Securing FTP on MPE/iX

Author : Murali P N

1 Table of Contents

2 Revision History

VERSION	DATE	DESCRIPTION OF REVISION
0.1	2006-07-05	Initial draft
0.2	2006-07-28	1st revision (Jeff V)
0.3	2007-03-02	2nd revision (Murali P N)
0.4	2007-03-28	3rd revision (Murali P N)
0.5	2007-04-27	4th revision (Murali P N)
0.6	2007-05-14	5th revision (Murali P N)

3 Executive Summary

This paper explores methods to increase FTP/iX security based on several recent FTP/iX enhancements. These recent enhancements reflect one of HP's responses to the growing number of security related audits facing IT professionals tasked with implementing modern, robust security.

These FTP/iX enhancements were driven via the (now defunct) System Improvement Ballot (SIB) process whereby customers nominated and then voted for various MPE/iX enhancements. Increasing FTP/iX security was voted in the top two out of over a dozen requests.

This paper discusses the strengths and limitations of FTP/iX with respect to overall security requirements, with examples covering proper configuration and usage. In cases where FTP/iX does not satisfy the strictest audit requirements, non-MPE alternatives are presented in text and example. Section 5 is geared to the IT manager and provides an overview of the most significant FTP/iX enhancements, without going into configuration and usage details. Section 6 is targeted to the System or Security Administrator and describes in-depth each enhancement covered in Section 5, with examples of feature usage, configuration, and default settings.

Briefly, these security enhancements are:

- Restricting unauthorized users from logging on to an FTP server,
- Restricting unauthorized users from retrieving certain files on an FTP sever,
- Quarantining certain FTP/iX users to single directory roots,
- Logging all FTP commands and all file transfers from both the server and client side,
- Preventing FTP users from *rename*, *delete*, and *overwrite* file operations,
- Disallowing read access of the NETRC configuration file (which contains sensitive logon data),
- Password hiding when running FTP/iX in debug mode.

Section 7 describes a few methods to enhance security of FTP/iX in addition to the recent security enhancements. Some of the alternatives discussed are

- An envelop FTP/iX script that provides encryption of the data transfer between hosts
- Using non-MPE intermediaries like HP-UX to facilitate secure FTP communication
- Porting of Open SSH on MPE/iX to provide secure data transfer
- Use of a firewall for sockisified FTP
- Hardware solutions for enhanced security

The intended audiences for this paper are MPE/iX system managers, security architects, and IT personnel. HP also has two FTP/iX Communicator articles on Jazz at: http://jazz.external.hp.com/papers/Communicator/index.html. See *FTP Phase I* and *Phase II Enhancement* articles.

4 Introduction

FTP [File Transfer Protocol] is one of the oldest and most popular internet services, serving as an easy and effective method by which to transfer files over a network. The information transferred through FTP can be anything from a simple file transfer to data necessary for authorizing clients to server login (via user and password commands). The increasing reliance on FTP to transfer ever more vital electronic information makes FTP more vulnerable to attacks from unauthenticated users. This represents a security risk whereby passwords can be stolen through the monitoring of local and wide-area networks. This aids potential attackers through password exposure and may limit accessibility of files by FTP servers which have been configured to not accept the inherent security risks. Certain security risks also exist where an authorized user may (accidentally or deliberately) retrieve, delete, rename and/or overwrite certain files residing on an FTP/iX server.

The following three relatively new laws or initiatives increase the incentive to transfer data in secure manner:

- Sarbanes-Oxley (2002),
- HIPAA (2001), and
- California Security Breach Notification Act (2003).

These new regulations, combined with pressure from E-commerce partners, are the major reasons for customers to search for secure MPE/iX file transfer solutions. Responding to these regulations and customer requests, HP has implemented several enhancements to increase the security of FTP/iX.

5 FTP/iX Security Overview

FTP/iX is based on RFC 959, which does not address encryption or user authentication. If these two security areas are essential for FTP file transfers we offer some solutions in the Alternatives section. Please note that

modern user authentication is beyond the scope of this paper. However, FTP/iX now offers better security than prescribed by RFC 959 in several key areas, which are described below and, in detail in Section 6.

- **User Restriction**: The FTP/iX server follows the current user ID/password authentication mechanism to validate users; however, this enhancement provides system managers the ability to restrict FTP access to specified users, even if they know the password. The configuration file, FTPUSERS.ARPA.SYS, can be populated with user names who will be denied logon access to the FTP/iX server. So, users appearing in this file cannot logon to FTP, while users who are not listed in this file are entitled to normal FTP logon authentication. All other MPE/iX security mechanisms are unaltered. An exception to this rule is that users with SM capability bypass the FTPUSERS configuration file, and thus are permitted to logon to the FTP/iX server (assuming a valid user name and password). Absence of the FTPUSERS file, which is the default, indicates that no users are being restricted other than normal logon requirements. Any changes in this file will get reflected in the next FTP login session.
- **File Retrieval Denial**: Under normal circumstances, there are no restrictions imposed by FTP/iX for retrieving files. The MPE/iX operating system's security may prevent file access but, up until now, FTP did not impose its own rules. The NORETRIEVE option in the new FTPACCES.ARPA.SYS file boosts the inherent security of FTP/iX by enabling the system administrator to restrict access to one or more files independent of the FTP user ID, and independent of MPE's security. FTPACCES is a configuration file which can contain a list of files which all FTP users will be denied access. It names files, or groups of files (i.e. files contained in groups, accounts, or directories), which will not be transferred by FTP/iX regardless of the underlying MPE security. As with the FTPUSERS file, users with SM capability are exempted from this enhanced file security rule. Absence of the FTPACCES file, which is the default, indicates that there are no additional file restrictions enforced by the FTP/iX server. Any changes to this file will get reflected in the next FTP logon session.
- **CHROOT**: The *chroot* FTPACCES.ARPA.SYS configuration option forces a specified user to be confined to a single group or directory (and below).when logging on to the FTP/iX server. If chroot is in effect then the FTP user is limited to this location and directories below it. This option limits the inbound FTP commands cd, put, get, mput, mget and dir to the configured root. Users will be unable to move up beyond the specified root location. In addition, users will not be permitted to reference files outside of their chroot location. As with the first two enhancements, this restriction does not apply to users with SM capability. By default, the FTPACCES file does not exist and thus all users are authorized to name and access all files within MPE's security guidelines.
- **Restriction on File Permissions**: New options "PERMISSION_DELETE", "PERMISSION_OVERWRITE" and "PERMISSION_RENAME" can be set in the existing configuration file, SETPARMS.APRA.SYS. When these options are enabled the corresponding action is denied. For instance, if PERMISSION_RENAME is set to 'ON', renaming of files, within the context of FTP/iX, is not allowed, regardless of the corresponding MPE file security. The default setting for these options is 'OFF', and these restrictions do not apply to users with SM capability.
- **Logs commands and file transfer statistics:** LOG_COMMANDS and LOG_TRANSFERS are new options in the SETPARMS.ARPA.SYS configuration file which enable FTP commands and file transfer logging. When these options are set to ON, the FTP client-server transactions and communications are logged. Log messages are recorded in the file FTPLOG##.ARPA.SYS, where ## indicates a two digit number in the range of 00 to 99.
- **NETRC file:** The NETRC file is an existing feature of the FTP client which facilitates automated logon to a remote host. Prior to this enhancement, the FTP client required read access to NETRC to establish this automated logon, which also meant that FTP users could read this file and gain password and user IDs. A new security enhancement has been added to the FTP client so that it can still read the NETRC file, but MPE security can deny read access to other users.
- **Prevent display of passwords in DEBUG mode:** The *debug* command at the FTP client places FTP into a diagnostic mode, whereby the client's and servers internal commands are displayed on the $stdlist of the client. If debug mode is turned on prior to an *open*, then the user ID and password are visible in plain text. In order to make FTP/iX compliant with SAR-OX, a new option, DEBUG_PASS, has been introduced and is set in the SETPARMS.ARPA.SYS file. If DEBUG_PASS is set to OFF (default), the password is not echoed, even in debug mode.

- **Banner:** This new feature allows the display of an FTP/iX welcome message. This enhancement is enabled by creating a file named FTPHELLO.ARPA.SYS, and adding the desired FTP banner text. The entire FTPHELLO file is displayed to $stdlist after a successful logon. FTPHELLO supports special substitution characters for values pertaining to the FTP connection. For instance, %T displays the server time; %R displays the remote host name, etc. Any user can create and modify this file provided the user has write access to ARPA.SYS. By default the FTPHELLO file does not exist, and hence FTP will display only its customary one line banner. Any changes to this file will get reflected in the next FTP logon session.

The above features do not capture all aspects of FTP security. For instance, robust user authentication and encrypted transmission of FTP commands and data are absent from this collection of FTP/iX enhancements. The Alternatives Section, at the end of this paper, describes a potential solution for file encryption.

The following section covers the detail of how to configure and utilize the new security enhancements described above.

6 FTP/iX Security Details

This section explains how to properly configure and use the new FTP/iX security features, and is divided into three sub-sections. Each sub-section describes how to build the different configuration files: FTPUSERS, SETPARMS, and FTPACCES.

6.1 FTPUSERS.ARPA.SYS

This file contains one or more user names that will be denied logon to the FTP/iX server, unless the user has SM capability. A hash mark (#) is used to denote a comment, as seen in the sample below.
Syntax: [UserName.]AcctName, one name per record.

6.1.1 Configuring FTPUSERS file

Sample configuration file FTPUSER:

```
# Purpose: to deny the list of users below, logon to the FTP/iX server running on this system.
# Syntax: [UserName.]AcctName  one name per line
# Wildcards, e.g., "@", are not supported. Leading and trailing spaces are ignored,
# and all text is case insensitive.
# Example: to deny FTP logon to the user MGR.PROD enter "MGR.PROD" below on a
# separate line (and without the quotes).
# Example: to deny FTP logon to all users in the PURCH account enter "PURCH" below on a
# separate line (and without the quotes).

# will restrict user Testmgr of SYS account.
Testmgr.SYS
# will restrict all the users of TELESUP account.
TELESUP
# will not be restricted as the user has SM capability.
MANAGER.SYS
```

The FTPUSERS file is not created automatically, thus the FTP/iX default is to *not* restrict logon based upon user ID. However, the absence of a particular user ID in the FTPUSERS file does *not* exempt the user from entering the necessary passwords so that MPE/iX can authorize the user on the FTP/iX server.

6.1.2 FTPUSERS configuration rules

Note: if this file is missing (default) or empty then there are no additional FTP/iX user ID based logon restrictions in place on the FTP/iX server. However, normal MPE/iX user validation is enforced, as always.

- ✓ FTPUSERS.ARPA.SYS may be created and edited with any supported editor. This file should be kept unnumbered, fixed width, ASCII with a record-width of not more than 72 bytes. The file can be up to 4GB in size, but the performance of the linear scan will be a limiting factor.
- ✓ The user names must appear one per line.
- ✓ The user names can be specified in two formats:
  - {username}.{accountname}: The specific user of the specific account will *not* be allowed to logon to the FTP/iX server.
  - {accountname}: All the users from the specified account will *not* be allowed to logon to the FTP/iX sever.
- ✓ Comments begin with"#". Embedded comments are not recognized Users with SM capability (such as MANAGER.SYS) are *not* restricted by the FTPUSERS configuration.
- ✓ Specification of the account name or user.account can include leading or trailing whitespace characters, and is not case sensitive. Upper, lower, and mixed case names are treated the same.
- ✓ In the case of redundant or ambiguous entries, the first file entry to match the user's FTP/iX logon ID is used. Thus, more specific entries, such as a certain user.account, should precede more generic entries, such as an account name.
- ✓ Invalid entries will silently be ignored without logging.
- ✓ Wildcards are not supported.

- ✓ Any changes in FTPUSERS file will get reflected during the next FTP login. The changes will also get reflected if the user issues a login command (e.g. user smith.sys) within an already running FTP session.

6.1.3 Examples

Let us assume a FTPUSERS.ARPA.SYS file as listed above (refer section 6.1.1):

This FTPUSERS file prevents the users TESTMGR.SYS, OPERATOR.SYS and all users in the TELESUP account from logging on to the hosting FTP/iX server.

1. If a user tries to logon FTP/iX as TESTMGR.SYS, the following error is displayed on the users $stdlist:

```
ftp> user testmgr.sys
530 User log on unsuccessful
User not logged in. (FTPERR 65)
Remote system type is MPE/iX
```

2. If a user attempts to logon to FTP/iX in the TELESUP account, the following error is reported:

```
ftp> user MGR. TELESUP
530 Logon failed, restricted in FTPUSERS.
User not logged in. (FTPERR 65)
Remote system type is MPE/iX.
```

3. If a user tries to logon to FTP/iX as MANAGER.SYS, FTP/iX will **not** restrict logon since this user as user has SM capability. However, the user will still need to provide the correct user and account passwords.

```
ftp> user manager.sys
230-'/SYS/PUB'
230-"pass1"
230-"pass2"
230-'12:51 PM'
230-end of ftp hello file
230 User logged on
Remote system type is MPE/iX
200 Type set to I.
```

If FTP Console Logging is enabled, the FTP/iX server will generate an error message of this type on the console:
13:00/#J4/83/FTP LOGON RESTRICTED FOR: "MGR.TELESUP" IP=aaa.bbb.ccc.ddd

6.2 FTPACCES.ARPA.SYS

This file implements two different FTP restrictions. The first is supported by the *NORETRIEVE* option, which prevents the FTP user from retrieving any of the listed files. The second restriction is the *CHROOT* option, which quarantines a user to a specified location in the FTP server's directory structure. The FTPACCES file is not created automatically, thus the FTP/iX default is to not impose extra restrictions on any file for any user.

General Configuration rules for FTPACCES

- ✓ Leading or trailing white space character(s) are neglected
- ✓ The FTPACCES file supports only three types of entries chroot, noretrieve and # (comment). Lines that do not start with any of the three keywords chroot, noretrieve or '#' are considered as invalid entries and are silently ignored.
- ✓ Comments are introduced by the hash (#) character and must start a new line.
- ✓ Users with SM capability (like MANAGER.SYS) are not restricted by the FTPACCES configuration.
- ✓ Any changes in FTPACCES file will get reflected in the next FTP login session.

6.2.1 Configuring the FTPACCES "NORETRIEVE" option

Syntax: noretrieve Name1 Name2 Name3… where "name" can be an absolute pathname, a single file name, or a directory name. Multiple "noretrieve" options are supported on separate records in the FTPACCES file.

Sample configurations file FTPACCES.ARPA.SYS:

```
# Purpose: support of the NORETRIEVE and CHROOT FTP/iX options.
# NORETRIEVE denies access to the listed files.
# CHROOT confines the user to the specified "root" directory.
# Syntax: NORETRIEVE name1 name2 name3 … where "name" can be an absolute pathname a simple
# file name or a directory name
# Syntax: CHROOT user.account [rooted-directory]   one entry per file record. User.Account can contain
# the "@" wildcard character.
#
# Leading and trailing spaces have no effect.
# All t the entries are case sensitive. MPE file names should be in uppercase only.

noretrieve /SYS/NET/STRACE    #  will restrict users from retrieving STRCE file from NET.SYS
noretrieve /SYS/ PUB/         #  will restrict users from retrieving all the files from PUB.SYS
noretrieve TMPTRACE           #  will restrict users from retrieving file TMPTRACE located anywhere on the system.
```

6.2.2 Specific configuration rules for NORETRIEVE option

The "FTPACCES" file-access configuration file has an exclusion list of files that are otherwise accessible with the FTP GET and MGET commands.

- ✓ The syntax of the "noretrieve" option is: ***noretrieve {file}|{/directory/file}|{/directory/}|{repeat}***
- ✓ The following three formats of the "***noretrieve***" option are supported in the FTPACCES.ARPA.SYS:
  - noretrieve /file1 /dir/file2 /dir/dir/file3 /ACCT/GROUP/FILE4
  - noretrieve file5 File6 FILE7
  - noretrieve /dir/ /dir/dir/ /ACCT/ /ACCT/GROUP/
- ✓ The entry "noretrieve {filelist}" is a space-separated list of file names specified in three formats as mentioned above. This is a list of files that can not be retrieved, either by get or mget. If the list of files that need to be made non-retrievable exceeds the record width, multiple lines starting with "noretrieve" can be used.
- ✓ All files or file sets specified in the filelist must follow the POSIX HFS notation (not the traditional MPE FILE.GROUP.ACCOUNT notation)
- ✓ Syntax:
  - Absolute path names can be specified, which will deny access to a single file. For example "noretrieve /tmp/syslog.log" prevents access to this one file.
  - A traditionally named MPE file FILE.GROUP.ACCT must be specified as /ACCT/GROUP/FILE, all in uppercase. For example "noretrieve /SYS/PUB/CATALOG" denies access to this one file.
  - If just the file name is mentioned (no directory, group, or account names present), then access will be denied to all files with that exact name, regardless of its location. Example: "noretrieve NETRC" would deny access to the file "NETRC" at /NETRC, /tmp/NETRC, /SYS/NETRC, /SYS/NET/NETRC etc. Note: the filename is case sensitive, and thus access is not restricted (by FTP) to a file named "/tmp/NetRC"

- A third format is an absolute path name terminated with a slash "/". This will deny access to all files in the absolute directory specified. For example "noretrieve /SYS/PUB/" denies access to all traditionally named MPE files in @.PUB.SYS as well as any HFS-namespace files or directories under /SYS/PUB. Likewise "noretrieve /etc/" denies access to all files contained in the directory /etc/ one below.
- Filenames embedded with invalid characters like '+', '-' etc., and wild characters like @ and* etc., are considered as invalid names and are reported in FTPLOG.ARPA.SYS as an invalid entry. Any changes to this file will get reflected in the next FTP logon session.

6.2.3 Examples for noretrieve option

Let us assume the sample configuration above(refer to section 6.2.1 for sample configuration file). This will restrict users from retrieving all the files from /SYS/NET or from a directory within /SYS/NET. Files named TMPTRACE irrespective of the directory, in which it resides, will also not be retrievable.

1. If one tries to retrieve any file that is marked irretrievable, FTP/iX will display the following message when debug mode is turned ON:

```
ftp > get /SYS/NET/STRACE
550 STRACE is marked non-retrievable.
File access denied, command restricted.  (FTPERR 78).
```

In the regular (non-debug) mode, FTP will issue the following error:

```
ftp > get /SYS/NET/STRACE
File access denied, command restricted.  (FTPERR 78).
```

6.2.4 Configuring the FTPACCES CHROOT option

As discussed in the previous section, the second restriction is the *CHROOT* option, which quarantines a user to a specified location in the FTP server's directory structure. This will limit inbound FTP client commands like Change Directory (CD), GET, PUT, MPUT, MGET and DIR to the configured current working directory (CWD) and below.

The syntax of the chroot option is:

chroot ***{user}|{@}.{account}|{@} {empty}|{/ACCT/GROUP}|{/{directory}***

Sample configuration file FTPACCES.ARPA.SYS for CHROOT:

```
# Purpose: support of the CHROOT FTP/iX option.
# CHROOT confines the user to the specified "root" directory.
# Syntax: CHROOT user.account [rooted-directory]
# One entry per file record. User.Account can contain the "@" wildcard character.
# Note: The following precedence is followed: user.acct > @.acct > user.@ > @.@.
# Leading and trailing spaces have no effect
# User name is case insensitive while the rooted directory name is case sensitive.

chroot Testmgr.@ /SYS/INSTALL
chroot @.TELESUP /TELESUP/WORK
```

6.2.5 Specific Configuration Rules for the CHROOT option

This option is available when FTPACCES.ARPA.SYS exists and contains one or more CHROOT entries. can be specified in any of the following eight formats:

a) chroot user.acct root_dir - The specified user is chroot'd to root_dir.

b) chroot user.acct - The specified user is chroot'd to the home group.

c) chroot @.acct root_dir - All the users of the specified account are chroot'd to root_dir.

d) chroot @.acct - All the users of the specified account are chroot'd to their home group.

e) chroot user.@ root_dir - FTP user of any account is chroot'd to root_dir.

f) chroot user.@ - FTP user of any account is chroot'd to the home group.

g) chroot @.@ root_dir - All users are chroot'd to root_dir.

h) chroot @.@ - All users are chroot'd to their home group.

- ✓ The user logon parameter of the chroot entry must be specified in the MPE USER.ACCT notation and wildcards except '@' are not allowed; '@' can be used only in these three formats: '@.@' or '@.acct' or 'user.@' but not for matching patterns like 'use@.acct' or 'user.@acct'.
- ✓ The root directory specification of a chroot entry must be in the POSIX HFS notation (and not in the traditional MPE FILE.GROUP.ACCOUNT syntax) as an absolute pathname from the system root ("/"). As done within the MPE/iX POSIX shell, all traditional MPE groups must be specified in HFS syntax (/ACCOUNT/GROUP), in uppercase only.
- ✓ The root directory specification is case sensitive irrespective of whether it is in the MPE name space or in the HFS name space.
- ✓ The precedence of the above mentioned eight chroot formats is: **a>b>c>d>e>f>g>h** irrespective of their order of occurrence in the FTPACCES file.
- ✓ CHROOT settings override the group name specification in ftp logon (user.acct,group) and the user's configured MPE home group.
- ✓ Invalid parameters in a chroot entry are reported in FTPLOG.ARPA.SYS as an invalid entry. This is done when FTPSRVR starts.
- ✓ Root directory specifications cannot be relative to any directory (e.g. ./dir1, ../dir2 etc), cannot include special characters like '+','-' etc. and do not support wild cards.
- ✓ Anonymous FTP behavior remains unchanged with the implementation of chroot. The root directory of an anonymous logon cannot be changed by specifying a chroot entry in FTPACCES. An anonymous FTP user will login into the directory /FTPGUEST/PUB, as before.
- ✓ Chroot does not follow soft links. This is consistent with the behavior throughout FTP/iX. Any changes to this file will get reflected in the next FTP logon session.

6.2.6 Examples for Chroot option

Consider a sample FTPACCES.ARPA.SYS file:

```
chroot Testmgr.@ /SYS/INSTALL
chroot @.TELESUP /TELESUP/WORK
chroot @.SYS
```

1. The home group of the user Testmgr.SYS is PUB.SYS and the user is directed to /SYS/INSTALL/ because of chroot option set in FTPACCES.APRA configuration file. The user is limited to the group /SYS/INSTALL and any directories under /SYS/INSTALL/.

```
ftp> user Testmgr.sys
230 User logged on
```

```
200 Type set to I.
ftp> pwd
257-"/" is the current directory.
257 "TESTMGR.SYS" is the current session.
ftp> cd ..
550 The last component of the pathname "/SYS/INSTALL/SYS" does not exist. (CIERR 93)
Could not change directory to "..". (FTPERR 48)
```

2. The users of the TELESUP account are limited to the group /TELESUP/WORK/ and any directories under /TELESUP/WORK/ irrespective of whatever is the user's home group.

If there is a directory called tmp under /TELESUP/WORK/ then "cd tmp" will be successful. But, changing to any other directory that is not under /TELESUP/WORK will result in an error:

```
Name(manager): mgrtest.telesup
230-"'/'"
230 User logged on
Remote system type is MPE/iX
200 TIMEOUT command ok.
ftp> pwd
257-"/" is the current directory.
257 "MGRTEST.TELESUP" is the current session.
ftp> cd tmp
250 CWD  file action successful.
ftp> pwd
257-"/tmp" is the current directory.
257 "MGRTEST.TELESUP" is the current session.
ftp> cd
ftp>

ftp> cd /TELESUP/PUB
550 A component of the pathname "/../TELESUP/WORK/TELESUP/PUB" does not exist. (CIERR 93)
Could not change directory to "/TELESUP/PUB". (FTPERR 48)
```

3. The users of the SYS account are limited to their home group and any directories under home group.

Here, the home group of basicusr.sys is review.sys

```
Name(testmgr): basicusr.sys
230 User logged on
Remote system type is MPE/iX
200 TIMEOUT command ok.
ftp> pwd
257-"/" is the current directory.
257 "BASICUSR.SYS" is the current session.
ftp> cd ..
550 The last component of the pathname "/SYS/REVIEW/SYS" does not exist. (CIERR935)
Could not change directory to "..". (FTPERR 48)
ftp>
```

6.3 SETPARMS.ARPA.SYS

Hitherto, this file was used to set only the system wide FTP server configuration options like password, console_logging etc. Additional configuration options have since been introduced to define the system wide behavior of FTP servers and clients. These configuration options define various permissions, log options, and the display of logon passwords in debug mode.

General configuration rules for SETPARMS

- ✓ This file can be created and edited with any supported editor.
- ✓ SETPARMS.ARPA.SYS entries are case insensitive.
- ✓ Blank spaces are ignored.
- ✓ The configuration settings can be mentioned in any order.
- ✓ Any changes to this file will get reflected in the next FTP logon session.
- ✓ Only the following keywords and the corresponding values are allowed:

POSIX = {ON/Off}
PASSWORD = {ON/Off}
CONSOLE_LOGGING = {ON/Off}
PERMISSION_RENAME = {ON/Off}
PERMISSION_DELETE = {ON/Off}
PERMISSION_OVERWRITE = {ON/Off}
LOG_COMMANDS = {ON/Off}
LOG_TRANSFERS = {ON/Off}
DEBUG_PASS = {ON/off}

All of the options below CONSOLE _LOGGING in the list above are new.

6.3.1 Configuring SETPARMS for file permission denial

Here is a sample listing of the SETPARMS configuration file:

Purpose: support of file PERMISSION DENIAL, LOG COMMANDS and TRANSFER, and DEBUG_PASS FTP/iX options.
PERMISSION DENIAL restricts FTP users from deleting, renaming, and overwriting files on the FTP server
LOG COMMANDS and TRANSFER will log the statistics of the command and transfers between the FTP/iX server
and client.
DEBUG_PASS will prevent the display of the password while logging on in debug mode.
Syntax: PERMISSION_DELETE= ON/OFF, PERMISSION_RENAME=ON/OFF, PERMISSION_OVERWRITE=ON/OFF
Syntax: LOG_COMMANDS = ON/OFF and LOG_TRANSFER = ON/OFF.
Syntax: DEBUG = ON/OFF
Leading and trailing spaces have no affect and all the entries are case insensitive.

PERMISSION_DELETE = off

PERMISSION_OVERWRITE = ON

PERMISSION_RENAME = On

6.3.2 Specific configuration rules for file permission denial

- ✓ The default setting for each of these options is "ON". This prevents FTP/iX from enforcing its own file security rules
- ✓ Users with SM capability are not restricted by the permission configuration options.
- ✓ Any changes to this file will get reflected in the next FTP logon session.

6.3.3 Examples for Permission denial

6.3.3.1 Permission RENAME

- ✓ If a user does not have SM capability and rename permission in configuration file SETPARMS.ARPA is set to OFF, any rename done by this user should fail:

```
ftp> rename strace strace01

350 File exists, ready for destination name.
550 Command access denied, permission restricted.
Rename command "RNTO strace01" failed.  (FTPERR 69)
ftp>
```

- ✓ If a user does not have SM capability and the rename permission in SETPARMS.ARPA is set to ON, all renames should be successful, provided there are no MPE/iX based security restrictions on the file being renamed:

```
ftp> rename strclean strcln
350 File exists, ready for destination name.
250 RNTO file action successful.
ftp>
```

6.3.3.2 Permission Delete:

- ✓ If a user lacks the SM capability and PERMISSION_DELETE is turned OFF, the user should **not** be able to delete a file:

```
ftp> delete strsyss
---> DELE strsyss
550 Command access denied, permission restricted.
Delete command "DELE strsyss" failed.  (FTPERR 71)
ftp>
```

- ✓ When PERMISSION_DELETE is set to ON, users should be able to delete a file, again provided there are no restrictions imposed by MPE/iX:

```
ftp> delete strerr01
550 Command access denied, permission restricted.
Delete command "DELE strerr01" failed.  (FTPERR 71)
ftp>
```

6.3.3.3 Permission Overwrite:

The following are examples of overwriting a file when PEMISSION_OVERWRITE is set to ON and OFF respectively. The FTP/iX user lacks the SM capability.

```
ftp> put FINFOHLP
200 PORT command ok.
150 File: FINFOHLP opened; data connection will be opened
200 PORT command ok.
150 File: FINFOHLP;REC=-256,1,V,ASCII;DISC=10000,8 opened; data connection will be opened
226 Transfer complete.
8207 bytes sent in 0.00 seconds (8014.65 Kbytes/sec)
ftp>


ftp> put finfohlp
```

200 PORT command ok.
150 File: finfohlp opened; data connection will be opened
200 PORT command ok.
550 Command access denied, permission restricted.
Data Transfer Request Failed. (FTPERR 13)
ftp>

6.3.4 Configuring SETPARMS for Logging commands and transfers

Please refer to section 6.3 for details on SETPARMS configuration file.

Here is a sample listing of the SETPARMS.ARPA.SYS configuration file:

Purpose: To log the FTP internal commands and data transfers between the FTP/iX server and a FTP/iX client.
Commands are logged in FTPLOG##.ARPA.SYS where, ## ranges from 00-99.
Syntax:
Log_commands = {ON/OFF}
Log_transfers = {ON/OFF}

Log_commands = On
Log_transfers = ON

6.3.5 Specific configuration rules for Log Commands and Log Transfers

- ✓ The default setting for each of these options is "OFF", thus logging of commands and file transfers is disabled.

Note:

- ✓ The FTPLOG##.ARPA.SYS file is automatically built by the FTP/iX Client or server.
- ✓ The limit on the number of records before a log file is automatically switched is 65500.
- ✓ Fields in the log are ":" delimited to support importing of the log to spreadsheet applications
- ✓ The date/time stamps are from the system on which the logging is done, and are not of the remote system.
- ✓ Any changes to this file will get reflected in the next FTP logon session.

The format of a *command* log is:

yyyy/mm/dd:hh.mm.ss:#J/#S:jobname,user.account:ip.ip.ip.ip:C/S:FTP Command

where:

- yyyy/mm/dd:hh.mm.ss = Date/Time Stamp of the command
- #J/#S = The MPE Job/Session number
- jobname,user.account = MPE logon
- ip.ip.ip.ip = IP address of the remote system
- C/S = FTP Client or FTP server that is performing the logging
- FTP Command = FTP internal client/server command.

The format of a *transfer* log is:

yyyy/mm/dd:hh.mm.ss:#J/#S:jobname,user.account:ip.ip.ip.ip:C/S:FTPCommands:I/O:I/A/B:bytes_received:seconds:Kbytes/sec

where:

- yyyy/mm/dd:hh.mm.ss = Date and Time Stamp of the transfer
- #J/#S = The MPE/iX Job/Session number
- jobname,user.account = MPE logon
- ip.ip.ip.ip = IP address of the remote system
- C/S = FTP client or FTP server that is performing the logging
- FTP Commands = Shortened FTP internal client/server file transfer command.
- I/O = Transfer request is Inbound or Outbound
- I/A/B = Transfer Mode = Image (binary), ASCII, Byte Stream
- bytes_received = Bytes received for entire file transfer
- seconds = Time in seconds for entire file transfer.
- Kbytes/sec = Kilobytes transferred per second.

Note: The transfer statistics are the approximately the same as those displayed in the FTP session following a get/mget.

6.3.6 Example

The following are some sample FTP/iX commands and transfers logged in FTPLOG## of a FTPSRVR:

2007/02/20:17.12.52:#J4:MINUSSM.SYS:aaa.bbb.ccc.ddd:S:PORT 15,70,193,18,254,253:
2007/02/20:17.12.52:#J4:MINUSSM.SYS: aaa.bbb.ccc.ddd:S:STOR FINFOHLP;REC=-256,1,V,ASCII;DISC=10000,8:
2007/02/20:17.12.52:#J4:MINUSSM.SYS: aaa.bbb.ccc.ddd:S:STOR FINFOHLP:I:I:8207:0.03:250.46:
2007/02/20:17.14.06:#J4:MINUSSM.SYS: aaa.bbb.ccc.ddd:S:QUIT:
2007/02/20:17.14.19:#J4:JINETD,MANAGER.SYS: aaa.bbb.ccc.ddd:S:USER manager.sys:
2007/02/20:17.14.19:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:SYST:
2007/02/20:17.14.19:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:SITE MPE/iX FTP Client [A0012H14]:
2007/02/20:17.14.19:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:TYPE I:
2007/02/20:17.14.19:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:SITE TIMEOUT 900:
2007/02/20:17.14.23:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:CWD /SYS/MPN:
2007/02/20:17.14.24:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:TYPE A:
2007/02/20:17.14.24:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:PORT 15,70,193,18,255,0:
2007/02/20:17.14.24:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:NLST:
2007/02/20:17.14.25:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:TYPE I:
2007/02/20:17.14.39:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:PORT 15,70,193,18,255,1:
2007/02/20:17.14.39:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:SITE FILELABEL STOR FINFOHLP:
2007/02/20:17.14.39:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:PORT 15,70,193,18,255,2:
2007/02/20:17.14.39:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:STOR FINFOHLP;REC=-256,1,V,ASCII;DISC=10000,8:
2007/02/20:17.14.40:#J4:MANAGER.SYS: aaa.bbb.ccc.ddd:S:STOR FINFOHLP:I:I:8207:0.03:242.87:

* Text marked in blue (bold and italics) indicate logs of file transfer

Note: Once logging is enabled, it is important for the system or network manager to archive and purge the FTPLOG## files in order to reclaim disk space.

6.3.7 Configuring the SETPARMS DEBUG_PASS option

Setting DEBUG_PASS = OFF in SETPARMS.ARPA.SYS prevents the display of logon passwords while in debug mode, and is the default value for this option. If this option is turned ON, logon passwords will be displayed as part of the debug information. This option is to be used with care as it can potentially cause a security violation.

6.3.8 Example

The following is a snapshot of information displayed when the DEBUG_PASS option is set to ON.

:ftp

```
File Transfer Protocol [A0012003] (C) Hewlett-Packard Co. 2000 [PASSIVE SUPPORT]
ftp> debug
Debugging on (debug=1).  (FTPINFO 24)
ftp> o Mymachine
220 HP ARPA FTP Server [A0012003] (C) Hewlett-Packard Co. 2000 [PASV SUPPORT]
Connected to Mymachine (aaa.bbb.ccc.ddd).  (FTPINFO 40)
Name(manager): manager.sys
---> USER manager.sys
331 Password required for MANAGER.SYS.  Syntax: userpass
Password:
---> PASS secret
230 User logged on
---> SYST
215 MPE/iX LF system type.
Remote system type is MPE/iX
---> SITE MPE/iX FTP Client [A0012003]
200 MPE/iX command ok.
---> TYPE I
200 Type set to I.
```

When DEBUG_PASS is turned OFF, the logon password is not displayed:

```
:ftp
ftp> debug
ftp> o system
Name(manager): MGR.TEST
---> USER MGR.TEST
331 Password required for MGR.TEST.  Syntax: userpass
Password:
---> PASS ********
```

6.4 Disallowing READ access to NETRC file

The NETRC file defines logons used by the FTP auto login feature and may contain embedded passwords. It is a security vulnerability if read/write access is allowed to this file , yet FTP needs to read the file to extract the user logon information. This enhancement allows the file security to be set such that no users have read (or write) access, but they can be permitted execute access. FTP will now open the file requesting execute access and will be able to read the contents. The NETRC file can be secured via an ACD or via the traditional file/group/account security as long as only execute permission is granted.

The syntax of NETRC entries is as follows:

- ✓ machine ["]machine_name["] login ["]user["] password ["]passwd["]
- ✓ default login ["]user["] password ["]passwd["]

Example:

```
machine "HPSYS" login "MANAGER.SYS" password "USERPASS,ACCTPASS"
default login "MGR.TELESUP" password "USERPASS,ACCTPASS"
```

6.4.1 Rules of the NETRC configuration file

- ✓ This file can be created and edited with any editor. This file should be unnumbered, of fixed width ASCII, with a record-width of no more than 72 bytes.

- ✓ The NETRC file should reside in the home group of the user logged-in. The user will have only the execute permission for the NETRC file.
- ✓ Only one "default" entry is allowed per file.
- ✓ Each of the tokens "machine", "login", "password" and "default" must match exactly, and must be in lower-case.
- ✓ Each token must be separated by any number of SPACE or TAB characters.

  Each {string} identifier can be a double quoted string. This feature would be useful when a space is embedded as part of a password, for example. Single quoted strings are not supported

  machine "HPSYS" login "MGR.TELESUP" password "USERPASS,ACCTPASS"

  and

  machine HPSYS login MGR.TELESUP password USERPASS,ACCTPASS are equivalent.
- ✓ The node name specified in the NETRC file is "CaSe SeNsItIvE" and must match the case of the node name specified in the open command.
- ✓ Any changes to this file will get reflected in the next FTP logon session.
- ✓ Unencrypted passwords stored in a file like this constitute a security risk. ACDs can be enforced on the NETRCFILE. For e.g.,

  :altsec NETRC.{home-group}.{account};access=(R,A,W,L:CR;X:AC)

6.4.2 Example

Consider the following entry in a NETRC file:

Machine "HPSYS" login "TEMPMGR.SYS" password "USERPASS,ACCTPASS"

This file should exist in the home group of TEMPMGR.SYS but will not have any permission other than the execute permission as shown below:

```
:listfile netrc,security
****************************************
FILE: NETRC.PUB.SYS

ACCOUNT ------  READ          : ANY
                WRITE         : AC
                APPEND        : AC
                LOCK          : ANY
               EXECUTE        : ANY

GROUP --------  READ          : ANY
                WRITE         : ANY
               APPEND         : ANY
                LOCK          : ANY
                EXECUTE       : ANY
                SAVE          : ANY

FILE ---------      READ          : ANY          FCODE: 0
        WRITE           : ANY          **SECURITY IS ON
        APPEND          : ANY          ACD EXISTS
        LOCK            : ANY
        EXECUTE         : ANY

FOR TEMPMGR.SYS: EXECUTE
```

Read access is denied to this file:

```
:print netrc
      ^
SECURITY VIOLATION  (FSERR 93)
The PRINT command failed. (CIERR 9080)


:ftp HPSYS
File Transfer Protocol [A0012H14] (C) Hewlett-Packard Co. 2002 [PASSIVE SUPPORT]

220 HP ARPA FTP Server [A0012H14] (C) Hewlett-Packard Co. 2000 [PASV SUPPORT]
Connected to HPSYS (AAA.BBB.CCC.DDD).  (FTPINFO 40)
331 Password required for TEMPMGR.SYS.  Syntax: userpass,acctpass
230 User logged on
Remote system type is MPE/iX
200 TIMEOUT command ok.
ftp>
```

6.5 Banner

This feature allows a FTP user to display a welcome message on successful login to a FTP/iX server. This enhancement is enabled by creating a file named FTPHELLO.ARPA.SYS, and adding the desired FTP banner text. The entire FTPHELLO file is displayed to $stdlist after a successful logon. FTPHELLO supports special substitution characters for values pertaining to the FTP connection. The Banner file supports special substitution characters that can be used to display information relevant to the FTP connection.

6.5.1 Configuring FTPHELLO

Sample FTPHELLO.ARPA.SYS:

Access to this system is restricted, unauthorized use is prohibited by law. Normal text, will be displayed as is.

```
The current working directory is %C
Connected to local host %L
Connection from remote host %R
Connection time %T
```

6.5.2 Rules of FTPHELLO configuration file

- ✓ This file can be created and edited with any supported editor. A maximum of one screen (24 lines) can be displayed at a time. This file exists on a FTP/iX server and there is no corresponding file on the client side.
- ✓ The width of each line should not exceed 72 characters i.e. FTPHELLO should have a record size of 72. Larger record size files would be truncated.
- ✓ A welcome message based on the configuration settings in FTPHELLO will be displayed on successful logon, after "User Logged on" is displayed on $STDLIST.
- ✓ The file supports substitution tokens, which help in displaying the date, working directory, remote host name and the local host name. The following are these special tokens:

%T	The server time
%C	The current working directory, i.e. the login directory
%R	The remote host name
%L	The local host name.

- ✓ If FTPHELLO.ARPA.SYS does not exist or is empty, then FTP will display the default customary one line message.
- ✓ In any line of FTPHELLO, all characters after a hash (#) are considered as comments and hence are ignored.
- ✓ Any changes to this file will get reflected in the next FTP logon session.

6.5.3 Examples

Consider a sample FTPHELLO file as mentioned above. Following will be the welcome screen after a successful logon:

```
ftp> o FTPDEST
220 HP ARPA FTP Server [A0012H15] (C) Hewlett-Packard Co. 2000 [PASV SUPPORT]
Connected to FTPDEST (aa.bb.ccc.dd).  (FTPINFO 40)
Name(manager): manager.sys
230-Access to this system is restricted, unauthorized use is prohibited by law.
230-The current working directory is '/SYS/PUB'
230-Connected to local host "FTPSRC'
230-Connection from remote host ' FTPDEST '
230-Connection time ' 5:09 AM'
230 User logged on
Remote system type is MPE/iX
ftp>
```

When FTPHELLO does not exist, only the "User logged on" message is displayed

```
ftp> o RemHost
220 HP ARPA FTP Server [A0012H15] (C) Hewlett-Packard Co. 2000 [PASV SUPPORT]
Connected to RemHost (aaa.bbb.ccc.ddd).  (FTPINFO 40)
Name(manager): manager.sys
230 User logged on
Remote system type is MPE/iX
ftp>
```

7 Encryption Alternatives

As seen in the prior sections, FTP/iX provides features that can improve security. However, an important consideration is that all data (files, user names, passwords) are still transferred as clear-text, thus increasing the risks of sniffer and middleman attacks. The following section describes several alternatives which help overcome this limitation. The methods suggested below are feasible yet simple.

7.1 A Script to transfer files (securely) using FTP/iX

HP has designed a script which will allow FTP/iX users to transfer files securely from MPE/iX to remote systems running HP-UX, Linux, MPE/iX etc. The script provides an option to encrypt files prior to the transfer. Depending on this "encrypt" option and a few other considerations, the file will be encrypted using the POSIX CRYPT utility, before it is transferred via FTP/iX. If the remote system is also an MPE/iX system, a job will be streamed, via the "site stream" FTP command, to automatically decrypt the file on the remote machine.

If the remote machine is UNIX based, the file will be encrypted on the local system and then transferred to the remote system. This file can be decrypted using the crypt command available on the remote system but, the decryption is not done automatically via FTP/iX. For this case there is a script provided on the Jazz webserver to simplify the decryption process, -- see: http://jazz.external.hp.com/src/scripts/sftpput/

which also contains the secure FTP script. The CRYPT utility is available on Jazz at http://jazz.external.hp.com/src

The following parameters and features are supported by this utility:

SFTPPUT fileset, remoteSystem, remoteUser, remoteDir, encrypt,
remoteSysType, remSysHasCrypt

where:

- ***'fileset'*** (required) a single file, a wildcarded fileset, or an indirect file (^filename) which can be supplied in MPE or POSIX syntax. The format for indirect files is one fully qualified file name per record with MPE or POSIX style names supported. Eg. F@, ./f#, /ACCT/dir/f2, ^ftplist, ^/ftp/ftplist.

- ***'remoteSystem'*** (required) the name or IP address of the system where the echo file is being transferred. The remote system can be all flavors of Unix, Windows or MPE, as long as the remote system can decrypt encrypted files via the POSIX crypt utility. The decryption is done automatically for MPE systems; whereas, non-MPE systems will need to run the crypt utility using the key which is transferred in its own file, "FileName.key". For a NETRC file to be used by FTP the machine name must match the 'remotesystem' parm name.

- ***'remoteUser'*** - (sometimes optional) the user name which FTP will use to connect to the remote system. The syntax is: "*username[:password]*" or "*username[/password]*". To suppress password prompting the username should terminate with a ":" or "/", meaning a null password, eg. 'foo:'. For MPE remote systems the username field consists of "*user.account*". If passwords are embedded in MPE user names the format is: "user/upass.acct/apass" or "*user.acct:upass,apass*". If all passwords are omitted the user may be prompted for the passwords. The expected user response for MPE passwords is: "*userpass,acctpass*".

  Note: if a comma is used then the entire name needs to be quoted so that it is treated as a single token.

  Note: this parameter is optional if a NETRC file is present since NETRC provides automatic FTP logins without the need to specify user names and passwords. However, if the remote system is MPE and the file is encrypted, a job will be streamed, named JDECRYPT, to auto-matically decrypt the FTP'd file. This job needs to be able to logon to the remote MPE system and thus may need user and/or account pass-words. This SFTPPUT script has no access to passwords contained in a NETRC file. Therefore, in order for the JDECRYPT job to logon, either :JOBSECURITY must be set to allow the desired users to logon without passwords, or the passwords must be provided to this script.

- ***'remoteDir'*** - (optional) the name of the directory (or group.account) on the remote system where the file will be sent. Syntax: "/dir", "./dir" "../dir", "~user", "group.acct", or "group". If omitted the remote user's home directory is assumed. If can be useful to specify 'remotedir' even when the logon is done via NETRC. This allows the files to be transferred to a location other than the remote user's home directory/group.

- ***'encrypt'*** - (optional) TRUE (default) means to encrypt text files. FALSE means no encryption. However, even if 'encrypt' is TRUE, only non-empty ASCII files will be encrypted.

- ***'remoteSysT'*** - (optional) "MPE/iX", default, means the remote system is known to be an MPE system. "Unix" means the remote system is known to be a Unix system. '*' means the remote system type will be determined by this script, which is extra overhead. If the file is encrypted and

the remote system is an MPE system then a job will be streamed on the remote system to decrypt the file and do some minor cleanup.

- ***'remSysHasCrypt'*** - (optional) only applies when the remote system is MPE. TRUE (default) indicates that the remote system already has the crypt utility, and thus it does not need to be FTP'd across the wire. FALSE means the remote MPE system may not have Crypt, in which case, a non-PH version is FTP'd to the remote system, executed, and then removed. If the remote system is not MPE this parameter is ignored.

If a CI variable named _SFTP_DEBUG is set to TRUE prior to executing this script, diagnostic information will be displayed to $STDLIST and the temp files and variables used by the script are kept. Otherwise, the script still displays errors and some useful information, and deletes all TEMP files and variables.

7.1.1 Examples

Few examples are illustrated below

1. Transferring a file from MPE/iX to another MPE/iX. Please note that here both source and target machines are of type MPE/iX.

```
:sftpput TMPBS, myMachine.myCompany.com, mgrtest.sys, remotedir.sys

 --- SFTPPUT ---   version A.06

Password for mgrtest.sys on myMachine.myCompany.com?
(MPE password syntax is 'user[,acct]')

** encrypting file  : /VANCE/FTPTEST/TMPBS
** transferring file: /VANCE/FTPTEST/TMPBS

==================================================
1 file transferred successfully.
1 file was encrypted.
```

2. Transferring more than one file from MPE/iX to another MPE/iX. Again here, the source and target machines are of type MPE/iX

```
:sftpput t@, myMachine.myCompany.com, mgrtest.sys, remotDir.sys

 --- SFTPPUT ---   version A.06

Password for mgrtest.sys on myMachine.myCompany.com?
 (MPE password syntax is 'user[,acct]')

** encrypting file  : /TestAcct/FTPTEST/T2
** transferring file: /TestAcct//FTPTEST/T2
** encrypting file  : /TestAcct//FTPTEST/T3
** transferring file: /TestAcct//FTPTEST/T3
. . .
** transferring file: /TestAcct//FTPTEST/TNMOBJ
** transferring file: /TestAcct//FTPTEST/TNMPRG
** transferring file: /TestAcct//FTPTEST/TPENV

==================================================
13 files transferred successfully.
4 files were encrypted.
```

3. Transferring a file from an MPE/iX machine to an HP-UX system. Please note that in this example the encryption parameter is set to FALSE.

```
:sftpput TBASFP myMachine.myCompany.com, myMachineUsrName /home/MyDirectory/, false, UNIX

--- SFTPPUT ---   version A.06

Password for myMachineUsrName on myMachine.myCompany.com

** transferring file: /TestAcct/FTPTEST/TBASFP

=====================================================
1 file transferred successfully.
0 files were encrypted.
```

4. Transferring file from an MPE/iX system to an HP-UX machine. Please note that in this example the encryption parameter is set to TRUE.

```
:sftpput TBASFP, myMachine.myCompany.com, myMachineUsrName, &
 /home/MyDirectory/ true UNIX

--- SFTPPUT ---   version A.06

Password for myMachineUsrName on myMachine.myCompany.com

** encrypting file  : /TestAcct/FTPTEST/TBASFP
** transferring file: /TestAcct/FTPTEST/TBASFP
=====================================================
1 file transferred successfully.
1 file was encrypted.
```

5. Transferring file from MPE/iX system to another MPE/ix system. Note, in the presence of NETRC file and debug mode is on. When NETRC file is in used, FTP/iX ignores user name and password, even when supplied via the 3rd parameter of this script.

Sample NETRC file:

```
:print netrc.pub
machine "remsys.hp.com" login "mgrtest.testacct" password "uPass,aPass"
default login "mgr.sys" password "u1,a1"

:sftpput catalog.pub.sys, remsys.cup.hp.com  <-- note no username

--- SFTPPUT ---   version A.06

++ Note: _SFTP_DEBUG=TRUE so all temporary files and variables
used by the script are preserved, and more verbose messages
are displayed. To see the script variables enter :showvar
_ftp_@. To see the temp files enter :listfile ./@;temp
To disable this feature :deletevar _sftp_debug, or set this
variable to FALSE.

++ Note: NETRC file in effect
++ Note: No 'remoteuser' parm supplied so "MGRTEST.TESTACCT" is assumed
```

```
to be the remote MPE logon user.acct. If this is not the same
login user id found in the NETRC file, then the MPE job to
decrypt the file(s) will logon as a different user (or the
logon could fail), and could logon to a different group, and
thus may not find the encrypted file(s). When using NETRC it's
better to supply the NETRC machine user as the 'remoteuser'
parm (parm 3) here.

** encrypting file  : /SYS/PUB/CATALOG
++ Note: key=14095-801127856, written to file ./CATALOG.key
** transferring file: /SYS/PUB/CATALOG
++ Note: transfer of /SYS/PUB/CATALOG successful

======================================================
1 file transferred successfully.
1 file was encrypted.
```

6. Transferring file from MPE/iX system to another MPE/iX system in the presence of NETRC file and not in debug mode.

Sample NETRC file is as follows:

```
:print netrc.pub
machine "remsys.hp.com" login "mgrtest.testacct" password "uPass,aPass"
default login "mgr.sys" password "u1,a1"

:setvar _sftp_debug false  -- or -- :deletevar _sftp_debug
:
:sftpput catalog.pub.sys, remsys.cup.hp.com  <-- note no username

  --- SFTPPUT ---   version A.06

  ** encrypting file  : /SYS/PUB/CATALOG
  ** transferring file: /SYS/PUB/CATALOG

======================================================
1 file transferred successfully.
1 file was encrypted.
```

7.2 Using Linux/HP-UX intermediaries

HP-UX/Linux machines support SFTP (more information can be found on SFTP at www.openssh.org or http://en. wikipedia.org/wiki/). These machines can serve as intermediaries between the source and destination MPE/iX machines and transfer data across the internet using SFTP. However, the data transfer between the MPE/iX server and the HP-UX server remains insecure. This method is simple and does not require any configuration changes; however it is still recommended to also use the new FTP/iX security features previously described

7.3 Sockisified FTP on MPE

This solution is useful if the MPE/iX FTP client is attempting to reach a FTP server behind a firewall which allows ftp socks. This configuration does not provide any encryption for logon passwords or data transfers. This requires the firewall to be configured for the socksfied FTP at port number 1080. SOCKS on MPE/iX is developed for MPE/iX 6.0 and can be downloaded from http://jazz.external.hp.com/src/ftp/index.html. This has "minimally" been tested on more current MPE/iX releases.

Note: It is an unfortunate bit of history that the sockisified version of FTP for MPEiX was called SFTP.ARPA.SYS. Just to make it clear, the socksified FTP/iX client is not Secure FTP.

7.4 OpenSSH on MPE

Open SSH was partially ported to MPE/iX by Ken Hirsh several years ago. He ported a working ssh client but was unable to port a running server. Details of the untested porting of OpenSSH are provided in the following link. http://invent3k.external.hp.com/~KEN.HIRSCH/opensshnotes.html

7.5 Hardware Solutions

7.5.1 Isolating MPE behind IPv6

IPSec is a suite of protocols for securing the Internet Protocol (IP) communication by authenticating and/or encrypting the data stream. Thus, it provides a secure channel for data/command transfers. Since, MPE/iX does not have an IPv6 implementation it, it is an option to have a MPE/iX box behind an IPv6 router. The intent is to keep the MPE/iX box in an entirely isolated network behind IPv6. But, a major concern would be that IPV6 is not even minimally implemented in the industry. More information on IPv6 can be found at: http://en.wikipedia.org/wiki/IPv6

7.5.2 HP procurve Network solutions

Modern network switches "isolate" local traffic based on the learning's of network addressing. In the not so distant past, network switches echoed all LAN traffic to each switch port and this traffic could be sniffed with common and more prevalent tools (ethereal as an example). Modern network switches (examples HP Procurve 2300, 2500, 2700 series managed plug & play switches) split the traffic seen on any switch port to the traffic that is sent on a broadcast address and the traffic that is for the specific "learned" network (IP) address.

For our concerns of FTP/iX and non-encrypted logon/passwords, commands and data over an "intranet", our customers with newer technology network switches should be less concerned as this traffic path is isolated to the source and destination systems. That is not to say that the traffic could not be sniffed at the network interface of either the source or destination system. Especially, if the source or destination system is a UNIX, Linux or Microsoft system where sniffing tools are prevalent. More information on HP procurve network is found at http://www.procurve.com

7.5.3 Encrypting router

Routers with embedded encryption facility offer a point to point, secure way of data transfer. This technology requires an encrypting router at the both the sending and receiving ends. However, this is not an open solution, but is feasible and affordable.

8 Conclusion

The standard FTP/iX features of the past have met the security requirements to a certain extent; however, these new enhancements, when enabled and used correctly, significantly improve the security of MPE/iX FTP servers. These new features can be used alone or may be combined with other security enhancements. The FTP/iX security patches are available to all HP customers on all supported releases of MPE/iX. In addition, the *Alternatives* section outlines techniques that can further boost security on FTP iX.