

Secure Shell Working Group  
Internet-Draft  
Expires: July 22, 2006

J. Galbraith  
VanDyke Software  
O. Saarenmaa  
F-Secure  
January 18, 2006

**SSH File Transfer Protocol**  
**draft-ietf-secsh-filexfer-extensions-00.txt**

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with [Section 6 of BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/1id-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on July 22, 2006.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

The SSH File Transfer Protocol provides a rich infrastructure for sharing information about files. This document describes several optional extensions that build on this infrastructure.

Table of Contents

- [1.](#) Introduction . . . . . [3](#)
- [2.](#) Vendor Id . . . . . [3](#)
- [3.](#) File Hashing . . . . . [4](#)
- [4.](#) Querying Available Space . . . . . [6](#)
- [5.](#) Querying User Home Directory . . . . . [7](#)
- [6.](#) Copying Remote Files . . . . . [7](#)
- [7.](#) Copying remote data . . . . . [7](#)
- [8.](#) Temporary files & directories . . . . . [8](#)
- [9.](#) Security Considerations . . . . . [9](#)
- [10.](#) References . . . . . [9](#)
  - [10.1.](#) Normative References . . . . . [9](#)
  - [10.2.](#) Informative References . . . . . [9](#)
- Authors' Addresses . . . . . [11](#)
- Intellectual Property and Copyright Statements . . . . . [12](#)



## [1.](#) Introduction

This is a collection of optional extensions to the SSH File Transfer Protocol [[I-D.ietf-secsh-filexfer](#)]. This extensions make it possible for clients to query the server for additional information which may not be widely supported, but can increase the quality of the users experience when the server can support them.

## [2.](#) Vendor Id

It is often necessary to detect the version of the server so that bugs can be worked around. This extension allows the client to do so.

```
string "vendor-id"  
string vendor-structure  
    string vendor-name      [UTF-8]  
    string product-name     [UTF-8]  
    string product-version [UTF-8]  
    uint64 product-build-number
```

vendor-name

Arbitrary name identifying the maker of the product.

product-name

Arbitrary name identifying the product.

product-version

Arbitrary string identifying the version of the product.

product-build-number

A build-number for the product, such that if a bug is fixed in build-number 'x', it can be assumed that (barring regression in the product) it is fixed in all build-numbers > 'x'.

This extension may also be sent by the client as a SSH\_FXP\_EXTENDED packet; in this case the contents of the vendor-structure become the extension-specific data:

```
byte    SSH_FXP_EXTENDED  
uint32  request-id  
string  "vendor-id"  
string  vendor-name      [UTF-8]  
string  product-name     [UTF-8]  
string  product-version [UTF-8]  
uint64  product-build-number
```



The server responds to this request with a SSH\_FXP\_STATUS message. The status SHOULD be SSH\_FX\_OK.

### 3. File Hashing

This extension allows a client to easily check if a file (or portion thereof) that it already has matches what is on the server.

```
byte    SSH_FXP_EXTENDED
uint32  request-id
string  "check-file-handle" / "check-file-name"
string  handle / name
string  hash-algorithm-list
uint64  start-offset
uint64  length
uint32  block-size
```

#### handle

For "check-file-handle", 'handle' is an open file handle returned by SSH\_FXP\_OPEN. If 'handle' is not a handle returned by SSH\_FXP\_OPEN, the server MUST return SSH\_FX\_INVALID\_HANDLE. If ACE4\_READ\_DATA was not included when the file was opened, the server MUST return STATUS\_PERMISSION\_DENIED.

If this file handle was opened in SSH\_FXF\_ACCESS\_TEXT\_MODE mode, the check must be performed on the data as it would be sent on the wire.

#### name

For "check-file-name", 'name' is the path to the file to check. If 'check-file-name' is a directory, SSH\_FX\_FILE\_IS\_A\_DIRECTORY SHOULD be returned. If 'check-file-name' refers to a SSH\_FILEXFER\_TYPE\_SYMLINK, the target should be opened. The results are undefined file types other than SSH\_FILEXFER\_TYPE\_REGULAR.

The file MUST be opened without the SSH\_FXF\_ACCESS\_TEXT\_MODE access flag (in binary mode.)

#### hash-algorithm-list

A comma separated list of hash algorithms the client is willing to accept for this operation. The server MUST pick the first hash on the list that it supports.



Currently defined algorithms are "md5", "sha1", "sha224", "sha256", "sha384", "sha512", and "crc32". Additional algorithms may be added by following the DNS extensibility naming convention outlined in [[RFC4251](#)].

MD5 is described in [[RFC1321](#)]. SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512 are described in [[FIPS-180-2](#)]. [[ISO.3309.1991](#)] describes crc32, and is the same algorithm used in [[RFC1510](#)]

#### start-offset

The starting offset of the data to include in the hash.

#### length

The length of data to include in the hash. If length is zero, all the data from start-offset to the end-of-file should be included.

#### block-size

An independent hash MUST be computed over every block in the file. The size of blocks is specified by block-size. The block-size MUST NOT be smaller than 256 bytes. If the block-size is 0, then only one hash, over the entire range, MUST be made.

The response is either a SSH\_FXP\_STATUS packet, indicating an error, or the following extended reply packet:

This extension MUST be represented in the "supported2" extension as "check-file" and both version MUST either be support or neither.

```
byte    SSH_FXP_EXTENDED_REPLY
uint32  request-id
string  hash-algo-used
byte    hash[n][block-count]
```

#### hash-algo-used

The hash algorithm that was actually used.

#### hash

The computed hashes. The hash algorithm used determines the size of n. The number of block-size chunks of data in the file determines block-count. The hashes are placed in the packet one after another, with no decoration.

Note that if the length of the range is not an even multiple of block-size, the last hash will have been computed over only the remainder of the range instead of a full block.



#### 4. Querying Available Space

The following extension provides a way to discover the available space for an arbitrary path.

```
byte    SSH_FXP_EXTENDED
uint32  request-id
string  "space-available"
string  path      [UTF-8]
```

path

'path' for which the available space should be reported. This 'path' is not required to be the mount point path, but MAY be a directory or file contained within the mount.

The reply to the request is as follows:

```
byte    SSH_FXP_EXTENDED_REPLY
uint32  request-id
uint64  bytes-on-device
uint64  unused-bytes-on-device
uint64  bytes-available-to-user
uint64  unused-bytes-available-to-user
uint32  bytes-per-allocation-unit
```

bytes-on-device

The total number of bytes on the device which stores 'path', both used and unused, or 0 if unknown.

unused-bytes-on-device

The total number of unused bytes available on the device which stores 'path', or 0 if unknown.

bytes-available-to-user

The total number of bytes, both used and unused, available to the authenticated user on the device which stores 'path', or 0 if unknown.

unused-bytes-available-to-user

The total number of unused bytes available to the authenticated user on the device which stores 'path', or 0 if unknown.

bytes-per-allocation-unit

The number of bytes in each allocation unit on the device, or in other words, the minimum number of bytes that a file allocation size can grow or shrink by. If the server does not know this information, or the file-system in use does not use allocation blocks, this value MUST be 0.



## 5. Querying User Home Directory

Many users are used to being able to type '~' as an alias for their home directory, or ~username as an alias for another user's home directory. To support this feature, a server MAY support following extension.

```
byte    SSH_FXP_EXTENDED
uint32  request-id
string  "home-directory"
string  username      [UTF-8]
```

username

Username whose home directory path is being requested. An empty string implies the current user.

The reply to the request is either a SSH\_FXP\_STATUS packet or a SSH\_FXP\_NAME packet containing the absolute real-path of the home directory.

## 6. Copying Remote Files

```
byte    SSH_FXP_EXTENDED
uint32  request-id
string  "copy-file"
string  source-file
string  destination-file
bool    overwrite-destination
```

This request copies a file from one location to another on the server. The server responds with SSH\_FXP\_STATUS.

## 7. Copying remote data

```
byte    SSH_FXP_EXTENDED
uint32  request-id
string  "copy-data"
string  read-from-handle
uint64  read-from-offset
uint64  read-data-length
string  write-to-handle
uint64  write-to-offset
```

Copy data from one handle to another on the server. The server responds with SSH\_FXP\_STATUS.



The server MUST copy the data exactly as if the client had issued a series of SSH\_FXP\_READ requests on the read-from-handle starting at read-from-offset and totaling read-data-length bytes, and issued a series of SSH\_FXP\_WRITE packets on the write-to-handle, starting at the write-from-offset, and totaling the total number of bytes read by the SSH\_FXP\_READ packets.

If one of the files is open using SSH\_FXF\_TEXT\_MODE, then operation on that handle honors all of the SSH\_FXF\_TEXT\_MODE behaviors.

The server SHOULD allow 'read-from-handle' and 'write-to-handle' to be the same handle as long as the range of data is not overlapping. This allows data to efficiently be moved within a file. The server MUST fail the request with a SSH\_FX\_INVALID\_PARAMETER if the range is overlapping and it doesn't correctly handle this case. The server is not required to detect overlapping ranges if read-from-handle and write-to-handle are different handles referring to the same file.

If 'data-length' is 0, this implies data should be read until EOF is encountered.

There are no protocol restrictions on this operation; however, the server MUST ensure that the user does not exceed quota, etc. The server is, as always, free to complete this operation out of order if it is too large to complete immediately, or to refuse a request that is too large.

## 8. Temporary files & directories

```
byte    SSH_FXP_EXTENDED
uint32  request-id
string  "get-temp-folder"
```

Retrieves the users preferred location for temporary files and folders. The server responds with SSH\_FXP\_STATUS or SSH\_FXP\_NAME containing the realpath of the preferred location.

```
byte    SSH_FXP_EXTENDED
uint32  request-id
string  "make-temp-folder"
```

Requests that the server create a folder in the users preferred location for temporary files and folders for use by the client. The server responds with SSH\_FXP\_STATUS or SSH\_FXP\_NAME containing the realpath of the new folder.

It is the clients responsibility to cleanup and remove the folder



when it is done with it. However, the server MAY remove the folder and it's contents when the client closes the connection.

## **9. Security Considerations**

The home directory extension could be used to discover whether a given user is valid; however, since users are assumed to be authenticated by the underlying protocol, this is probably not significant in most situations. If a server would not normally allow an authenticated user to query the existence of another user, the server MUST NOT allow the "home-directory" extension.

## **10. References**

### **10.1. Normative References**

- [RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", [RFC 1321](#), April 1992.
- [RFC1510] Kohl, J. and B. Neuman, "The Kerberos Network Authentication Service (V5)", [RFC 1510](#), September 1993.
- [RFC4251] Ylonen, T. and C. Lonvick, "The Secure Shell (SSH) Protocol Architecture", [RFC 4251](#), January 2006.
- [I-D.ietf-secsh-filexfer]  
Galbraith, J. and O. Saarenmaa, "SSH File Transfer Protocol", [draft-ietf-secsh-filexfer-10](#) (work in progress), October 2005.
- [FIPS-180-2]  
National Institute of Standards and Technology, "Secure Hash Standard (SHS)", Federal Information Processing Standards Publication 180-2, August 2002.
- [ISO.3309.1991]  
International Organization for Standardization, "Information Technology - Telecommunications and information exchange between systems - High-level data link control (HDLC) procedures - Frame structure", ISO Standard 3309, June 1991.

### **10.2. Informative References**

Trademark notice



"ssh" is a registered trademark in the United States and/or other countries.

Authors' Addresses

Joseph Galbraith  
VanDyke Software  
4848 Tramway Ridge Blvd  
Suite 101  
Albuquerque, NM 87111  
US

Phone: +1 505 332 5700  
Email: galb-list@vandyke.com

Oskari Saarenmaa  
F-Secure  
Tammasaarencatu 7  
Helsinki 00180  
FI

Email: oskari.saarenmaa@f-secure.com



## Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in [BCP 78](#) and [BCP 79](#).

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at [ietf-ipr@ietf.org](mailto:ietf-ipr@ietf.org).

## Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

## Copyright Statement

Copyright (C) The Internet Society (2006). This document is subject to the rights, licenses and restrictions contained in [BCP 78](#), and except as set forth therein, the authors retain all their rights.

## Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.

