer1_Error -tjason@tps.com -fsupport@tps.com -FC:\Transfer1
```

This plan contains 2 Plan Phases: Phase Transfer and Phase Error Alert.

In the first phase (Phase Transfer), a file transfer transfers a set of files from one node to another. If that phases returns `> 0`, meaning a warning or error was generated, the second phase (Phase Error Alert) will run.

In the Error Alert Phase we have defined an execute function (with `Use Shell` option checked) that will run the SMTP Client program notifying us that this plan has an error or warning.

```
set NFMUSER=jason
set NFMPASSWORD=password
cd "C:\Program Files\TPS Systems\NFMServer"
nfm list,audits,,,,$(NFMPLNID) > C:\Transfer1.error
```

```
smtpclient -sTransfer1_Error -tjason@tps.com -
fsupport@tps.com
       -FC:\Transfer1.error -Smailhub.tpssys.com
```

The `set NFMUSER=` and `set NFMPASSWORD=` lines defines a user that is able to list audit records. ** Set is a Windows command, use export if UNIX. **

If this is Windows and you don't have `nfm.exe` on your `PATH` you need to change directory to the `nfm.exe` file. `cd C:\Program Files\TPS Systems\NFMServer."` If this is UNIX you can ignore this step.

`nfm list,audits....` extracts the audits for this plan id (using the NFM environment variable `$(NFMPLNID`) redirecting it to `C:\Transfer1.error`.

`smtpclient ..` sends attaches the file `C:\Transfer1.error` and sends the email. Please consult the command line options above for detailed syntax.

The `nfm list,audits` command has several options that will allow you to filter audits. One option is the `level` option so you can only display warnings, errors or informational messages. Please consult your "NFM User's Guide Chapter 8, NFM Command Line Interface" for more details on the `nfm list, audits` command.

The email that is sent contains all audit messages up to the point `nfm list,audits` was executed.:

```
05/22/2007:16:24:03,273,[Public],INFO,READ,Transfer1,29,,,0,0,0,Plan  instance  ID
29 submitted by root.,

05/22/2007:16:24:03,274,[Public],INFO,READ,Transfer1,29,,,0,0,0,Plan  instance  ID
29 starts.,

05/22/2007:16:24:03,275,[Public],INFO,READ,Transfer1,29,,,0,0,0,Phase   'Transfer'
starts.,

05/22/2007:16:24:03,276,[Public],ERROR,UNREAD,Transfer1,29,Jason,SHADOW,4,1000,0,
Unable to create file C:/formulas.pdf on target node, No such file or directory.,

05/22/2007:16:24:03,277,[Public],INFO,READ,Transfer1,29,,,4,0,0,Phase   'Transfer'
ends, rc=4.,

05/22/2007:16:24:03,278,[Public],INFO,READ,Transfer1,29,,,0,0,0,Phase      'Error
Alert' starts.,

:error:0:
```

Index

Index