A SOAP Binding for NETCONF
draft-goddard-netconfsoap-00.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/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 December 18, 2003.

Copyright Notice

Copyright (C) The Internet Society (2003). All Rights Reserved.

Abstract

While the device management protocol NETCONF is generally well served by BEEP, there are environments where additional transports are desirable. The binding to SOAP described here may find application where SOAP-based tools and implementations are prevalent or where the network configuration favors HTTP. When used with multiple HTTP connections, SOAP over HTTP is sufficient for all NETCONF features except those involving asynchronous notification.
# Table of Contents

1. Introduction ................................................. 3
2. SOAP Background for NETCONF ............................... 4
  2.1 Use and Storage of WSDL and XSD ......................... 4
  2.2 SOAP over HTTP ........................................... 5
  2.3 HTTP Drawbacks ........................................... 5
  2.4 Important HTTP 1.1 Features ............................... 6
3. A SOAP Web Service for NETCONF ............................. 7
  3.1 Fundamental Use Case .................................... 7
  3.2 Mapping BEEP Channels to HTTP Connections ............... 7
  3.2.1 Asynchronous Functionality ........................... 7
  3.3 NETCONF Sessions ......................................... 8
  3.4 Capabilities Exchange .................................... 9
  3.5 A NETCONF/SOAP example .................................. 9
  3.6 Managing Multiple Devices ................................. 10
4. Security Considerations ....................................... 11
  4.1 Integrity, Privacy, and Authentication .................... 11
  4.2 Vulnerabilities ........................................... 11
  4.3 Environmental Specifics ................................... 11
  4.3.1 Normative References ................................ 12
  4.3.2 Informative References ............................... 14
  4.3.3 Author’s Address ..................................... 14
5. WSDL Definitions ............................................. 15
  5.1 NETCONF SOAP Binding .................................... 15
  5.2 Sample Service Definition ................................. 16
6. Intellectual Property and Copyright Statements ............. 17
1. Introduction

Given the use of XML [2] and the remote procedure call characteristics, it is natural to consider a binding of the NETCONF [1] operations to a SOAP [3] transport. The purpose here is to provide a concrete description of such a binding so that the technical issues can be discussed.

Note that a SOAP binding for NETCONF is not necessarily intended only for managing individual devices. For instance, a server providing a SOAP interface can act as a proxy for multiple devices, possibly connecting to those devices over BEEP [16] or serial lines.

In general, SOAP is a sufficient transport for NETCONF (essentially because of the remote procedure call character of both) but there are two areas of difficulty: the <rpc-progress> operation and the notification channel. The reason for this difficulty is the asynchronous aspect (from the point of view of the manager) of these features.

Five basic topics are presented: points about SOAP of interest to NETCONF, specifics on implementing NETCONF as a SOAP-based web service, security considerations, and an appendix with functional WSDL. In some sense, the most important part of the document is the brief WSDL document presented in the Appendix. In theory, this WSDL combined with the NETCONF XML Schemas provide machine readable descriptions sufficient for the development of software applications using NETCONF.
2. SOAP Background for NETCONF

Why introduce SOAP as yet another wrapper around what is already a remote procedure call message? There are, in fact, both technical and practical reasons. The technical reasons are perhaps less compelling, but let's examine them first.

The use of SOAP does offer a few technical advantages. SOAP is fundamentally an XML messaging scheme (which is capable of supporting remote procedure call) and it defines a natural message format composed of a "header" and a "body" contained within an "envelope". The "header" contains meta-information relating to the message, and can be used to indicate such things as store-and-forward behaviour or transactional characteristics. In addition, SOAP specifies an optional encoding for the "body" of the message. However, this encoding is not applicable to NETCONF as one of the goals is to have highly readable XML, and SOAP-encoding is optimized instead for ease of automated deserialization. These benefits of SOAP are message structure are simple, but worthwhile due to the fact that they are already standardized.

It is the practical reasons that truly make SOAP over HTTP an interesting choice for device management. It is not difficult to invent a mechanism for exchanging XML messages over TCP, but what is difficult is getting that mechanism supported in a wide variety of tools and operating systems and having that mechanism understood by a great many developers. SOAP over HTTP (with WSDL) is seeing good success at this, and this means that a device management protocol making use of these technologies has advantages in being implemented and adopted. Admittedly, there are interoperability problems with SOAP and WSDL, but such problems have wide attention and can be expected to be resolved.

2.1 Use and Storage of WSDL and XSD

One of the advantages of using machine readable formats such as Web Services Description Language (WSDL) [4] and XML Schemas [5] is that they can be used automatically in the software development process. With appropriate tools, WSDL and XSD can be used to generate classes that act as remote interfaces or application specific data structures. Other uses, such as document generation and service location, are also common. A great innovation found with many XML-based definition languages is the use of hyperlinks for referring to documents containing supporting definitions. For instance, in WSDL, the import statement

```xml
<import namespace="http://ietf.org/netconf/1.0/base"
    location="base.xsd"/>
```
imports the definitions of XML types and elements from the base
NETCONF schema. Ideally, the file containing that schema is hosted
on a web server under the authority of the standards body that
defined the schema. In this way, dependent standards can be built up
over time and all are accessible to automated software tools that
ensure adherence to the standards. Thus, it will gradually become as
important for ietf.org to host documents like

http://ietf.org/netconf/1.0/base/base.xsd

as they now host documents such as

http://www.ietf.org/rfc/rfc2616.txt

2.2 SOAP over HTTP

While it is true that SOAP focuses on messages and can be bound to
different underlying protocols such as HTTP or SMTP, most existing
SOAP implementations support only HTTP or HTTP/TLS. For this
discussion we will assume SOAP over HTTP or HTTP/TLS unless otherwise
specified. (This also includes applications of IPSec to SOAP over
HTTP.)

Note that there are a number of advantages to considering SOAP over
protocols other than HTTP, as HTTP is asymmetric with respect to
client and server. This causes difficulties in supporting
asynchronous notification (relieved in many ways by replacing HTTP
with BEEP). However, it is also the case that the full potential of
HTTP is not currently used by SOAP. For instance, multiple SOAP
replies to a single request could be contained in a multipart MIME
response. This would be a similar strategy to the use of
multipart/related with SOAP attachments [14].

2.3 HTTP Drawbacks

HTTP is not the ideal transport for messaging, but it is adequate for
the most basic interpretation of "remote procedure call". HTTP is
based on a communication pattern of the client (which initiates the
tcp connection) making a "request" to the server. The server returns
a "response" and this process is continued (possibly over a
persistent connection, as described below). This matches the basic
idea of a remote procedure call where the caller invokes a procedure
on a remote server and waits for the return value.

Potential criticisms of HTTP could include the following:

- server-initiated data flow is awkward
o headers are verbose and text-based

o idle connections may be closed by intermediate proxies

o data encapsulation must adhere to MIME

o bulk transfer relies on stream-based ordering

In many ways these criticisms are directed at particular compromises in the design of HTTP. As such, they are important to consider, but it is not clear that they result in fatal drawbacks for a device management protocol.

2.4 Important HTTP 1.1 Features

HTTP 1.1 [8] includes two important features that provide for relatively efficient transport of SOAP messages. These features are "persistent connections" and "chunked transfer-coding".

Persistent connections allow a single TCP connection to be used across multiple HTTP requests. This permits multiple SOAP request/response message pairs to be exchanged without the overhead of creating a new TCP connection for each request. Given that a single stream is used for both requests and responses, it is clear that some form of framing is necessary. For messages whose length is known in advance, this is handled by the HTTP header "Content-length". For messages of dynamic length, "Chunking" is required.

HTTP "Chunking" or "chunked transfer-coding" allows the sender to send an indefinite amount of binary data. This is accomplished by informing the receiver of the size of each "chunk" (substring of the data) before the chunk is transmitted. The last chunk is indicated by a chunk of zero length. Chunking can be effectively used to transfer a large XML document where the document is generated on-line from a non-XML form in memory.

In terms of application to SOAP message exchanges, persistent connections are clearly important for performance reasons, and are particularly important when it is the persistence of authenticated connections that is at stake. When one considers that messages of dynamic length are the rule rather than the exception for SOAP messages, it is also clear that Chunking is very useful. In some cases it is possible to buffer a SOAP response and determine its length before sending, but the storage requirements for this are prohibitive for many devices. Together, these two features provide a good foundation for device management using SOAP over HTTP.
3. A SOAP Web Service for NETCONF

3.1 Fundamental Use Case

The fundamental use case for NETCONF over SOAP (NETCONF/SOAP) over HTTP is that of a management console ("manager" role) managing one or more devices running NETCONF agents ("agent" role). The manager initiates one or more HTTP connections to the agent and drives the NETCONF sessions through repeated SOAP messages over HTTP requests. When the manager closes all HTTP connections associated with a session, the NETCONF session is also closed.

3.2 Mapping BEEP Channels to HTTP Connections

While the transport of SOAP over BEEP [17] has been specified, the purpose of this discussion is to describe how to map the BEEP [16] semantics and performance characteristics already assumed by NETCONF onto a (possibly persistent) SOAP over HTTP connection. This configuration is chosen because it is the one that benefits most from existing SOAP tools and implementations. It is true that BEEP has many advantages over HTTP for the transport of SOAP messages, but the fact remains that HTTP is currently more widely deployed than BEEP. At some point in the future, NETCONF/SOAP over BEEP may also be of interest. It can be easily dealt with as many of the issues already discussed in this document are pertinent. There would simply be a few enhancements regarding asynchronous notification.

NETCONF employs potentially three BEEP channels per session: the management channel, the operation channel, and the notification channel. In the SOAP over HTTP binding, each of these channels should be mapped to an individual HTTP connection (although the notification channel may remain in a BEEP channel in a separate TCP connection). Thus, SOAP messages on one connection (corresponding to the management channel) must be able to refer to SOAP messages on another connection (corresponding to the operation channel) as the "session" is potentially spread across multiple TCP connections. For instance, it may be necessary to abort a time-extended SOAP request on the "operation" HTTP connection by sending an "<rpc-abort>" message on the "management" HTTP connection.

It would be possible to assign distinct characteristics to the "operation" and "management" HTTP connections, but the simpler option is to allow any number of connections in the same session, each capable of "management" and "operation" procedure calls.

3.2.1 Asynchronous Functionality

NETCONF uses two types of asynchronous functionality and the mapping
of these onto SOAP over HTTP is somewhat problematic. The two asynchronous functions are <rpc-progress> and notifications on the notification channel. For <rpc-progress> it is recommended that a polling mechanism be supported by NETCONF. With this, a client could periodically poll on a secondary HTTP connection to obtain progress information on any outstanding operations on other HTTP connections in the same session.

The notification mechanism for NETCONF is specified in an existing standard for reliable syslog [13] and it is suggested that the same mechanism be used with the SOAP binding (it is simply external). If notifications via SOAP over HTTP are desired, it is probably most effective if an HTTP connection is established from the agent to the management console. Such a connection could be established in response to the manager connecting to the device. More sophisticated functionality, such as multiple SOAP replies to a single request, would require enhancements to the SOAP over HTTP specification.

### 3.3 NETCONF Sessions

NETCONF sessions are persistent for both performance and semantic reasons. NETCONF session state contains the following:

1. Authentication Information
2. Capability Information
3. Locks
4. Pending Operations
5. Operation Sequence Numbers

Authentication must be maintained throughout a session due to the fact that it is expensive to establish. Capability Information is maintained so that appropriate operations can be applied during a session. Locks are released upon termination of a session as this makes the protocol more robust. Pending operations come and go from existence during the normal course of RPC operations. Operation sequence numbers provide the small but necessary state information to refer to operations during the session.

Since it is generally not possible to support a full NETCONF session with a single HTTP connection, it is necessary to identify the NETCONF session in a way that can span multiple HTTP connections. This can be performed with the SOAPAction HTTP header, as in:

```plaintext
POST /netconf HTTP/1.0
```
Content-Type: text/xml; charset=utf-8
SOAPAction: "netconfsession:123"
Content-Length: 470

Note that the session identifier must either be known by the manager
(in order to attach to an existing session) or be communicated from
the agent to the manager prior to the exchange of any significant
NETCONF messages. It is recommended that the session identifier be
generated and placed in a SOAP header in the reply to the first SOAP
request with an empty SOAPAction. It may not be an error to continue
to perform operations without specifying a NETCONF session, but the
user must be aware that the only way to abort such operations is to
close the HTTP connection.

Thus, in the case of SOAP over HTTP, a NETCONF "session" is a
collection of HTTP connections with common authenticated users and a
common session identifier as indicated in the SOAPAction HTTP header.
To support automated cleanup, a NETCONF session is closed when all
connections associated with that session are closed.

3.4 Capabilities Exchange

Capabilities exchange, if defined through a NETCONF RPC operation,
can easily be accommodated in the SOAP binding.

3.5 A NETCONF/SOAP example

Since the proposed WSDL (in Appendix A.1) uses document/literal
encoding, the use of a SOAP header and body has little impact on the
representation of a NETCONF operation. This example shows HTTP/1.0
for simplicity.

POST /netconf HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, text/*
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: "netconfsession:123"
Content-Length: 470

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope
 xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:soapenv:Body>
 <soapenv:Body>
   <rpc id="101" xmlns="http://ietf.org/netconf/1.0/base">
     <get-config>
       <source>
         <running/>
The HTTP/1.0 response is also straightforward:

HTTP/1.0 200 OK
Content-Type: text/xml; charset=utf-8

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:soapenv2="http://schemas.xmlsoap.org/soap/uri/2001/12/soap-envelope"
  xmlns:i="http://xmlns://example.com/schema/1.2/config"
  xmlns:rpc="http://ietf.org/netconf/1.0/base">
  <soapenv:Body>
    <rpc-reply id="101" xmlns="http://ietf.org/netconf/1.0/base">
      <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>
  </soapenv:Body>
</soapenv:Envelope>

3.6 Managing Multiple Devices

When a server is acting as a proxy for multiple devices, the URL for
the HTTP POST can be used to indicate which device is the target. It
may also be desirable to use the HTTP POST URL as a means for
selecting from multiple virtual devices on a single device.
4. Security Considerations

4.1 Integrity, Privacy, and Authentication

The NETCONF SOAP binding relies on an underlying secure transport for integrity and privacy. Such transports are expected to include TLS [11] and IPSec. There are a number of options for authentication (some of which are deployment-specific):

- within the transport (such as with TLS client certificates)
- within HTTP (such as Digest Access Authentication [9])
- within SOAP (such as a digital signature in the header [15])

HTTP and SOAP level authentication can be integrated with RADIUS [12] to support remote authentication databases.

4.2 Vulnerabilities

The above protocols may have various vulnerabilities, and these may be inherited by NETCONF/SOAP.

NETCONF itself may have vulnerabilities due to the fact that an authorization model is not currently specified.

It is important that device capabilities and authorization remain constant for the duration of any outstanding NETCONF session. In the case of NETCONF/SOAP, this constancy must be given particular attention as a session may span multiple HTTP connections.

4.3 Environmental Specifics

Some deployments of NETCONF/SOAP may choose to use HTTP without encryption. This presents vulnerabilities but is reasonable for closed networks or debugging scenarios.

A device managed by NETCONF may interact (over protocols other than NETCONF) with devices managed by other protocols, all of differing security. Each point of entry brings with it a potential vulnerability.
Normative References


Informative References


Author’s Address

Ted Goddard
Wind River Systems
#180, 6815-8th Street NE
Calgary, AB T2E 7H7
Canada

Phone: (403) 730-7590
EMail: ted.goddard@windriver.com
URI: http://www.windriver.com
Appendix A. WSDL Definitions

A.1 NETCONF SOAP Binding

The following WSDL document assumes a hypothetical location for the
NETCONF schema.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<definitions
    xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/
    xmlns:tns="http://ietf.org/netconf/1.0/soap"
    xmlns:xb="http://ietf.org/netconf/1.0/base"
    targetNamespace="http://ietf.org/netconf/1.0/soap"
    name="http://ietf.org/netconf/1.0/soap">
    <import namespace="http://ietf.org/netconf/1.0/base"
        location="base.xsd"/>
    <message name="rpcRequest">
        <part name="in" element="xb:rpc"/>
    </message>
    <message name="rpcResponse">
        <part name="out" element="xb:rpc-reply"/>
    </message>
    <portType name="rpcPortType">
        <operation name="rpc">
            <input message="tns:rpcRequest"/>
            <output message="tns:rpcResponse"/>
        </operation>
    </portType>
    <binding name="rpcBinding" type="tns:rpcPortType">
        <SOAP:binding style="document"
            transport="http://schemas.xmlsoap.org/soap/http"/>
        <SOAP:operation/>
        <input>
            <SOAP:body use="literal"
                namespace="http://ietf.org/netconf/1.0/base"/>
        </input>
        <output>
            <SOAP:body use="literal"
                namespace="http://ietf.org/netconf/1.0/base"/>
        </output>
    </SOAP:binding>
</definitions>
```
A.2 Sample Service Definition

The following WSDL document assumes a hypothetical location for the NETCONF/SOAP WSDL definitions. A typical deployment of a device manageable via NETCONF/SOAP would provide a service definition similar to the following to identify the address of the device.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<definitions
xmlns="http://schemas.xmlsoap.org/wSDL/
xmlns:SOAP="http://schemas.xmlsoap.org/wSDL/soap/
xmlns:xs="http://ietf.org/netconf/1.0/soap"
targetNamespace="urn:myNetconfService"
name="myNetconfService.wsdl">
    <import namespace="http://ietf.org/netconf/1.0/soap"
     location="soap.wsdl"/>
    <service name="netconf">
        <port name="rpcPort" binding="xs:rpcBinding">
            <SOAP:address location="http://localhost:8080/netconf"/>
        </port>
    </service>
</definitions>
```
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 (2003). 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
HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF
MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

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