Internet-Draft D. Bider
Updates: 4254 (if approved) Bitvise Limited
Intended status: Standards Track December 18, 2018
Expires: May 18, 2019

Sending and Handling of Global Requests in Secure Shell (SSH)
draft-ssh-global-requests-ok-00.txt

Abstract

This memo updates RFC 4254 to clarify when global requests can be sent or received and provides a mechanism for SSH software to indicate it will accept global requests according to this requirement.

Status

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and 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/1id-abstracts.html

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

Copyright

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

1. Overview and Rationale

Secure Shell (SSH) is a common protocol for secure communication on the Internet. [RFC4254] requires both clients and servers to correctly handle messages of type SSH_MSG_GLOBAL_REQUEST received at any time. In practice, several client implementations and some servers mishandle this requirement. This discourages implementations from deploying protocol enhancements including host key synchronization and active keep-alives. Software that uses such enhancements must rely on remote version information to decide if global requests are safe to use. However, this is not accurate as to the remote party's capabilities.

This memo updates RFC 4254 to clarify when software may send and must accept global requests. An [RFC8308] extension is defined allowing SSH software to indicate it complies with this requirement.

1.1. Requirements Terminology

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].

1.2. Wire Encoding Terminology

The wire encoding types in this document - "string", "byte" and "boolean" - have meanings as described in [RFC4251].

2. Global Request Sending and Handling

The requirement in [RFC4254], which states that both a client and a server must correctly handle global requests at any time, is replaced as defined in this section.

A server MAY send a message of type SSH_MSG_GLOBAL_REQUEST at any time after it has sent the message SSH_MSG_USERAUTH_SUCCESS (defined in [RFC4252]), including immediately following that message. A server MUST NOT send SSH_MSG_GLOBAL_REQUEST before it has sent SSH_MSG_USERAUTH_SUCCESS.

A client MAY send a message of type SSH_MSG_GLOBAL_REQUEST at any time after it has received SSH_MSG_USERAUTH_SUCCESS from the server. A client MUST NOT send SSH_MSG_GLOBAL_REQUEST before it has received SSH_MSG_USERAUTH_SUCCESS.

A server MUST handle correctly - as defined in [RFC4254] - any message of type SSH_MSG_GLOBAL_REQUEST received after the server has sent SSH_MSG_USERAUTH_SUCCESS. A server MAY implement arbitrary behavior for global requests received before this. However, see Section 2.1 (Security Consideration).

A client MUST handle correctly - as defined in [RFC4254] - any message of type SSH_MSG_GLOBAL_REQUEST received after it has received SSH_MSG_USERAUTH_SUCCESS from the server. A client MAY implement arbitrary behavior for global requests received before this.

Implementations MUST correctly handle SSH_MSG_GLOBAL_REQUEST messages received during SSH key re-exchange. When implementations receive global requests during key re-exchange, they MAY defer processing them and responding until key re-exchange has completed.

2.1. Security Consideration

A server that chooses to handle SSH_MSG_GLOBAL_REQUEST before it has sent SSH_MSG_USERAUTH_SUCCESS MUST apply precautions which take into account that the client has not yet authenticated.

3. "global-requests-ok" Extension

SSH software that implements [RFC8308] MAY include the following extension when sending an SSH_MSG_EXT_INFO message:

```
string  extension-name  = "global-requests-ok"
string  extension-value = ""
```

The sender MUST send an empty extension value. A receiver that does not expect an extension value MUST ignore it. A receiver MUST tolerate and ignore non-printable binary characters in the extension value. Future specifications MAY define meanings for this value.

A receiver SHOULD assume, if the remote party includes this extension in its SSH_MSG_EXT_INFO, that the remote will handle global requests as required by this document, regardless of any heuristic knowledge the receiver may have about the remote party's software and version. The receiver SHOULD enable any functionality that relies on global requests if this extension is received.

4. Practical Uses of Global Requests

The following are some uses of the SSH_MSG_GLOBAL_REQUEST message which are prevented or made difficult by software which incorrectly disconnects when receiving a global request:

4.1. Active Keep-Alive

Network connections can terminate in ways that prevent SSH software from immediately detecting the disconnection. The TCP stack might not report the disconnection for minutes. Meanwhile resources used by the previous session, such as port numbers for TCP forwarding, may remain in use so that a reconnected client cannot resume its functions.

A common strategy to detect if the remote party is still connected is to send a global request which the remote does not have to recognize, only reply to. For example:

```
byte      SSH_MSG_GLOBAL_REQUEST
string    request-name = "keep-alive@implementation.example.com"
boolean   want-reply   = true
```

This requires the remote party to reply with SSH_MSG_REQUEST_FAILURE, which is sufficient to confirm the connection is still active.

This strategy cannot be used if the remote party might disconnect on receiving a global request.

4.2. Host Key Synchronization

A practical deficiency of SSH as standardized and widely used is that it provides no mechanism for host key rollover. A server that wishes to migrate its host key from e.g. DSA to RSA, or from RSA to Curve25519, or from 1024-bit RSA to 3072-bit RSA, has no automated way of informing clients of the intended new host key. Instead, server administrators must contact all clients - which sometimes number in thousands - where host key information must be updated manually. The common result is that servers rarely change host keys until forced.

OpenSSH supports and documents an extension ([OPENSSH]) which uses a global request named "hostkeys-00@openssh.com" to synchronize host keys. After successful authentication, the server sends this request to the client, listing all of the server's host keys. The client can respond with a further request for the server to prove possession of those host keys.

This mechanism cannot be used if the remote party might disconnect on receiving a global request.

5. IANA Considerations

IANA is requested to update the "Secure Shell (SSH) Protocol Parameters" registry established with [RFC4250], adding the following entry in the table Extension Names [IANA-EXT]:

Extension Name	Reference	Note
global-requests-ok	[this document]	Section 3

6. Security Considerations

Security considerations appear where applicable in the document.

The security considerations of [RFC4251] also apply to this document.

7. References

7.1. Normative References

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

[RFC4251] Lehtinen, S. and C. Lonvick, Ed., "The Secure Shell (SSH) Protocol Architecture", RFC 4251, January 2006.

[RFC4252] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Authentication Protocol", RFC 4252, January 2006.

[RFC4254] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Connection Protocol", RFC 4254, January 2006.

[RFC8308] Bider, D., "Extension Negotiation in the Secure Shell (SSH) Protocol", RFC 8308, March 2018.

7.2. Informative References

[RFC4250] Lehtinen, S. and C. Lonvick, Ed., "The Secure Shell (SSH) Protocol Assigned Numbers", RFC 4250, January 2006.

[OPENSSH] "OpenSSH deviations and extensions to the published SSH protocol", <https://cvsweb.openbsd.org/src/usr.bin/ssh/PROTOCOL?annotate=HEAD>.

[IANA-EXT] "Secure Shell (SSH) Protocol Parameters", <https://www.iana.org/assignments/ssh-parameters/ssh-parameters.xhtml#extension-names>.

Author's Address

Denis Bider
Bitvise Limited
4105 Lombardy Court
Colleyville, Texas 76034
United States of America

Email: ietf-ssh3@denisbider.com
URI: https://www.bitvise.com/