                 The Secure Shell (SSH) Public Key File Format

Status of This Memo

Copyright Notice

Abstract

   This document formally documents an existing public key file format
   in use for exchanging public keys between different Secure Shell
   (SSH) implementations.

   In addition, this document defines a standard textual representation
   for SSH public key fingerprints.

Table of Contents

1. Introduction

   The SSH protocol supports the use of public/private key pairs in
   order to perform authentication based on public key cryptography.
   However, in order to use public key authentication in the SSH
   protocol, public keys must first be exchanged between client and
   server.

   This document formally describes an existing public key file format
   that can be used with any of the common existing file transfer
   mechanisms in order to exchange public keys.

   The SSH protocol also uses public/private key pairs to authenticate
   the server. In this scenario, it is important to verify that the
   public key provided by the server is indeed the server's public key.
   This document describes a mechanism for creating a short text string
   that uniquely represents a particular public key, called
   fingerprinting.

2. Conventions Used in This Document

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
   document are to be interpreted as described in [RFC2119].

3. Key File Format

   In order to implement public key authentication, SSH implementations
   must share public key files between the client and the server in
   order to interoperate.

   A key file is a text file, containing a sequence of lines. Each line
   in the file MUST NOT be longer than 72 8-bit bytes excluding line
   termination characters.

3.1. Line Termination Characters

   Implementations SHOULD generate public key files using their system's
   local text file representation.

   In the event that public key files are not transferred as text files,
   implementations SHOULD be prepared to read files using any of the
   common line termination sequence, <CR>, <LF>, or <CR><LF>.

3.2. Begin and End Markers

   The first line of a conforming key file MUST be a begin marker, which
   is the literal text:

   ---- BEGIN SSH2 PUBLIC KEY ----

   The last line of a conforming key file MUST be an end marker, which
   is the literal text:

   ---- END SSH2 PUBLIC KEY ----

3.3. Key File Header

   The key file header section consists of multiple RFC822-style header
   fields. Each field is a line of the following format:

   Header-tag ':' ' ' Header-value

   The Header-tag MUST NOT be more than 64 8-bit bytes and is case-
   insensitive. The Header-value MUST NOT be more than 1024 8-bit
   bytes. Each line in the header MUST NOT be more than 72 8-bit bytes.

   A line is continued if the last character in the line is a '\'. If
   the last character of a line is a '\', then the logical contents of
   the line are formed by removing the '\' and the line termination
   characters, and appending the contents of the next line.

   The Header-tag MUST be encoded in US-ASCII. The Header-value MUST be
   encoded in UTF-8 [RFC3629].

   A line that is not a continuation line that has no ':' in it is the
   first line of the base64-encoded body. (See Section 3.4.)

   The space of header-tags is managed as described in Section 5.

   Compliant implementations MUST ignore headers with unrecognized
   header-tags. Implementations SHOULD preserve such unrecognized
   headers when manipulating the key file.

3.3.1. Subject Header

   This field is used to store the login-name that the key was generated
   under. For example:

   Subject: user

3.3.2. Comment Header

   The comment header contains a user-specified comment. The comment
   SHOULD be displayed when using the key.

   It is suggested that this field default to user@hostname for the user
   and machine used to generate the key. For example:

   Comment: user@example.com

   Currently, common practice is to quote the Header-value of the
   Comment by prefixing and suffixing it with '"' characters, and some
   existing implementations fail if these quotation marks are omitted.

   Compliant implementations MUST function correctly if the quotation
   marks are omitted.

   Implementations MAY include the quotation marks. If the first and
   last characters of the Header-value are matching quotation marks,
   implementations SHOULD remove them before using the value.

3.3.3. Private Use Headers

   Headers with header-tags beginning with "x-" are reserved for private
   use.

3.4. Public Key File Body

   The body of a public key file is the base64 encoded ([RFC2045])
   public key data as specified by [RFC4253], Section 6.6:

        string   certificate or public key format identifier
        byte[n]   key/certificate data

   As with all other lines, each line in the body MUST NOT be longer
   than 72 8-bit bytes excluding line termination characters.

3.5. Differences with RFC 1421 PEM Formats

   Implementers should take care to notice that while the format is
   superficially similar to those specified by PEM [RFC1421] and OpenPGP
   [RFC2440], it is not identical; most notably:

   o The other specifications use different BEGIN/END delimiters (five
      dashes, no space rather than four dashes and a space).

   o There is no blank line before the start of the base64-encoded
      contents.

o There is no Cyclic Redundancy Check (CRC) at the end of the
   base64-encoded block.

o Header continuation uses a backslash at the end of the continued
   line rather than whitespace at the start of the next line.

3.6. Examples

The following are some examples of public key files that are
compliant (note that the examples all wrap before 72 bytes to meet
IETF document requirements; however, they are still compliant.)

---- BEGIN SSH2 PUBLIC KEY ----
Comment: "1024-bit RSA, converted from OpenSSH by me@example.com"
x-command: /home/me/bin/lock-in-guest.sh
AAAAB3NzaC1yc2EAAAABIwAAAIEA1on8gxCGJJWSRT4uOrR13mUaUk0hRf4RzxSZ1zRb
YYFw8pfGesIFoEuVth4HKyF8k1y4mRUnYHP1XNMNMJll1JcEArC2asV8sHf6zSPVffozZ
5TT4SfsUu/iKy9lUcCfXzwre4WWZSXXcPff+EHtWshahu3WzBdnGxm5Xoi89zcE=
---- END SSH2 PUBLIC KEY ----

---- BEGIN SSH2 PUBLIC KEY ----
Comment: This is my public key for use on \
servers which I don't like.
AAAAB3NzaC1kc3MAAACBAPY8ZOHY2yFSJA6XYC9HRwNHxaehvx5wOJ0rzZdzoSOXxbET
W6ToHv8D1UJ/z+zHo9Fiko5XybZnDIaBDHtblQ+Yp7StxyltHnXF1YLfKD1G4T6JYrdH
YI14Om1eg9e4NnCRleaqoZPF3UGfZia6bXrGTQf3gJq2e7Yisk/gF+1VAAAAFQDb8D5c
vwHWTZDPfX0D2s9Rd7NBvQAAAIEAlN92+Bb7D4KLYk3IwRbXblwXdkPggA4pfdtW9vGf
J0/RHd+NjB4eo1D+0dix6tXwYGN7PKS5R/FXPNwxHPapcj9uL1Jn2AWQ2dsknf+i/FAA
vioUPkmdMc0zuWoSOEsSNhVDtX3WdvVcGcBq9cetzrtOKWOocJmJ80qadxTRHtUAAACB
AN7CY+KKv1gHpRzFwdQm7HK9bb1LAo2KwaoXnadFgeptNBQeSXG1vO+JsvphVMBJc9HS
n24VYtYtsMu74qXviYjziVucWKjjKEb11juqnF0GDlB3VVmxHLmxnAz643WK42Z7dLM5
sY29ouezv4Xz2PuMch5VGPP+CDqzCM4loWgV
---- END SSH2 PUBLIC KEY ----

---- BEGIN SSH2 PUBLIC KEY ----
Comment: DSA Public Key for use with MyIsp
AAAAB3NzaC1kc3MAAACBAPY8ZOHY2yFSJA6XYC9HRwNHxaehvx5wOJ0rzZdzoSOXxbET
W6ToHv8D1UJ/z+zHo9Fiko5XybZnDIaBDHtblQ+Yp7StxyltHnXF1YLfKD1G4T6JYrdH
YI14Om1eg9e4NnCRleaqoZPF3UGfZia6bXrGTQf3gJq2e7Yisk/gF+1VAAAAFQDb8D5c
vwHWTZDPfX0D2s9Rd7NBvQAAAIEAlN92+Bb7D4KLYk3IwRbXblwXdkPggA4pfdtW9vGf
J0/RHd+NjB4eo1D+0dix6tXwYGN7PKS5R/FXPNwxHPapcj9uL1Jn2AWQ2dsknf+i/FAA
vioUPkmdMc0zuWoSOEsSNhVDtX3WdvVcGcBq9cetzrtOKWOocJmJ80qadxTRHtUAAACB
AN7CY+KKv1gHpRzFwdQm7HK9bb1LAo2KwaoXnadFgeptNBQeSXG1vO+JsvphVMBJc9HS
n24VYtYtsMu74qXviYjziVucWKjjKEb11juqnF0GDlB3VVmxHLmxnAz643WK42Z7dLM5
sY29ouezv4Xz2PuMch5VGPP+CDqzCM4loWgV
---- END SSH2 PUBLIC KEY ----

```
---- BEGIN SSH2 PUBLIC KEY ----
Subject: me
Comment: 1024-bit rsa, created by me@example.com Mon Jan 15 \
08:31:24 2001
AAAAB3NzaC1yc2EAAAABJQAAAIEAiPWx6WM4lhHNedGfBpPJNPpZ7yKu+dnn1SJejgt4
596k6YjzGGphH2TUxwKzxcKDKKezwkpfnxPkSMkuEspGRt/aZZ9wa++Oi7Qkr8prgHc4
soW6NUlfDzpvZK2H5E7eQaSeP3SAwGmQKUFHCddNaP0L+hM7zhFNzjFvpaMgJw0=
---- END SSH2 PUBLIC KEY ----
```

4. Public Key Fingerprints

The security of the SSH protocols relies on the verification of
public host keys. Since public keys tend to be very large, it is
difficult for a human to verify an entire host key. Even with a
Public Key Infrastructure (PKI) in place, it is useful to have a
standard for exchanging short fingerprints of public keys.

This section formally describes the method of generating public key
fingerprints that is in common use in the SSH community.

The fingerprint of a public key consists of the output of the MD5
message-digest algorithm [RFC1321]. The input to the algorithm is
the public key data as specified by [RFC4253]. (This is the same
data that is base64 encoded to form the body of the public key file.)

The output of the algorithm is presented to the user as a sequence of
16 octets printed as hexadecimal with lowercase letters and separated
by colons.

For example: "c1:b1:30:29:d7:b8:de:6c:97:77:10:d7:46:41:63:87"

5. IANA Considerations

Section 3.3 defines a new namespace of "Header-tags". These are
US-ASCII strings of maximum length 64 characters and are
case-insensitive.

IANA has created and maintains a registry of these header-tags. The
registry maps each header-tag to a reference defining the header.

The initial contents of the registry are as follows:

   subject defined in Section 3.3.1

   comment defined in Section 3.3.2

Header-tags beginning with "x-" are reserved for private use, as
defined in [RFC2434].

All other allocations are to be made by IETF consensus, as defined in
[RFC2434].

6. Security Considerations

The file format described by this document provides no mechanism to
verify the integrity or otherwise detect tampering with the data
stored in such files. Given the potential of adversarial tampering
with this data, system-specific measures (e.g., Access Control Lists,
UNIX permissions, other Discretionary and/or Mandatory Access
Controls) SHOULD be used to protect these files. Also, if the
contents of these files are transferred it SHOULD be done over a
trusted channel.

The header data allowed by this file format could contain an
unlimited range of information. While in many environments the
information conveyed by this header data may be considered innocuous
public information, it may constitute a channel through which
information about a user, a key, or its use may be disclosed
intentionally or otherwise (e.g., "Comment: Mary E. Jones, 123 Main
St, Home Phone:..."). The presence and use of this header data
SHOULD be reviewed by sites that deploy this file format.

The public key fingerprint method presented here relies on the MD5
one-way hash function, which is known to have certain weaknesses
regarding its collision resistance; however, the particular use made
of MD5 here depends solely on its 2nd-preimage resistance, not on its
collision resistance.

MD5 is used here for historical reasons.

7. References

7.1. Normative References

   [RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321,
              April 1992.

   [RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail
              Extensions (MIME) Part One: Format of Internet Message
              Bodies", RFC 2045, November 1996.

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

   [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO
              10646", STD 63, RFC 3629, November 2003.

   [RFC4253] Ylonen, T. and C. Lonvick, "The Secure Shell (SSH)
              Transport Layer Protocol", RFC 4253, January 2006.

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

7.2. Informative References

   [RFC1421] Linn, J., "Privacy Enhancement for Internet Electronic
              Mail: Part I: Message Encryption and Authentication
              Procedures", RFC 1421, February 1993.

   [RFC2440] Callas, J., Donnerhacke, L., Finney, H., and R. Thayer,
              "OpenPGP Message Format", RFC 2440, November 1998.

Authors' Addresses

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

   Phone: +1 505 332 5700
   EMail: galb@vandyke.com

   Rodney Thayer
   Canola & Jones
   650 Castro Street Suite 120-205
   Mountain View CA 94041
   US

   Phone: +1 650 704 8389
   EMail: rodney@canola-jones.com

Full Copyright Statement

Intellectual Property

Acknowledgement