Secure Shell Working Group
Internet-Draft
Expires: January 16, 2006

J. Galbraith
VanDyke Software
P. Remaker
Cisco Systems, Inc
July 15, 2005

Secure Shell (SSH) Session Channel Break Extension
draft-ietf-secsh-break-04

Status of this Memo

Copyright Notice

Abstract

The Session Channel Break Extension provides a means to send a BREAK signal over a Secure Shell (SSH) terminal session.

Table of Contents

1. Introduction

The Secure Shell (SSH) session channel provides a mechanism for the client-user to interactively enter commands and receive output from a remote host while taking advantage of the SSH transport's privacy and integrity features. SSH is increasingly being used to replace Telnet for terminal access applications.

A common application of the Telnet protocol is the "Console Server" [7] whereby a Telnet Network Virtual Terminal (NVT) can be connected to a physical RS-232/V.24 asynchronous port, making the Telnet NVT appear as a locally attached terminal to that port, and making that physical port appear as a network addressable device. A number of major computer equipment vendors provide high level administrative functions through an asynchronous serial port and generally expect the attached terminal to be capable of sending a BREAK signal.

A BREAK signal is defined as the TxD signal being held in a SPACE ("0") state for a time greater than a whole character time. In practice, a BREAK signal is typically 250 to 500 ms in length.

The Telnet protocol furnishes a means to send a "BREAK" signal, which RFC0854 [1] defines as a "a signal outside the USASCII set which is currently given local meaning within many systems." [1] Console Server vendors interpret the TELNET BREAK signal as a physical BREAK signal, which can then allow access to the full range of administrative functions available on an asynchronous serial console port.

The lack of a similar facility in the SSH session channel has forced users to continue the use of Telnet for the "Console Server" function.

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 [2].

The "byte", "boolean", "uint32", and "string" data types are defined in [3].

3. The Break Request

The following channel specific request can be sent over a session channel to request that the remote host perform a BREAK operation.

```
byte      SSH_MSG_CHANNEL_REQUEST
uint32    recipient channel
string    "break"
boolean   want_reply
uint32    break-length in milliseconds
```

If the BREAK length cannot be controlled by the application receiving this request, the BREAK length parameter SHOULD be ignored and the default BREAK signal length of the chipset or underlying chipset driver SHOULD be sent. If no default exists, 500ms can be used as the BREAK length.

If the application receiving this request can control the BREAK-length, the following suggestions are made regarding BREAK duration. If a BREAK duration request of greater than 3000ms is received, it SHOULD be intepreted as a request for a 3000ms BREAK. This safeguard prevents an unreasonably long BREAK request from causing a port to become unavailable for as long as 49.7 days while executing the BREAK. Applications that require a longer BREAK may choose to ignore this suggestion. If BREAK duration request of less than 500ms is received, it SHOULD be interpreted as a 500ms BREAK since most devices will recognize a BREAK of that length. Applications that require a shorter BREAK may choose to ignore this suggestion. If the BREAK-length parameter is 0 or not present, the BREAK SHOULD be interpreted as the default BREAK signal length of the chipset or underlying chipset driver. If no default exists, 500ms can be used as the BREAK length.

If the SSH connection does not terminate on a physical serial port, the BREAK indication SHOULD be handled in a manner consistent with the general use of BREAK as an attention/interrupt signal; for instance, a service processor which requires an out-of-band facility to get the attention of a system it manages.

In a case where an SSH connection cascades to another connection, the BREAK SHOULD be passed along the cascaded connection. For example, a Telnet session from an SSH shell should carry along an SSH initiated BREAK and an SSH client initiated from a Telnet connection SHOULD pass a BREAK indication from the Telnet connection.

If the 'want_reply' boolean is set, the server MUST reply using an SSH_MSG_CHANNEL_SUCCESS or SSH_MSG_CHANNEL_FAILURE [5] message. If a BREAK of any kind was preformed, SSH_MSG_CHANNEL_SUCCESS MUST be

sent. If no BREAK was preformed, SSH_MSG_CHANNEL_FAILURE MUST be sent.

This operation SHOULD be supported by any general purpose SSH client.

4. Security Considerations

Many computer systems treat serial consoles as local and secured, and interpret a BREAK signal as an instruction to halt execution of the operating system or to enter privileged configuration modes. Because of this, extra care should be taken to ensure that SSH access to BREAK-enabled ports are limited to users with appropriate privileges to execute such functions. Alternatively, support for the BREAK facility MAY be implemented as configurable on a per-port or per-server basis.

Implementations that literally interpret the BREAK length parameter without imposing the suggested BREAK time limit may cause a denial of service to or unexpected results from attached devices receiving the very long BREAK signal.

5. IANA Considerations

IANA is requested to assign the Connection Protocol Channel Request Name "break" in accordance with [6].

6. References

6.1 Normative References

[1] Postel, J. and J. Reynolds, "Telnet Protocol Specification", STD 8, RFC 854, May 1983.

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

[3] Ylonen, T. and C. Lonvick, "SSH Protocol Architecture", draft-ietf-secsh-architecture-22 (work in progress), March 2005.

[4] Lonvick, C., "SSH Transport Layer Protocol", draft-ietf-secsh-transport-24 (work in progress), March 2005.

[5] Lonvick, C. and T. Ylonen, "SSH Connection Protocol", draft-ietf-secsh-connect-25 (work in progress), March 2005.

[6] Lehtinen, S. and C. Lonvick, "SSH Protocol Assigned Numbers", draft-ietf-secsh-assignednumbers-12 (work in progress), March 2005.

6.2 Informative References

[7] Harris, D., "Greater Scroll of Console Knowledge", March 2004, <http://www.conserver.com/consoles/>.

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

Phillip Remaker
Cisco Systems, Inc
170 West Tasman Drive
San Jose, CA 95120
US

Phone: +1 408 526 8614
Email: remaker@cisco.com

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

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