Network Working Group                                            E. Lear
Request for Comments: 4744                                  Cisco Systems
Category: Standards Track                                     K. Crozier
                                                           December 2006


                    Using the NETCONF Protocol over
              the Blocks Extensible Exchange Protocol (BEEP)

Status of This Memo

   This document specifies an Internet standards track protocol for the
   Internet community, and requests discussion and suggestions for
   improvements.  Please refer to the current edition of the "Internet
   Official Protocol Standards" (STD 1) for the standardization state
   and status of this protocol.  Distribution of this memo is unlimited.

Copyright Notice

Abstract

   This document specifies an application protocol mapping for the
   Network Configuration Protocol (NETCONF) over the Blocks Extensible
   Exchange Protocol (BEEP).

Table of Contents

1.  Introduction

   The NETCONF protocol [1] defines a simple mechanism through which a
   network device can be managed.  NETCONF is designed to be usable over
   a variety of application protocols.  This document specifies an
   application protocol mapping for NETCONF over the Blocks Extensible
   Exchange Protocol (BEEP) [7].

   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 RFC 2119 [2].

1.1.  Why BEEP?

   Use of BEEP is natural as an application protocol for transport of
   XML.  As a peer-to-peer protocol, BEEP provides an easy way to
   implement NETCONF, no matter which side of the connection was the
   initiator.  This "bidirectionality" allows for either manager or
   agent to initiate a connection.  This is particularly important to
   support large numbers of intermittently connected devices, as well as
   those devices that must reverse the management connection in the face
   of firewalls and network address translators (NATs).

   BEEP makes use of the Simple Authentication and Security Layer (SASL)
   [3].  The SASL profile used by BEEP allows for a simple and direct
   mapping to the existing security model for command line interface
   (CLI), while Transport Layer Security (TLS) [4] provides a strong,
   well-tested encryption mechanism with either server or server and
   client-side authentication.

2.  BEEP Transport Mapping

   All NETCONF over BEEP implementations MUST implement the profile and
   functional mapping between NETCONF and BEEP as described below.

   For purposes of this document, a manager is a NETCONF client, and an
   agent is a NETCONF server.  Use of client/server language in BEEP is
   avoided because of the common notion that in networking clients
   connect to servers.

2.1.  NETCONF Session Establishment

   Managers may be either BEEP listeners or initiators.  Similarly,
   agents may be either listeners or initiators.  To establish a
   connection, the initiator connects to the listener on TCP port 831.
   Thus, the initial exchange takes place without regard to whether a
   manager or the agent is the initiator.  After the transport
   connection is established, as greetings are exchanged, they SHOULD

each announce their support for TLS and optionally SASL.  Once BEEP
greeting messages are exchanged, if TLS is to be used and available
by both parties, the listener STARTs a channel with the TLS profile.

Once TLS has been started, a new BEEP greeting message is sent by
both initiator and listener, as required by the BEEP RFC.

After all BEEP greeting messages are exchanged in order for roles to
be clear, the agent MUST advertise the NETCONF profile.  The manager
MUST NOT advertise the NETCONF profile.  If the agent side of the
communication (either initiator or listener) receives a BEEP
<greeting> element that contains the NETCONF profile, it MUST close
the connection.  Similarly, if neither side issues a NETCONF profile
it is equally an error, and the listener MUST close the connection.

At this point, if SASL is desired, the initiator starts a BEEP
channel to perform a SASL exchange to authenticate itself.  Upon
completion of authentication the channel is closed.  That is, the
channel is exclusively used to authenticate.

Examples of both TLS and SASL profiles can be found in [7].

It is anticipated that the SASL PLAIN mechanism will be heavily used
in conjunction with TLS [5].  In such cases, in accordance with RFC
2595 the PLAIN mechanism MUST NOT be advertised in the first BEEP
<greeting>, but only in the one following a successful TLS
negotiation.  This applies only if TLS and SASL PLAIN mechanisms are
both to be used.  To avoid risk of eavesdropping, the SASL PLAIN
mechanism MUST NOT be used over unencrypted channels.  More specifics
about the use of SASL and TLS are mentioned in Security
Considerations below.

Once authentication has occurred, there is no need to distinguish
between initiator and listener.  We now distinguish between manager
and agent, and it is assumed that each knows its role in the
conversation.

2.2.  Starting a Channel for NETCONF

   The manager now establishes a new channel and specifies the single
   NETCONF profile.  For example:

        (M = Manager; A = Agent)

        M: MSG 0 1 . 10 48 118
        M: Content-type: application/beep+xml
        M:
        M: <start number="1">
        M:   <profile uri="http://iana.org/beep/netconf" />
        M: </start>
        M: END
        A: RPY 0 1 . 38 87
        A: Content-Type: application/beep+xml
        A:
        A: <profile uri="http://iana.org/beep/netconf" />
        A: END

   At this point, we are ready to proceed on BEEP channel 1 with NETCONF
   operations.

   NETCONF messages are transmitted with a Content-type header set to
   "text/xml".

   Next the manager and the agent exchange NETCONF <hello> elements on
   the new channel so that each side learns the other's capabilities.
   This occurs through a MSG.  Each side will then respond positively.
   The following example is adapted from [1] Section 8.1:

        A: MSG 1 0 . 0 457
        A: Content-type: application/beep+xml
        A:
        A: <?xml version='1.0' encoding="UTF-8"?>
        A: <hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
        A:   <capabilities>
        A:     <capability>
        A:       urn:ietf:params:netconf:base:1.0
        A:     </capability>
        A:     <capability>
        A:       urn:ietf:params:netconf:capability:startup:1.0
        A:     </capability>
        A:     <capability>
        A:       http://example.net/router/2.3/core#myfeature
        A:     </capability>
        A:   </capabilities>

```
A:    <session-id>4</session-id>
A: </hello>
A: END

M: RPY 1 0 . 0 0
M: END
```

Future NETCONF capabilities may require additional BEEP channels.
When such capabilities are defined, a BEEP mapping must be defined as
well.

At this point, the NETCONF session is established, and capabilities
have been exchanged.

2.3.  NETCONF Session Usage

Nearly all NETCONF operations are executed through the <rpc> element.
To issue a remote procedure call (RPC), the manager transmits on the
operational channel a BEEP MSG containing the RPC and its arguments.
In accordance with the BEEP standard, RPC requests may be split
across multiple BEEP frames.

Once received and processed, the agent responds with BEEP RPY
messages on the same channel with the response to the RPC.  In
accordance with the BEEP standard, responses may be split across
multiple BEEP frames.

2.4.  NETCONF Session Teardown

Upon receipt of <close-session> from the manager, once the agent has
completed all RPCs, it will close BEEP channel 0.  When an agent
needs to initiate a close, it will do so by closing BEEP channel 0.
Although not required to do so, the agent should allow for a
reasonable period for a manager to release an existing lock prior to
initiating a close.  Once the agent has closed channel 0, all locks
are released, and each side follows teardown procedures as specified
in [8].  Having received a BEEP close or having sent <close-session>,
a manager MUST NOT send further requests.  If there are additional
activities due to expanded capabilities, they MUST cease in an
orderly manner and should be properly described in the capability
mapping.

2.5.  BEEP Profile for NETCONF

   Profile Identification: http://iana.org/beep/netconf

   Messages exchanged during Channel Creation: not applicable

   Messages starting one-to-one exchanges: "hello", "rpc", "rpc-reply"

   Messages in positive replies: "rpc-reply"

   Messages in negative replies: "rpc-reply"

   Messages in one-to-many exchanges: none

   Message syntax: [1]

   Message semantics: [1]

   Contact Information: See the "Author's Address" section of this memo.

3.  Security Considerations

   Configuration information is by its very nature sensitive.  Its
   transmission in the clear and without integrity checking leaves
   devices open to classic so-called "person-in-the-middle" attacks.
   Configuration information often times contains passwords, user names,
   service descriptions, and topological information, all of which are
   sensitive.  A NETCONF application protocol, therefore, must minimally
   support options for both confidentiality and authentication.

   The BEEP mapping described in this document addresses both
   confidentiality and authentication in a flexible manner through the
   use of TLS and SASL profiles.  Confidentiality is provided via the
   TLS profile and is used as discussed above.  In addition, the server
   certificate shall serve as the server's authentication to the client.
   The client MUST be prepared to recognize and validate a server
   certificate and SHOULD by default reject invalid certificates.

   In order to validate a certificate, the client must be able to access
   a trust anchor.  While such validation methods are beyond the scope
   of this document, they will depend on the type of device and
   circumstance.  Both the implementor and the administrator are
   cautioned to be aware of any circular dependencies that various
   methods may introduce.  For instance, Online Certificate Status
   Protocol (OCSP) servers may not be available in a network cold-start
   scenario and would be ill-advised for core routers to depend on to
   receive configuration at boot.

For client-side authentication, there are several options.  The
client MAY provide a certificate during the initiation phase of TLS,
in which case the subject of that certificate shall be considered
principle for authentication purposes.  Once again, server
implementors should be aware of any interdependencies that could be
created through protocols used to validate trust anchors.

TLS endpoints may be authorized based on subject name or certificate
authority (CA), depending on circumstances.  For instance, it would
be unwise for a core Internet router to allow a netconf agent
connection simply based on a valid certificate signed by a common CA,
but not unreasonable to allow a connection from an agent with a
particular distinguished name.  On the other hand, it might be
desirable for enterprises to trust certificates signed by CAs of
their network operations team.

In the case where the client has not authenticated through TLS, the
server SHOULD advertise one or more SASL profiles, from which the
client will choose.  In the singular case where TLS is established,
the minimum profile MAY be PLAIN.  Otherwise, implementations MUST
support the DIGEST-MD5 profile as described in [6], and they MAY
support other profiles such as the One-Time Password (OTP) mechanism
[10].

Different environments may well allow different rights prior to and
then after authentication.  An authorization model is not specified
in this document.  When an operation is not properly authorized, then
a simple rpc-error containing "permission denied" is sufficient.
Note that authorization information may be exchanged in the form of
configuration information, which is all the more reason to ensure the
security of the connection.

4.  IANA Considerations

   IANA assigned TCP port (831) for NETCONF over BEEP.

5.  Acknowledgments

   This work is the product of the NETCONF IETF working group, and many
   people have contributed to the NETCONF discussion.  Most notably, Rob
   Ens, Phil Schafer, Andy Bierman, Wes Hardiger, Ted Goddard, and
   Margaret Wasserman all contributed in some fashion to this work,
   which was originally to be found in the NETCONF base protocol
   specification.  Thanks also to Weijing Chen, Keith Allen, Juergen
   Schoenwaelder, Marshall Rose, and Eamon O'Tuathail for their very
   constructive participation.  The authors would also like to thank
   Elwyn Davies for his constructive review.

6.  References

6.1.  Normative References

   [1]  Enns, R., Ed., "NETCONF Configuration Protocol", RFC 4741,
        December 2006.

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

   [3]  Melnikov, A. and K. Zeilenga, "Simple Authentication and
        Security Layer (SASL)", RFC 4422, June 2006.

   [4]  Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS)
        Protocol Version 1.1", RFC 4346, April 2006.

   [5]  Newman, C., "Using TLS with IMAP, POP3 and ACAP", RFC 2595,
        June 1999.

   [6]  Leach, P. and C. Newman, "Using Digest Authentication as a SASL
        Mechanism", RFC 2831, May 2000.

   [7]  Rose, M., "The Blocks Extensible Exchange Protocol Core",
        RFC 3080, March 2001.

   [8]  Rose, M., "Mapping the BEEP Core onto TCP", RFC 3081,
        March 2001.

6.2.  Informative References

   [9]  Sperberg-McQueen, C., Paoli, J., Maler, E., and T. Bray,
        "Extensible Markup Language (XML) 1.0 (Second Edition)", World
        Wide Web Consortium, First Edition,
        http://www.w3.org/TR/2000/REC-xml-20001006, October 2000.

   [10] Newman, C., "The One-Time-Password SASL Mechanism", RFC 2444,
        October 1998.

Authors' Addresses

   Eliot Lear
   Cisco Systems
   Glatt-com
   CH-8301 Glattzentrum, Zurich
   CH

   EMail: lear@cisco.com


   Ken Crozier

   EMail: ken.crozier@gmail.com