        Using the NETCONF Configuration Protocol over Secure Shell (SSH)
                        draft-ietf-netconf-ssh-04.txt

Status of this Memo

    This document is an Internet-Draft and is subject to all provisions
    of Section 3 of RFC 3667. By submitting this Internet-Draft, each
    author represents that any applicable patent or other IPR claims of
    which he or she is aware have been or will be disclosed, and any of
    which he or she become aware will be disclosed, in accordance with
    RFC 3668.

    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/1id-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 October 11, 2005.

Copyright Notice

Abstract

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

Table of Contents

[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 NETCONF 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 application.

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 using the mapping defined in
this document, 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 application)
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.

A user (or application), could use the following command line to
invoke NETCONF as an SSH subsystem on the IANA-assigned port:

[user@client]$ ssh -s server.example.org -p <TBD> netconf

2.1 Capabilities Exchange

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 application) 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.

The following example shows a capability exchange. Messages sent by
the client are marked with "C:" and message sent by the server are
marked with "S:".


```
S: <?xml version="1.0" encoding="UTF-8"?>
S: <hello>
S:   <capabilities>
S:     <capability>
S:        urn:ietf:params:xml:ns:netconf:base:1.0
S:     </capability>
S:     <capability>
S:        urn:ietf:params:xml:ns:netconf:base:1.0#startup
S:     </capability>
S:   </capabilities>
S:   <session-id>4<session-id>
S: </hello>
S: ]]>]]>

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

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,
]]>]]>, MUST be sent by both the client and the server after each XML
document in the NETCONF exchange. This character sequence cannot
legally appear in an XML document, so it can be unambigiously 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 NETCONF over SSH session to
retrieve a set of configuration information might look like this:


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

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

4. **Exiting the NETCONF Subsystem**

   Exiting NETCONF is accomplished using the <close-session> operation.
   An agent will processe RPC messages from the manager in the order in
   which the are received. When the agent processes a <close-session>
   command, the agent shall respond, terminate the SSH session, and
   close the TCP connection. The agent MUST NOT process any RPC
   commands received on the current session after the <close-session>
   command.

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

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

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

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

   Configuration or state 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 rsh or 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.

6. **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.

7. **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
Schoenwaelder and Steve Waldbusser. The following people have also
reviewed this document and provided valuable input: Bill Sommerfeld.

8. **References**

8.1 Normative References

[I-D.ietf-netconf-prot]
          Enns, R., "NETCONF Configuration Protocol",
          draft-ietf-netconf-prot-05 (work in progress),
          February 2005.

[I-D.ietf-secsh-connect]
          Lonvick, C., "SSH Connection Protocol",
          draft-ietf-secsh-connect-25 (work in progress),
          March 2005.

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

[I-D.ietf-secsh-userauth]
          Lonvick, C., "SSH Authentication Protocol",
          draft-ietf-secsh-userauth-27 (work in progress),
          March 2005.

8.2 Informative References

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

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

[RFC3667] Bradner, S., "IETF Rights in Contributions", RFC 3667,
          February 2004.

[RFC3668] Bradner, S., "Intellectual Property Rights in IETF
          Technology", RFC 3668, February 2004.

Authors' Addresses

     Margaret Wasserman
     ThingMagic
     One Broadway, 5th Floor
     Cambridge, MA 02142
     USA

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

     Ted Goddard
     ICEsoft Technologies, Inc.
     Suite 300, 1717 10th St. NW
     Calgary, AB T2M 4S2
     Canada

     Phone: +1 403 663-3322
     Email: ted.goddard@icesoft.com
     URI: http://www.icesoft.com