HP NonStop SFTP API Reference Manual

HP Part Number: 659755-003

Published: January 2014
Edition: HP NonStop SFTP API 1.4

G06.21 and subsequent G-series RVUs
H06.07 and subsequent H-series RVUs
J06.03 and subsequent J-series RVUs

Hewlett Packard Company
3000 Hanover Street
Palo Alto, CA 94304-1185

The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

Export of the information contained in this publication may require authorization from the U.S. Department of Commerce.

OSF MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THE OSF MATERIAL PROVIDED HEREIN, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. OSF shall not be liable for errors contained herein or for incidental consequential damages in connection with the furnishing, performance, or use of this material.

OSF acknowledges the following individuals and institutions for their role in its development: Kenneth C.R.C. Arnold, Gregory S. Couch, Conrad C. Huang, Ed James, Symmetric Computer Systems, Robert Elz.

Contents

Preface

Who Should Read This Guide

This manual is intended for application programmers and system administrators developing and maintaining applications which need to securely transfer files programmatically over the SFTP/SSH protocol.

Related Reading

SFTP API requires that one of the following products be installed and licensed on the system in order to work: HP NonStop SSH, comForte SecurFTP/SSH, or comForte SecurSH. Depending on which product you use, you should refer to the following document:

- HP NonStop SSH Reference Manual, or
- SSH2 Reference Manual

As SFTP transfers will be invoked using the standard FTP API (a part of the HP NonStop™ TCP/IP applications and utilities), you should also refer to the following document:

- TCP/IP Applications and Utilities User Guide, chapter 8: "FTP API External Specification"

Document History

Version 1.4

- Minor formatting changes.
- Added more detail and corrected code example in “Making your application invoke SFTP” section.

Version 1.3

- Mentioned $SYSTEM.ZSFTPAPI in sections “License Installation” and “Making your applications invoke SFTP”.
- Removed FTPappend from section “Unsupported library routines” (FTPappend supported starting with version 0092).
- Added section "Controlling Transfer Summary".

Version 1.2

- Enhanced sections “License Installation” and “Making your applications invoke SFTP”: SFTP object and SFTPAPI LICENSE file should reside in separate, customer-specific subvolume.

Version 1.1

- This is the first revised version of this documentation.

Version 1.0

- This is the first version of this documentation.

Introduction

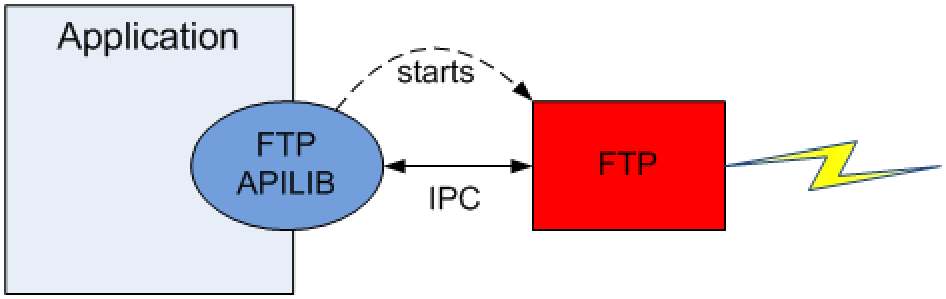
The Problem

A predominant option for securing file transfers with NonStop servers is to implement the SSH/SFTP protocol suite. As many sites used the FTP protocol for their file transfers before, this involves converting their transfer procedures from FTP to SFTP. This is usually very straight forward for interactive and batch transfers which can invoke the SFTP instead of FTP client program. However, some sites have built applications which programmatically invoke FTP file transfers using the FTP API library, which is part of the HP NonStop™ TCP/IP applications and utilities. Although it would be possible to use FTP tunneling over SSH with those applications, a complete migration to SFTP is preferred in many cases.

The SFTP API Solution

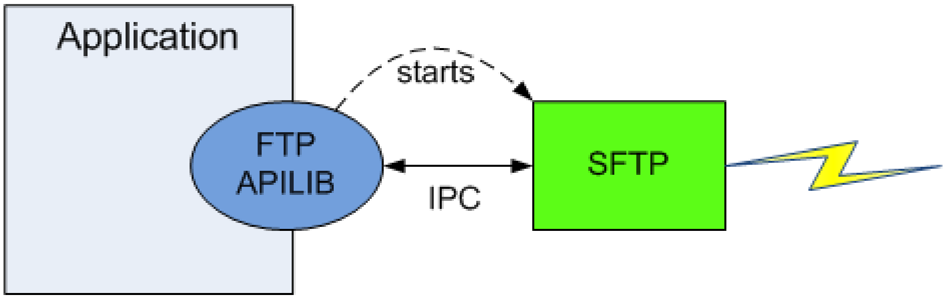
The SFTP API allows applications that previously used the FTP API to convert to SFTP in an easy manner. In many cases, the conversion can be accomplished with only a few program changes. In the ideal case, programs do not need to be changed or even re-compiled at all.

The following picture describes how applications transfer files with the FTP API:



When initiating an FTP session via the FTP APILIB, the library will start an FTP client process to handle the actual file transfers for the application. APILIB will then communicate via inter-process messages with the FTP client process, mapping the library calls to FTP commands to be processed by the FTP client.

The SFTP API solution works exactly the same way, as the following picture illustrates:



For transferring files via SFTP rather than FTP, the application still uses the same APILIB, which is part of the HP NonStop™ TCP/IP applications and utilities. However, APILIB is directed to start an SFTP rather than an FTP client. The SFTP client will support the same inter-process communication messages like FTP, mapping the programmatic commands it to the appropriate SFTP operations.

Note: SFTP API not only enables existing applications programmed to FTP API to facilitate SFTP, it also allows developing new applications to implement programmatic file transfers. Programs can use the documented FTP API to facilitate either FTP or SFTP or even both.

Installation

System requirements

To use the SFTP API, the following software must be installed on the NonStop server:

- HP NonStop SSH, comForte SecurFTP/SSH, or comForte SecurSH.

The SFTP client included with one of the packages above must have the following minimal version:

- T9999H06_17Dec2009_comForte_SFTP_0088
- For NonStop SSH on J Series and H Series, SPRs T0801AAQ or later must be installed.
- For NonStop SSH on G Series, SPRs T0801AAT or later must be installed.

Please contact Support for instructions if your current SSH installation does not include an appropriate version of SFTP.

How is SFTP API delivered?

SFTP API is not delivered as a separate package. It is rather incorporated into the SFTP program included with HP NonStop SSH, comForte SecurFTP/SSH, or comForte SecurSH. However, an appropriate license file is required to unlock the SFTP API functionality.

License Installation

To unlock the SFTP API, you will receive an SFTP API license file.

This license file contains a list of all systems that a customer wishes to license for SFTP API. It must be located within the same subvolume where the SFTP program file resides, and must be named "LICENSE". SFTP is distributed in $SYSTEM.SYSnn, but it is not recommended to place license files in $SYSTEM.SYSnn. If SFTP is copied into $SYSTEM.ZSSH, there will be a conflict if an SSH or SFTP license file named "LICENSE" already exists, unless this license file is renamed and a LICENSE parameter placed in the SSH configuration file.

Note: Even if there is no file named LICENSE on $SYSTEM.ZSSH, the SFTP API license file must not be copied to this subvolume! This can cause that the SSH2 process may not access (decrypt) the ssh database as long as the SFTP API license resides on $SYSTEM.ZSSH.

Therefore, it is strongly recommended to copy the SFTP program file into a dedicated subvolume where the license must also be placed. The application using the SFTP API should be changed to reference the SFTP object in the dedicated subvolume. Newer NonStop SSH releases copy the SFTP object to subvolume $SYSTEM.ZSFTPAPI during installation.

Note: Do not overwrite any license file in $SYSTEM.ZSSH.

Migrating FTP API Applications to SFTP API

Overview

The SFTP API has been designed to make migrating FTP API applications as easy as possible. Migration of an existing application will typically involve the following tasks:

1. Making your application invoke SFTP instead of FTP
2. Analyzing your application according to the aspects described under "Migration Considerations" below
3. Implementing any required application changes or SSH configuration uncovered by the analysis
4. Testing the application with SFTP

Making your application invoke SFTP

There are two options how you can make your application use SFTP instead of FTP.

1. Setting the FTPPGM variable to refer to the SFTP program file.

   By default it points to the FTP object. In $SYSTEM.ZTCPIP.FTPEXTH:

   ```
   /* variable pointing to which ftp program to use */
   char *FTPPGM = "$SYSTEM.SYSTEM.FTP";
   ```

   and must be set to have it point to an SFTP object.

   It is possible to modify it between opens allowing to establish both FTP and SFTP sessions.

   The SFTP program resides in $SYSTEM.SYSnn and should be copied from there to the custom location for programmatic use.

   ```
   > VOLUME $MYVOL.MYFTP
   > FUP DUP $SYSTEM.SYS00.SFTP, *, SAVEALL
   ```

   In your application code, add a line of code (or alter any existing) as follows:

   ```
   FTPPGM = "$MYVOL.MYFTP.SFTP";   /*  $MYVOL.MYFTP is the dedicated subvolume where
                                       SFTP has been copied */
   ```

   Remember that the license file for the SFTP API must then also reside within the custom location.

   Changing your application code requires recompilation to make the changes effective.

   Newer NonStop SSH releases copy the SFTP object to subvolume $SYSTEM.ZSFTPAPI during installation. This subvolume can be used as custom location for the SFTP object and no further copy is needed.

2. Replace the FTP program file that your code refers to with the SFTP program.

   This option is recommended only if your application code already refers to an FTP program file at a custom location. In this case you may replace the FTP program with the SFTP program. The commands would be somewhat as follows:

```
> VOLUME $MYVOL.MYFTP
> RENAME FTP, FTPOLD
> FUP DUP $SYSTEM.SYS00.SFTP, FTP, SAVEALL
```

   This approach does not require recompiling your application, provided that no other code changes are needed. Remember that the SFTP API license file must be placed in the custom location as well.

Migration considerations

As the SFTP protocol has significant differences from FTP, there are some important issues that you need to consider when converting an FTP API application to SFTP.

Unsupported library routines

The following FTP API routines are not supported by SFTP API:

FTP API routine	**Comment**
FTPqoute[_nw]	SFTP has no equivalent functionality
FTPproxy[_nw]	SFTP does not support proxy connections. Consequently, all FTP API routines only function on the PRIMARY connection.
FTPsetfilecode[_nw]	SFTP does not support this function. You may specify the file codes of target files as file name attributes.
FTPsetstruct[_nw]	SFTP has no equivalent functionality. However, SFTP implicitly supports transfers of structured files.
FTPsystem	SFTP has no equivalent functionality

If your application invokes one of the above routines with SFTP, error 400 (INVALID COMMAND) will be returned.

Establishing Connections

Since SFTP connects to a remote SSH daemon rather than to an FTP server, the following issues need to be considered with the FTPopen[_nw] routine.

Selecting an SSH2 process to be used by the SFTP client

The SFTP client establishes SSH sessions over an SSH2 process. By default, the SFTP client will try to open one of the SSH2 process instances named $SSH<nn>, where <nn> is a number between 00 and 15 (usually the CPU number where the process is executing). If your SSH2 process configuration does not match the default naming schema, then you must set the DEFINE =SSH2^PROCESS^NAME in your application environment before calling FTPopen[_nw], to specify the process name of the SSH2 instance to be used by SFTP. You can set the DEFINE programmatically, or you can set it in your application's startup configuration.

Port number of the remote server

If FTPopen[_nw] is called with -1 as port number (i.e. let system select the port number), the SFTP API will automatically select the standard SSH port number as default (22).

If your application has coded a specific port number, you may need to adapt your code to switch to the appropriate port where an SSH daemon is listening on.

User Authentication

FTPopen[_nw] allows you to specify a user id, an account and a password for logging in to the remote server.

The user id and password will be used for SSH authentication methods "password" and "keyboard-interactive" if accepted with the remote server. You may also pre-configure a password for the login to the target server (refer to the SSH reference manual for details), in which case the password parameter in the FTPopen[_nw] call will be ignored. SFTP can also facilitate the "publickey" authentication method, if a) the user id under which the application is running has a key which is configured in the SSH key store, and b) the key is accepted for authentication on the remote system. If public key authentication is sufficient for logging in to the remote SSH daemon, then again the password parameter will be ignored by SFTP

As SFTP does not support accounts with login, the account parameter will be ignored.

Host Authentication

Unlike FTP, remote SSH hosts are authenticated by the SFTP client using the host's public host during SSH session establishment. Depending on the value of the SSH2 STRICTHOSTKEYCHECKING option, the SFTP client may prompt the user to accept an unknown host key. This cannot be accomplished programmatically over the API. Therefore, it is required to pre-configure the public key of the target SSH hosts as KNOWNHOST for the application user id (for details of the SSHCOM ADD KNOWNHOSTS command, please refer to the SSH2 reference manual)

Log Files

Like FTP, SFTP is capable of logging its activity into a log file specified in the FTPopen[nw] command. Due to the different nature of the SFTP protocol, the content of the log file will be different.

The log file is very useful for troubleshooting migration problems, as it will show any connection problems, command errors or calls to unsupported API functions.

ASCII mode transfers

Like FTP, SFTP supports ascii mode file transfers. However, the SFTP protocol versions in use by most SFTP servers do not support transparent negotiation of the newline convention used for ASCII files. To overcome this limitation, SFTP will accept the following type characters passed by the FTPsettype call, and map it to an equivalent SFTP command:

Type character	Equivalent SFTP command
'I'	binary
'A'	ascii
'U'	ascii unix
'D'	ascii dos
'M'	ascii mac

Reply Texts

The FTP API allows to obtain FTP reply texts using the FTPreplytext[_nw] routine. While the routine is supported by SFTP, the returned information will be different from FTP. In most cases, it will be empty. In case of an error, it usually contains the error message as it would be displayed by the SFTP client in interactive mode.

If your application checks the reply text for specific patterns, you may need to change your code when migrating to SFTP.

Controlling Transfer Summary

Summary information about each file transfer gets generated, e.g.:

```
165527760 bytes transferred in 86 seconds (  1.8MB/s)
```

By default the number of bytes transferred is set to the EOF value of a file. This ensures consistency between the size of a file displayed by the ls -l command and the summary information. But the size of the actual content of a Guardian edit or structured file can differ greatly from the EOF value.

If it is of interest to see the actual number of bytes transferred in the transfer summary, then a define =SFTP^BYTES^TRANSFERRED can be set to ACTUAL:

```
ADD DEFINE =SFTP^BYTES^TRANSFERRED, CLASS MAP, FILE ACTUAL
```

The default value for this define is EOF, meaning the "bytes transferred" line contain the EOF value of a file in case the transfer was successful.