Network Working Group
Request for Comments: 4250
Category: Standards Track

S. Lehtinen
SSH Communications Security Corp
C. Lonvick, Ed.
Cisco Systems, Inc.

January 2006

The Secure Shell (SSH) Protocol Assigned Numbers

Status of This Memo

Copyright Notice

Abstract

This document defines the instructions to the IANA and the initial state of the IANA assigned numbers for the Secure Shell (SSH) protocol. It is intended only for the initialization of the IANA registries referenced in the set of SSH documents.

Table of Contents

1. Introduction

This document does not define any new protocols. It is intended only to create the initial state of the IANA databases for the SSH protocol and also contains instructions for future assignments. Except for one HISTORIC algorithm generally regarded as obsolete, this document does not define any new protocols or number ranges not already defined in: [SSH-ARCH], [SSH-TRANS], [SSH-USERAUTH], [SSH-CONNECT].

2. Contributors

The major original contributors of this set of documents have been: Tatu Ylonen, Tero Kivinen, Timo J. Rinne, Sami Lehtinen (all of SSH Communications Security Corp), and Markku-Juhani O. Saarinen (University of Jyvaskyla). Darren Moffat was the original editor of this set of documents and also made very substantial contributions.

Many people contributed to the development of this document over the years. People who should be acknowledged include Mats Andersson, Ben Harris, Bill Sommerfeld, Brent McClure, Niels Moller, Damien Miller, Derek Fawcus, Frank Cusack, Heikki Nousiainen, Jakob Schlyter, Jeff Van Dyke, Jeffrey Altman, Jeffrey Hutzelman, Jon Bright, Joseph Galbraith, Ken Hornstein, Markus Friedl, Martin Forssen, Nicolas Williams, Niels Provos, Perry Metzger, Peter Gutmann, Simon Josefsson, Simon Tatham, Wei Dai, Denis Bider, der Mouse, and Tadayoshi Kohno. Listing their names here does not mean that they endorse this document, but that they have contributed to it.

3. Conventions Used in This Document

3.1. RFC 2119 Keywords

All documents related to the SSH protocols shall use the keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" to describe requirements. These keywords are to be interpreted as described in [RFC2119].

3.2. RFC 2434 Keywords

The keywords "PRIVATE USE", "HIERARCHICAL ALLOCATION", "FIRST COME FIRST SERVED", "EXPERT REVIEW", "SPECIFICATION REQUIRED", "IESG APPROVAL", "IETF CONSENSUS", and "STANDARDS ACTION" that appear in this document when used to describe namespace allocation are to be interpreted as described in [RFC2434]. These designations are repeated in this document for clarity.

PRIVATE USE - For private or local use only, with the type and purpose defined by the local site. No attempt is made to prevent multiple sites from using the same value in different (and incompatible) ways. There is no need for IANA to review such assignments and assignments are not generally useful for interoperability.

HIERARCHICAL ALLOCATION - Delegated managers can assign values provided they have been given control over that part of the name space. IANA controls the higher levels of the namespace according to one of the other policies.

FIRST COME FIRST SERVED - Anyone can obtain an assigned number, so long as they provide a point of contact and a brief description of what the value would be used for. For numbers, the exact value is generally assigned by the IANA; with names, specific names are usually requested.

EXPERT REVIEW - approval by a Designated Expert is required.

SPECIFICATION REQUIRED - Values and their meaning must be documented in an RFC or other permanent and readily available reference, in sufficient detail so that interoperability between independent implementations is possible.

IESG APPROVAL - New assignments must be approved by the IESG, but there is no requirement that the request be documented in an RFC (though the IESG has discretion to request documents or other supporting materials on a case-by-case basis).

IETF CONSENSUS - New values are assigned through the IETF consensus process. Specifically, new assignments are made via RFCs approved by the IESG. Typically, the IESG will seek input on prospective assignments from appropriate persons (e.g., a relevant Working Group if one exists).

STANDARDS ACTION - Values are assigned only for Standards Track RFCs approved by the IESG.

3.3. Protocol Fields and Values

Protocol fields and possible values to fill them are defined in this set of documents. Protocol fields will be defined in the message definitions. As an example, SSH_MSG_CHANNEL_DATA is defined as follows.

```
byte      SSH_MSG_CHANNEL_DATA
uint32    recipient channel
string    data
```

Throughout these documents, when the fields are referenced, they will appear within single quotes. When values to fill those fields are referenced, they will appear within double quotes. Using the above example, possible values for 'data' are "foo" and "bar".

4. IANA Considerations

This entire document is the IANA considerations for the SSH protocol, as defined in [SSH-ARCH], [SSH-TRANS], [SSH-USERAUTH], [SSH-CONNECT]. This section contains conventions used in naming the namespaces, the initial state of the registry, and instructions for future assignments.

4.1. Message Numbers

The Message Number is a byte value that describes the payload of a packet.

4.1.1. Conventions

Protocol packets have message numbers in the range 1 to 255. These numbers are allocated as follows:

Transport layer protocol:

1 to 19	Transport layer generic (e.g., disconnect, ignore, debug, etc.)
20 to 29	Algorithm negotiation
30 to 49	Key exchange method specific (numbers can be reused for different authentication methods)

User authentication protocol:

50 to 59	User authentication generic
60 to 79	User authentication method specific (numbers can be reused for different authentication methods)

Connection protocol:

80 to 89	Connection protocol generic
90 to 127	Channel related messages

Reserved for client protocols:

128 to 191	Reserved

Local extensions:

192 to 255	Local extensions

4.1.2. Initial Assignments

The following table identifies the initial assignments of the Message ID values.

Message ID	Value	Reference
SSH_MSG_DISCONNECT	1	[SSH-TRANS]
SSH_MSG_IGNORE	2	[SSH-TRANS]
SSH_MSG_UNIMPLEMENTED	3	[SSH-TRANS]
SSH_MSG_DEBUG	4	[SSH-TRANS]
SSH_MSG_SERVICE_REQUEST	5	[SSH-TRANS]
SSH_MSG_SERVICE_ACCEPT	6	[SSH-TRANS]
SSH_MSG_KEXINIT	20	[SSH-TRANS]
SSH_MSG_NEWKEYS	21	[SSH-TRANS]
SSH_MSG_USERAUTH_REQUEST	50	[SSH-USERAUTH]
SSH_MSG_USERAUTH_FAILURE	51	[SSH-USERAUTH]
SSH_MSG_USERAUTH_SUCCESS	52	[SSH-USERAUTH]
SSH_MSG_USERAUTH_BANNER	53	[SSH-USERAUTH]
SSH_MSG_GLOBAL_REQUEST	80	[SSH-CONNECT]
SSH_MSG_REQUEST_SUCCESS	81	[SSH-CONNECT]
SSH_MSG_REQUEST_FAILURE	82	[SSH-CONNECT]
SSH_MSG_CHANNEL_OPEN	90	[SSH-CONNECT]
SSH_MSG_CHANNEL_OPEN_CONFIRMATION	91	[SSH-CONNECT]
SSH_MSG_CHANNEL_OPEN_FAILURE	92	[SSH-CONNECT]
SSH_MSG_CHANNEL_WINDOW_ADJUST	93	[SSH-CONNECT]
SSH_MSG_CHANNEL_DATA	94	[SSH-CONNECT]
SSH_MSG_CHANNEL_EXTENDED_DATA	95	[SSH-CONNECT]
SSH_MSG_CHANNEL_EOF	96	[SSH-CONNECT]
SSH_MSG_CHANNEL_CLOSE	97	[SSH-CONNECT]
SSH_MSG_CHANNEL_REQUEST	98	[SSH-CONNECT]
SSH_MSG_CHANNEL_SUCCESS	99	[SSH-CONNECT]
SSH_MSG_CHANNEL_FAILURE	100	[SSH-CONNECT]

4.1.3. Future Assignments

Requests for assignments of new message numbers in the range of 1 to 29, 50 to 59, and 80 to 127 MUST be done through the STANDARDS ACTION method, as described in [RFC2434].

The meanings of message numbers in the range of 30 to 49 are specific to the key exchange method in use, and their meaning will be specified by the definition of that method.

The meanings of message numbers in the range of 60 to 79 are specific to the user authentication method in use, and their meaning will be specified by the definition of that method.

Requests for assignments of new message numbers in the range of 128 to 191 MUST be done through the IETF CONSENSUS method, as described in [RFC2434].

The IANA will not control the message numbers in the range of 192 through 255. This range will be left for PRIVATE USE.

4.2. Disconnection Messages Reason Codes and Descriptions

The Disconnection Message 'reason code' is a uint32 value. The associated Disconnection Message 'description' is a human-readable message that describes the disconnect reason.

4.2.1. Conventions

Protocol packets containing the SSH_MSG_DISCONNECT message MUST have Disconnection Message 'reason code' values in the range of 0x00000001 to 0xFFFFFFFF. These are described in [SSH-TRANS].

4.2.2. Initial Assignments

The following table identifies the initial assignments of the SSH_MSG_DISCONNECT 'description' and 'reason code' values.

Symbolic Name	reason code
SSH_DISCONNECT_HOST_NOT_ALLOWED_TO_CONNECT	1
SSH_DISCONNECT_PROTOCOL_ERROR	2
SSH_DISCONNECT_KEY_EXCHANGE_FAILED	3
SSH_DISCONNECT_RESERVED	4
SSH_DISCONNECT_MAC_ERROR	5
SSH_DISCONNECT_COMPRESSION_ERROR	6
SSH_DISCONNECT_SERVICE_NOT_AVAILABLE	7
SSH_DISCONNECT_PROTOCOL_VERSION_NOT_SUPPORTED	8
SSH_DISCONNECT_HOST_KEY_NOT_VERIFIABLE	9
SSH_DISCONNECT_CONNECTION_LOST	10
SSH_DISCONNECT_BY_APPLICATION	11
SSH_DISCONNECT_TOO_MANY_CONNECTIONS	12
SSH_DISCONNECT_AUTH_CANCELLED_BY_USER	13
SSH_DISCONNECT_NO_MORE_AUTH_METHODS_AVAILABLE	14
SSH_DISCONNECT_ILLEGAL_USER_NAME	15

4.2.3. Future Assignments

Disconnection Message 'reason code' values MUST be assigned sequentially. Requests for assignments of new Disconnection Message 'reason code' values, and their associated Disconnection Message 'description' text, in the range of 0x00000010 through 0xFDFFFFFF, MUST be done through the IETF CONSENSUS method, as described in [RFC2434]. The IANA will not assign Disconnection Message 'reason code' values in the range of 0xFE000000 through 0xFFFFFFFF. Disconnection Message 'reason code' values in that range are left for PRIVATE USE, as described in [RFC2434].

4.3. Channel Connection Failure Reason Codes and Descriptions

The Channel Connection Failure 'reason code' is a uint32 value. The associated Channel Connection Failure 'description' text is a human-readable message that describes the channel connection failure reason. This is described in [SSH-CONNECT].

4.3.1. Conventions

Protocol packets containing the SSH_MSG_CHANNEL_OPEN_FAILURE message MUST have Channel Connection Failure 'reason code' values in the range of 0x00000001 to 0xFFFFFFFF.

4.3.2. Initial Assignments

The initial assignments for the 'reason code' values and 'description' values are given in the table below. Note that the values for the 'reason code' are given in decimal format for readability, but they are actually uint32 values.

Symbolic Name	reason code
SSH_OPEN_ADMINISTRATIVELY_PROHIBITED	1
SSH_OPEN_CONNECT_FAILED	2
SSH_OPEN_UNKNOWN_CHANNEL_TYPE	3
SSH_OPEN_RESOURCE_SHORTAGE	4

4.3.3. Future Assignments

Channel Connection Failure 'reason code' values MUST be assigned sequentially. Requests for assignments of new Channel Connection Failure 'reason code' values, and their associated Channel Connection Failure 'description string', in the range of 0x00000005 to 0xFDFFFFFF MUST be done through the IETF CONSENSUS method, as described in [RFC2434]. The IANA will not assign Channel Connection

Failure 'reason code' values in the range of 0xFE000000 to 0xFFFFFFFF. Channel Connection Failure 'reason code' values in that range are left for PRIVATE USE, as described in [RFC2434].

4.3.4. Notes about the PRIVATE USE Range

While it is understood that the IANA will have no control over the range of 0xFE000000 to 0xFFFFFFFF, this range will be split in two parts and administered by the following conventions.

- The range of 0xFE000000 to 0xFEFFFFFF is to be used in conjunction with locally assigned channels. For example, if a channel is proposed with a 'channel type' of "example_session@example.com" but fails, then the server will respond with either a 'reason code' assigned by the IANA (as listed above and in the range of 0x00000001 to 0xFDFFFFFF), or with a locally assigned value in the range of 0xFE000000 to 0xFEFFFFFF. Naturally, if the server does not understand the proposed 'channel type', even if it is a locally defined 'channel type', then the 'reason code' MUST be 0x00000003, as described above. If the server does understand the 'channel type', but the channel still fails to open, then the server SHOULD respond with a locally assigned 'reason code' value that is consistent with the proposed local 'channel type'. It is assumed that practitioners will first attempt to use the IANA-assigned 'reason code' values and then document their locally assigned 'reason code' values.

- There are no restrictions or suggestions for the range starting with 0xFF. No interoperability is expected for anything used in this range. Essentially, it is for experimentation.

4.4. Extended Channel Data Transfer data_type_code and Data

The Extended Channel Data Transfer 'data_type_code' is a uint32 value. The associated Extended Channel Data Transfer 'data' is a human-readable message that describes the type of data allowed to be transferred in the channel.

4.4.1. Conventions

Protocol packets containing the SSH_MSG_CHANNEL_EXTENDED_DATA message MUST have Extended Channel Data Transfer 'data_type_code' values in the range of 0x00000001 to 0xFFFFFFFF. This is described in [SSH-CONNECT].

4.4.2. Initial Assignments

The initial assignments for the 'data_type_code' values and 'data' values are given in the table below. Note that the value for the 'data_type_code' is given in decimal format for readability, but the values are actually uint32 values.

Symbolic name	data_type_code
SSH_EXTENDED_DATA_STDERR	1

4.4.3. Future Assignments

Extended Channel Data Transfer 'data_type_code' values MUST be assigned sequentially. Requests for assignments of new Extended Channel Data Transfer 'data_type_code' values, and their associated Extended Channel Data Transfer 'data' strings, in the range of 0x00000002 to 0xFDFFFFFF, MUST be done through the IETF CONSENSUS method, as described in [RFC2434]. The IANA will not assign Extended Channel Data Transfer 'data_type_code' values in the range of 0xFE000000 to 0xFFFFFFFF. Extended Channel Data Transfer 'data_type_code' values in that range are left for PRIVATE USE, as described in [RFC2434].

4.5. Pseudo-Terminal Encoded Terminal Modes

SSH_MSG_CHANNEL_REQUEST messages with a "pty-req" string MUST contain 'encoded terminal modes'. The 'encoded terminal modes' value is a byte stream of opcode-argument pairs.

4.5.1. Conventions

Protocol packets containing the SSH_MSG_CHANNEL_REQUEST message with a "pty-req" string MUST contain an 'encoded terminal modes' value. The opcode values consist of a single byte and are in the range of 1 to 255. Opcodes 1 to 159 have a uint32 argument. Opcodes 160 to 255 are not yet defined.

4.5.2. Initial Assignments

The following table identifies the initial assignments of the opcode values that are used in the 'encoded terminal modes' value.

opcode	mnemonic	description
0	TTY_OP_END	Indicates end of options.
1	VINTR	Interrupt character; 255 if none. Similarly for the other characters. Not all of these characters are supported on all systems.
2	VQUIT	The quit character (sends SIGQUIT signal on POSIX systems).
3	VERASE	Erase the character to left of the cursor.
4	VKILL	Kill the current input line.
5	VEOF	End-of-file character (sends EOF from the terminal).
6	VEOL	End-of-line character in addition to carriage return and/or linefeed.
7	VEOL2	Additional end-of-line character.
8	VSTART	Continues paused output (normally control-Q).
9	VSTOP	Pauses output (normally control-S).
10	VSUSP	Suspends the current program.
11	VDSUSP	Another suspend character.
12	VREPRINT	Reprints the current input line.
13	VWERASE	Erases a word left of cursor.
14	VLNEXT	Enter the next character typed literally, even if it is a special character
15	VFLUSH	Character to flush output.
16	VSWTCH	Switch to a different shell layer.
17	VSTATUS	Prints system status line (load, command, pid, etc).
18	VDISCARD	Toggles the flushing of terminal output.
30	IGNPAR	The ignore parity flag. The parameter SHOULD be 0 if this flag is FALSE, and 1 if it is TRUE.
31	PARMRK	Mark parity and framing errors.
32	INPCK	Enable checking of parity errors.
33	ISTRIP	Strip 8th bit off characters.
34	INLCR	Map NL into CR on input.
35	IGNCR	Ignore CR on input.
36	ICRNL	Map CR to NL on input.
37	IUCLC	Translate uppercase characters to lowercase.
38	IXON	Enable output flow control.
39	IXANY	Any char will restart after stop.
40	IXOFF	Enable input flow control.
41	IMAXBEL	Ring bell on input queue full.
50	ISIG	Enable signals INTR, QUIT, [D]SUSP.
51	ICANON	Canonicalize input lines.

52	XCASE	Enable input and output of uppercase characters by preceding their lowercase equivalents with "\".
53	ECHO	Enable echoing.
54	ECHOE	Visually erase chars.
55	ECHOK	Kill character discards current line.
56	ECHONL	Echo NL even if ECHO is off.
57	NOFLSH	Don't flush after interrupt.
58	TOSTOP	Stop background jobs from output.
59	IEXTEN	Enable extensions.
60	ECHOCTL	Echo control characters as ^(Char).
61	ECHOKE	Visual erase for line kill.
62	PENDIN	Retype pending input.
70	OPOST	Enable output processing.
71	OLCUC	Convert lowercase to uppercase.
72	ONLCR	Map NL to CR-NL.
73	OCRNL	Translate carriage return to newline (output).
74	ONOCR	Translate newline to carriage return-newline (output).
75	ONLRET	Newline performs a carriage return (output).
90	CS7	7 bit mode.
91	CS8	8 bit mode.
92	PARENB	Parity enable.
93	PARODD	Odd parity, else even.
128	TTY_OP_ISPEED	Specifies the input baud rate in bits per second.
129	TTY_OP_OSPEED	Specifies the output baud rate in bits per second.

4.5.3. Future Assignments

Requests for assignments of new opcodes and their associated arguments MUST be done through the IETF CONSENSUS method, as described in [RFC2434].

4.6. Names

In the following sections, the values for the name spaces are textual. The conventions and instructions to the IANA for future assignments are given in this section. The initial assignments are given in their respective sections.

4.6.1. Conventions for Names

All names registered by the IANA in the following sections MUST be printable US-ASCII strings, and MUST NOT contain the characters at-sign ("@"), comma (","), whitespace, control characters (ASCII codes 32 or less), or the ASCII code 127 (DEL). Names are case-sensitive, and MUST NOT be longer than 64 characters.

A provision is made here for locally extensible names. The IANA will not register, and will not control, names with the at-sign in them.

Names with the at-sign in them will have the format of "name@domainname" (without the double quotes) where the part preceding the at-sign is the name. The format of the part preceding the at-sign is not specified; however, these names MUST be printable US-ASCII strings, and MUST NOT contain the comma character (","), whitespace, control characters (ASCII codes 32 or less), or the ASCII code 127 (DEL). They MUST have only a single at-sign in them. The part following the at-sign MUST be a valid, fully qualified internet domain name [RFC1034] controlled by the person or organization defining the name. Names are case-sensitive, and MUST NOT be longer than 64 characters. It is up to each domain how it manages its local namespace. It has been noted that these names resemble STD 11 [RFC0822] email addresses. This is purely coincidental and has nothing to do with STD 11 [RFC0822]. An example of a locally defined name is "ourcipher-cbc@example.com" (without the double quotes).

4.6.2. Future Assignments of Names

Requests for assignments of new names MUST be done through the IETF CONSENSUS method, as described in [RFC2434].

4.7. Service Names

The 'service name' is used to describe a protocol layer. The following table lists the initial assignments of the 'service name' values.

Service Name	Reference
ssh-userauth	[SSH-USERAUTH]
ssh-connection	[SSH-CONNECT]

4.8. Authentication Method Names

The Authentication Method Name is used to describe an authentication method for the "ssh-userauth" service [SSH-USERAUTH]. The following table identifies the initial assignments of the Authentication Method Names.

Method Name	Reference
publickey	[SSH-USERAUTH, Section 7]
password	[SSH-USERAUTH, Section 8]
hostbased	[SSH-USERAUTH, Section 9]
none	[SSH-USERAUTH, Section 5.2]

4.9. Connection Protocol Assigned Names

The following table lists the initial assignments to the Connection Protocol Type and Request names.

4.9.1. Connection Protocol Channel Types

The following table lists the initial assignments of the Connection Protocol Channel Types.

Channel type	Reference
session	[SSH-CONNECT, Section 6.1]
x11	[SSH-CONNECT, Section 6.3.2]
forwarded-tcpip	[SSH-CONNECT, Section 7.2]
direct-tcpip	[SSH-CONNECT, Section 7.2]

4.9.2. Connection Protocol Global Request Names

The following table lists the initial assignments of the Connection Protocol Global Request Names.

Request type	Reference
tcpip-forward	[SSH-CONNECT, Section 7.1]
cancel-tcpip-forward	[SSH-CONNECT, Section 7.1]

4.9.3. Connection Protocol Channel Request Names

The following table lists the initial assignments of the Connection Protocol Channel Request Names.

Request type	Reference
pty-req	[SSH-CONNECT, Section 6.2]
x11-req	[SSH-CONNECT, Section 6.3.1]
env	[SSH-CONNECT, Section 6.4]
shell	[SSH-CONNECT, Section 6.5]
exec	[SSH-CONNECT, Section 6.5]
subsystem	[SSH-CONNECT, Section 6.5]
window-change	[SSH-CONNECT, Section 6.7]
xon-xoff	[SSH-CONNECT, Section 6.8]
signal	[SSH-CONNECT, Section 6.9]
exit-status	[SSH-CONNECT, Section 6.10]
exit-signal	[SSH-CONNECT, Section 6.10]

4.9.4. Initial Assignment of Signal Names

The following table lists the initial assignments of the Signal Names.

Signal	Reference
ABRT	[SSH-CONNECT]
ALRM	[SSH-CONNECT]
FPE	[SSH-CONNECT]
HUP	[SSH-CONNECT]
ILL	[SSH-CONNECT]
INT	[SSH-CONNECT]
KILL	[SSH-CONNECT]
PIPE	[SSH-CONNECT]
QUIT	[SSH-CONNECT]
SEGV	[SSH-CONNECT]
TERM	[SSH-CONNECT]
USR1	[SSH-CONNECT]
USR2	[SSH-CONNECT]

4.9.5. Connection Protocol Subsystem Names

There are no initial assignments of the Connection Protocol Subsystem Names.

4.10. Key Exchange Method Names

The name "diffie-hellman-group1-sha1" is used for a key exchange method using an Oakley group, as defined in [RFC2409]. SSH maintains its own group identifier space, which is logically distinct from Oakley [RFC2412] and IKE; however, for one additional group, the Working Group adopted the number assigned by [RFC3526], using "diffie-hellman-group14-sha1" for the name of the second defined group. Implementations should treat these names as opaque identifiers and should not assume any relationship between the groups used by SSH and the groups defined for IKE.

The following table identifies the initial assignments of the key exchange methods.

Method name	Reference
diffie-hellman-group1-sha1	[SSH-TRANS, Section 8.1]
diffie-hellman-group14-sha1	[SSH-TRANS, Section 8.2]

4.11. Assigned Algorithm Names

4.11.1. Encryption Algorithm Names

The following table identifies the initial assignment of the Encryption Algorithm Names.

Encryption Algorithm Name	Reference
3des-cbc	[SSH-TRANS, Section 6.3]
blowfish-cbc	[SSH-TRANS, Section 6.3]
twofish256-cbc	[SSH-TRANS, Section 6.3]
twofish-cbc	[SSH-TRANS, Section 6.3]
twofish192-cbc	[SSH-TRANS, Section 6.3]
twofish128-cbc	[SSH-TRANS, Section 6.3]
aes256-cbc	[SSH-TRANS, Section 6.3]
aes192-cbc	[SSH-TRANS, Section 6.3]
aes128-cbc	[SSH-TRANS, Section 6.3]
serpent256-cbc	[SSH-TRANS, Section 6.3]
serpent192-cbc	[SSH-TRANS, Section 6.3]
serpent128-cbc	[SSH-TRANS, Section 6.3]
arcfour	[SSH-TRANS, Section 6.3]
idea-cbc	[SSH-TRANS, Section 6.3]
cast128-cbc	[SSH-TRANS, Section 6.3]
none	[SSH-TRANS, Section 6.3]
des-cbc	[FIPS-46-3] HISTORIC; See page 4 of [FIPS-46-3]

4.11.2. MAC Algorithm Names

The following table identifies the initial assignments of the MAC Algorithm Names.

MAC Algorithm Name	Reference
hmac-sha1	[SSH-TRANS, Section 6.4]
hmac-sha1-96	[SSH-TRANS, Section 6.4]
hmac-md5	[SSH-TRANS, Section 6.4]
hmac-md5-96	[SSH-TRANS, Section 6.4]
none	[SSH-TRANS, Section 6.4]

4.11.3. Public Key Algorithm Names

The following table identifies the initial assignments of the Public Key Algorithm names.

Public Key Algorithm Name	Reference
ssh-dss	[SSH-TRANS, Section 6.6]
ssh-rsa	[SSH-TRANS, Section 6.6]
pgp-sign-rsa	[SSH-TRANS, Section 6.6]
pgp-sign-dss	[SSH-TRANS, Section 6.6]

4.11.4. Compression Algorithm Names

The following table identifies the initial assignments of the Compression Algorithm names.

Compression Algorithm Name	Reference
none	[SSH-TRANS, Section 6.2]
zlib	[SSH-TRANS, Section 6.2]

5. Security Considerations

This protocol provides a secure encrypted channel over an insecure network.

Full security considerations for this protocol are provided in [SSH-ARCH].

6. References

6.1. Normative References

[SSH-ARCH] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Protocol Architecture", RFC 4251, January 2006.

[SSH-TRANS] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Transport Layer Protocol", RFC 4253, January 2006.

[SSH-USERAUTH] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Authentication Protocol", RFC 4252, January 2006.

[SSH-CONNECT] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Connection Protocol", RFC 4254, January 2006.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2409] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[RFC3526] Kivinen, T. and M. Kojo, "More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)", RFC 3526, May 2003.

6.2. Informative References

[RFC0822] Crocker, D., "Standard for the format of ARPA Internet text messages", STD 11, RFC 822, August 1982.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC2412] Orman, H., "The OAKLEY Key Determination Protocol", RFC 2412, November 1998.

[FIPS-46-3] US National Institute of Standards and Technology, "Data Encryption Standard (DES)", Federal Information Processing Standards Publication 46-3, October 1999.

Authors' Addresses

Sami Lehtinen
SSH Communications Security Corp
Valimotie 17
00380 Helsinki
Finland

EMail: sjl@ssh.com

Chris Lonvick (editor)
Cisco Systems, Inc.
12515 Research Blvd.
Austin 78759
USA

EMail: clonvick@cisco.com

Trademark Notice

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).