

Network Working Group
Internet-Draft
Expires: November 29, 2004

M. Wasserman
ThingMagic
T. Goddard
IceSoft
May 31, 2004

Using the NETCONF Configuration Protocol over Secure Shell (SSH)
draft-ietf-netconf-ssh-01.txt

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of [Section 10 of RFC2026](#).

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/lid-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 November 29, 2004.

Copyright Notice

Copyright (C) The Internet Society (2004). All Rights Reserved.

Abstract

This document describes a simple method for invoking and running the NETCONF configuration protocol within a Secure Shell (SSH) session as an SSH subsystem.

Internet-Draft

NETCONF over SSH

May 2004

Table of Contents

- [1. Introduction](#) [3](#)
- [2. Starting NETCONF over SSH](#) [4](#)
 - [2.1 Capabilities Exchange](#) [5](#)
- [3. Using NETCONF over SSH](#) [7](#)
- [4. Exiting the NETCONF Subsystem](#) [8](#)
- [5. Running NETCONF from an SSH Shell](#) [9](#)
 - [5.1 Starting a NETCONF Shell Session](#) [9](#)
 - [5.2 Exiting a NETCONF Shell Session](#) [9](#)
- [6. Security Considerations](#) [11](#)
- [7. IANA Considerations](#) [12](#)
- [8. Acknowledgements](#) [13](#)
- [9. References](#) [14](#)
 - [9.1 Normative References](#) [14](#)
 - [9.2 Informative References](#) [14](#)
- [Authors' Addresses](#) [14](#)
- [Intellectual Property and Copyright Statements](#) [16](#)

1. Introduction

The NETCONF protocol [[I-D.ietf-netconf-prot](#)] is an XML-based protocol used to manage the configuration of networking equipment. NETCONF is defined to be session-layer and transport independent, allowing mappings to be defined for multiple session-layer or transport protocols. This document defines how XMLCONF can be used within a Secure Shell (SSH) session, using the SSH connection protocol [[I-D.ietf-secsh-connect](#)] over the SSH transport protocol [[I-D.ietf-secsh-transport](#)]. This mapping will allow NETCONF to be executed from a secure shell session by a user or a simple script.

Throughout this document, the terms "client" and "server" are used to refer to the two ends of the SSH transport connection. The client actively opens the SSH connection, and the server passively listens for the incoming SSH connection. The terms "manager" and "agent" are used to refer to the two ends of the NETCONF protocol session. The manager issues NETCONF RPC commands, and the agent replies to those commands. When NETCONF is run over SSH, the client is always the manager, and the server is always the agent.

[2.](#) Starting NETCONF over SSH

To run NETCONF over SSH, the client will first establish an SSH transport connection using the SSH transport protocol, and the client and server will exchange keys for message integrity and encryption. The client will then invoke the "ssh-userauth" service to authenticate the user, as described in the SSH authentication protocol [[I-D.ietf-secsh-userauth](#)]. Once the user has been successfully authenticated, the client will invoke the "ssh-connection" service, also known as the SSH connection protocol.

After the ssh-connection service is established, the client will open a channel of type "session", which will result in an SSH session.

Once the SSH session has been established, the user (or script) will invoke NETCONF as an SSH subsystem called "netconf". Running NETCONF as an SSH subsystem avoids the need for the script to recognize shell prompts or skip over extraneous information, such as a system message that is printed at shell start-up.

In order to allow NETCONF traffic to be easily identified and filtered by firewalls and other network devices, NETCONF servers MUST default to providing access to the "netconf" SSH subsystem only when the SSH session is established using the IANA-assigned TCP port <TBD>. Servers SHOULD be configurable to allow access to the netconf SSH subsystem over other ports.

To the user (or script), running NETCONF as an SSH subsystem may look

similar to the following example. Although this example shows the text transmitted by both sides, the server MUST NOT echo the commands that it receives back to the client.

```
<!-- The user (or script) invokes the SSH subsystem. Depending upon
the configuration of the client and server, the passphrase prompt
may not be issued or may be replaced by a password prompt. -->
```

```
[user@client]$ ssh -s server.example.org netconf
Enter passphrase for key '/foo/.ssh/id_dsa':
```

```
<!-- The NETCONF subsystem running on the server sends a complete
XML document to the client/manager. -->
```

```
<?xml version="1.0" encoding="UTF-8"?>
<hello>
  <capabilities>
    <capability>urn:ietf:params:xml:ns:netconf:base:1.0</capability>
    <capability>urn:ietf:params:xml:ns:netconf:base:1.0#lock</capability>
  </capabilities>
</hello>
]]>]]>
```

```
<!-- The client/manager sends a complete XML document to the
server/agent. -->
```

```
<?xml version="1.0" encoding="UTF-8"?>
<hello>
  <capabilities>
    <capability>urn:ietf:params:xml:ns:netconf:base:1.0</capability>
  </capabilities>
</hello>
]]>]]>
```

While the NETCONF subsystem is active, the NETCONF manager can interact with the NETCONF agent by sending complete XML documents containing NETCONF RPC elements, and the NETCONF agent will respond by sending complete XML documents containing appropriate RPC replies.

[2.1](#) Capabilities Exchange

As indicated in the example above, the server MUST indicate its capabilities by sending an XML document containing a <hello> element as soon as the NETCONF session is established. The user (or the user's expect script) can parse this message to determine which NETCONF capabilities are supported by the server.

The client must also send an XML document containing a <hello> element to indicate the client's capabilities to the server. The document containing the <hello> element must be the first XML document that the client sends after the NETCONF session is established.

Although the example shows the server sending a <hello> message followed by the client's message, both sides will send the message as soon as the NETCONF subsystem is initialized, perhaps simultaneously.

[3.](#) Using NETCONF over SSH

A NETCONF over SSH session consists of the manager and agent exchanging complete XML documents. Once the session has been established and capabilities have been exchanged, the manager will send complete XML documents to the server containing <rpc> elements, and the agent will respond with complete XML documents containing

<rpc-reply> elements.

As the previous example illustrates, a special character sequence, `]]>]]>`, is sent after each XML document in the NETCONF exchange. This character sequence cannot legally appear in an XML document, so it can be unambiguously used to indentify the end of the current document in the event of an XML syntax or parsing error, allowing resynchronization of the NETCONF exchange.

To continue the example given above, an XMLCONF over SSH session to retrieve a set of configuration information might look like this:

```
<!-- The manager sends an XML document containing an <rpc>
element. -->
```

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="105" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get-config>
    <source><running/></source>
    <config xmlns="http://example.com/schema/1.2/config">
      <users/>
    </config>
  </get-config>
</rpc>
]]>]]>
```

```
<!-- The agent responds with an XML document containing an
<rpc-reply> element. -->
```

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="105" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <config xmlns="http://example.com/schema/1.2/config">
    <users>
      <user><name>root</name><type>superuser</type></user>
      <user><name>fred</name><type>admin</type></user>
      <user><name>barney</name><type>admin</type></user>
    </users>
  </config>
</rpc-reply>
]]>]]>
```


4. Exiting the NETCONF Subsystem

Exiting NETCONF is accomplished using the `<kill-session>` operation. When a `<kill-session>` command is issued by the manager, the agent shall respond, terminate the SSH session, and close the TCP connection.

To continue the example used in previous sections, an existing NETCONF subsystem session could be closed as follows:

```
<!-- The manager sends an XML document containing a <kill-session>
operation. Question: Where do we get the session-id? Should it be
sent in the <hello> message? Issue: The base specification forbids
the use of kill-session to kill the current session.-->

<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="106" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <kill-session>
    <session-id>0</session-id>
  </kill-session>
</rpc>
]]>]]>

<!-- The agent returns an "OK" reply. -->

<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply id="106" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>

<!-- The NETCONF subsystem exits, ending the SSH session and returning
the user (or script) to the local shell prompt. -->

[user@client]$
]]>]]>
```

[5.](#) Running NETCONF from an SSH Shell

The techniques described in this document could be used to access the NETCONF protocol over the SSH shell session, or from other shell types such as a console session or a Telnet [[RFC0854](#)] connection. However, there are serious security implications associated with allowing NETCONF access via any method that does not provide strong support for user authentication, server authentication and data privacy. See the Security Considerations section for more details.

If the server supports NETCONF invocation from an SSH shell session, the user may choose to invoke a NETCONF program from the shell command line. This would involve using SSH to establish a shell session, and entering the name of a NETCONF program (with the full path, if necessary) at the remote shell prompt.

[5.1](#) Starting a NETCONF Shell Session

To the user, the establishment of an SSH shell and the invocation of the NETCONF program may look similar to the following example:

```
<!-- The user enters an SSH shell session. -->
```

```
[user@client]$ ssh server.example.org  
user@server.example.org's password: *****
```

```
<!-- At the shell prompt, the user invokes the NETCONF program, which  
in this example is called 'netconf', but which might have different  
names on different systems. -->
```

```
[user@server]$ netconf
```

```
<!-- The NETCONF program sends an XML document to the client. -->
```

```
<?xml version="1.0" encoding="UTF-8"?>  
<hello>  
  <capabilities>  
    <capability>http://ietf.org/xmlconf/1.0/base</capability>  
    <capability>http://ietf.org/xmlconf/1.0/base#lock</capability>  
  </capabilities>  
</hello>  
]]>]]>
```

[5.2](#) Exiting a NETCONF Shell Session

When the user has run NETCONF from a shell, he will need to exit the

NETCONF program using the `<kill-session>` operation, and then exit the remote shell to return to the local shell. To continue the example used in previous sections, an existing NETCONF shell session could be closed as follows:

```
<!-- The manager sends an XML document containing an <kill-session>
operation Question: Where do we get the session-id? Should it be
sent in the <hello> message? -->
```

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="102" xmlns="http://ietf.org/xmlconf/1.0/base">
  <kill-session>
    <session-id>0</session-id>
  </kill-session>
</rpc>
]]>]]>
```

```
<!-- The agent returns an "OK" reply. -->
```

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply id="102" xmlns="http://ietf.org/netconf/1.0/base">
  <ok/>
</rpc-reply>
]]>]]>
```

```
<!-- The NETCONF program exits, returning the user to the SSH prompt.
The user then types 'exit' to exit the SSH shell and return to the
local shell. -->
```

```
[user@server]$ exit
[user@client]$
```

[6.](#) Security Considerations

NETCONF is used to access and modify configuration and state information, so the ability to access this protocol should be limited to users and systems that are authorized to view or modify the agent's configuration and state data.

The identity of the server **MUST** be verified and authenticated by the client before password-based authentication data or any configuration data is sent to the server. The identity of the client **MUST** also be verified and authenticated by the server to ensure that the incoming client request is legitimate before any configuration or state data is sent to the client. Neither side should establish a NETCONF over SSH connection with an unknown, unexpected or incorrect identity on the opposite side.

Configuration data may include sensitive information, such as usernames or security keys. So, NETCONF should only be used over communications channels that provide strong encryption for data privacy. This document defines a NETCONF over SSH mapping which provides for support of strong encryption and authentication.

If the NETCONF server provides remote shell access through insecure protocols, such as Telnet, care should be taken to prevent execution of the NETCONF program when strong user authentication or data privacy is not available. Because it may be difficult or impossible in some operating environments to determine whether a shell command was accessed over a secure protocol such as SSH or an insecure protocol such as Telnet, it may be necessary to disable insecure

shell access to the system to prevent insecure access to the NETCONF program. Alternatively, it would be possible to disable NETCONF access from the command line, only allowing NETCONF to be accessed through invocation of the SSH "netconf" subsystem.

This document requires that servers default to allowing access to the "netconf" SSH subsystem only when using a specific TCP port assigned by IANA for this purpose. This will allow NETCONF over SSH traffic to be easily identified and filtered by firewalls and other network nodes. However, it will also allow NETCONF over SSH traffic to be more easily identified by attackers.

This document also recommends that servers be configurable to allow access to the "netconf" SSH subsystem over other ports. Use of that configuration option without corresponding changes to firewall or network device configuration may unintentionally result in the ability for nodes outside of the firewall or other administrative boundary to gain access to "netconf" SSH subsystem.

[7.](#) IANA Considerations

IANA is requested to assign a TCP port number which will be the default port for NETCONF over SSH sessions as defined in this document.

8. Acknowledgements

This document was written using the xml2rfc tool described in [RFC 2629](#) [[RFC2629](#)].

Extensive input was received from the members of the NETCONF design team, including: Andy Bierman, Weijing Chen, Rob Enns, Wes Hardaker, David Harrington, Eliot Lear, Simon Leinen, Phil Shafer, Juergen Shoenwaelder and Steve Waldbusser. The following people have also reviewed this document and provided valuable input: Bill Sommerfeld.

[9.](#) References

[9.1](#) Normative References

[I-D.ietf-netconf-prot]

Enns, R., "NETCONF Configuration Protocol",
[draft-ietf-netconf-prot-02](#) (work in progress), February
2004.

[I-D.ietf-secsh-connect]

Ylonen, T., Kivinen, T., Rinne, T. and S. Lehtinen, "SSH Connection Protocol", [draft-ietf-secsh-connect-18](#) (work in progress), October 2003.

[I-D.ietf-secsh-transport]

Ylonen, T., Kivinen, T., Saarinen, M., Rinne, T. and S. Lehtinen, "SSH Transport Layer Protocol", [draft-ietf-secsh-transport-17](#) (work in progress), October 2003.

[I-D.ietf-secsh-userauth]

Ylonen, T. and C. Lonvick, "SSH Authentication Protocol", [draft-ietf-secsh-userauth-20](#) (work in progress), May 2004.

[9.2](#) Informative References

[RFC0854] Postel, J. and J. Reynolds, "Telnet Protocol Specification", STD 8, [RFC 854](#), May 1983.

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", [BCP 9](#), [RFC 2026](#), October 1996.

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", [RFC 2629](#), June 1999.

Authors' Addresses

Margaret Wasserman
ThingMagic
One Broadway, 14th Floor
Cambridge, MA 02142
USA

Phone: +1 617 758-4177
EMail: margaret@thingmagic.com
URI: <http://www.thingmagic.com>

IceSoft

E-Mail: ted@icesoft.com

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any intellectual property 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; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in [BCP-11](#). Copies of claims of rights made available for publication 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 implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

Full Copyright Statement

Copyright (C) The Internet Society (2004). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assignees.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING

TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING
BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION

Wasserman & Goddard

Expires November 29, 2004

[Page 16]

Internet-Draft

NETCONF over SSH

May 2004

HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF
MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgment

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

Wasserman & Goddard

Expires November 29, 2004

[Page 17]