Netconf
Internet-Draft
Intended status: Informational
Expires: December 28, 2018

G. Zheng
M. Wang
Huawei

June 26, 2018

A NETCONF Extension for Data Fragmentation
draft-zheng-netconf-fragmentation-02

Abstract

This document introduces an extension to NETCONF (Network Configuration) protocol. The extension allows NETCONF to handle large size data as fragmented RPC messages. Specifically, this document defines a new <get-block> capability and relevant operations to handle the fragmentations.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on December 28, 2018.

Table of Contents

1. Introduction

NETCONF [RFC6241] is the next generation network management protocol for configuring devices. It is becoming more and more popular, and some NMS (Network Management System) only use NETCONF as its southbound interface. The message procedures of NETCONF are based on RPC (Remote Procedure Call) interactions. A NETCONF client/server sends a <rpc> message to the counterpart and then receives a replying <rpc-reply> message.

In some situations, the <rpc-reply> message might be very large. For example, when NMS is retrieving a large amount of routes in a core router or doing a full-synchronizing with a device, the <rpc-reply> data might exceed Mega-Byte amount. And also in some scenarios, the client may retrieve a continuous stream of operational data from the operational datastore [I-D.ietf-netmod-revised-datastores] to perform network analytics. Then there comes the problem of how to handle the large size data. This document briefly introduces two typical ways of current handling on this issue; and analyzes the problems of them.

To fix the problems, in Section 4, this document proposes a method of extending the NETCONF protocol to allow handling large size data as fragmented <rpc-reply> messages. The fragmentation is done at the NETCONF level, so it allows the NETCONF client to terminate the large

size data processing momentarily by protocol interactions; and also allows the fragmented messages to be instantly parsed piece by piece. Specifically, the fragmentation is achieved through a newly defined <get-block> capability and relevant operations.

2. Concept and Terminology

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 [RFC2119].

2.1. Terminology

DOM: Document Object Model, which is a cross-platform and language-independent convention for representing and interacting with objects in HTML, XHTML and XML documents. Objects in the DOM tree may be addressed and manipulated by using methods on the objects.

SAX: Simple API for XML, which is an event sequential access parser API developed by the XML-DEV mailing list for XML documents. SAX provides a mechanism for reading data from an XML document that is an alternative to that provided by the DOM. Where the DOM operates on the document as a whole, SAX parsers operate on each piece of the XML document sequentially.

libxml: a software library for parsing XML documents.

<get-block>: a capability and operation defined in this document to handle large size

3. Current Large size Handling Methods and Problems

3.1. Stream-Oriented Handling

Stream-Oriented handling mainly includes the following two aspects:

- The server encapsulates the large size replying data in a <rpc-reply> message and streams it to the client through TCP protocol.

- The client parses the received <rpc-reply> content in a stream-oriented way. More specifically, the client could utilize SAX to

instantly parse the received content without waiting for the whole message been transported.

Problems:

o Stream-Oriented method lacks the capability of discontinuing large size processing in the server. It would cause unnecessary resource/performance cost in the devices if the NMS has already got the intended portion or just canceled by the administrators.

o Another problem is the implementation of SAX parsing is more complex than DOM parsing in the Netconf client. More computing burden will be taken in Netconf client to support SAX parsing.

3.2. Requesting a Portion of Data

The clients actively limit the search range of the data so that the servers only need to reply with a part of the large size data. Thus the clients could control the replies in a reasonable size. One example is that the clients get a list of the content, and provide a start offset and a max-count, to get a portion at a time.

Problems:

o This method has an implication that the client needs to know the list/index of the intended large size data in advance before it starting the search request. It can't fit the scenarios of real-time on-demand data retrieving. And there is no standard to specify the list/index format in a uniform way. Thus it is only suitable for private implementation, thus multi-vendor interaction is not supported.

o More important, it is just an indirect way to solve the problem. It could not fit the scenarios where the client just needs the whole large size data in the server.

3.3. Using tools to extract data respectively

There are some tools/functions which can handle the iteration. For

example, the XPATH's position() function can help to do this work. The following examples show some instances:

```
<filter type="xpath" select="/interfaces/interface[position() >= 0 and position

<filter type="xpath" select="/interfaces/interface[position() >= 25 and positio
```

Problems:

- Xpath function just can meet the "xpath filter" case, which cannot cover the "subtree filter" case.

- Different encoding format request different tools, and there is no standard to process the received data in a uniform way. Thus it is only suitable for private implementation, thus multi-vendor interaction is not supported.

4. Netconf Fragmentation Mechanism

4.1. Fragmentation Requirements

this document proposes an RPC fragmentation mechanism to handle the large size data. Two essential requirements of the fragmentation are:

- It needs to allow the NETCONF client to terminate the large size data processing momentarily by protocol interactions. In the proposed mechanisms in this draft, when the NETCONF server replies the client an <rpc-reply> fragmentation, it will wait the response from the client that whether it needs to send the next fragmentation. So if the initiator has got the intended portion, it could terminate the large size process immediately.

- It needs to allow the NETCONF client to instantly parse the fragmentations piece by piece through the more widely supported DOM parsing. So in this document, it specifies that each <rpc-reply> fragmentation MUST be in a complete XML form.

4.2. <get-block> extention

o Function

The devices can only use <get-block> operation when the Get-block capability was announced.

The <get-block> fragmentation rules are:

A. There should be a Max-Size for fragmentation. [Ope Question]Should there be a clear specification of the size? E.g. 64K bytes.

B. When the message reaches the Max-Size, it is sent to the client and the next message could be created in advance.

C. Different records from one same table could be put into different <rpc-reply> messages

D. All of the fields in one record MUST be put into one <rpc-reply> message.

E. XML syntax MUST be complete in each fragmented message, so that each fragmentation could be parsed individually.

F. If the record(s) of the child node(s)/table(s) and the parent node(s)/table(s) are replied in different fragmentations, the

child node/table fragmentations MUST include the path and index information of all the ancestor node(s)/table(s) in a hierarchical mode.

o Parameters

<discard-fragmentation>: in <get-block> operation, if the <discard-fragmentation> parameter is conveyed, it means the operation is terminated. Then it doesn't need to reply the remaining fragmentations.

o Successful Operation Reply

A <rpc-reply> message conveying a <data> element indicates the operation is successful. If there exists a next fragment, then an

set-id attribute MUST be included in the <rpc-reply> messge. The attribute set-id is used to identify different fragment sets.

o Exception Handling

After the NETCONF server replies a fragment, if there is no corresponding Get-block request from the client in a reasonable period (the time valued to be specified in the future), then the server release the offset of the replying data and cannot use <get-block> operation anymore, and the remaining data needs to be replied.

Please refer to Appendix A.1 for an example.

5. YANG data model

```
<CODE BEGINS> file "ietf-netconf-fragmentation@2018-06-20.yang"
 module ietf-netconf-fragmentation {
     yang-version 1.1;
     namespace "urn:ietf:params:xml:ns:yang:ietf-netconf-fragmentation";
     prefix fgm;

         import ietf-datastores{
          prefix ds;
         }

     import ietf-yang-types {
       prefix yang;
     }

     import ietf-netconf {
       prefix nc;
     }
```



```
     organization
       "IETF NETCONF Working Group";
     contact
       "WG Web:  <https://datatracker.ietf.org/wg/netconf/>

        WG List:  <mailto:netconf@ietf.org>

        Author:   Guangying Zheng
```

```
                        <zhengguangying@huawei.com>

    Author:   Zitao Wang
                        <wangzitao@huawei.com>";
  description
    "This document introduces an extension to NETCONF (Network Configuration
       The extension allows NETCONF to handle large size data as fragmented
       Specifically, this document defines a new get-block capability and r
       operations to handle the fragmentations.";

  revision 2018-06-20 {
    description
      "Initial revision.";
    reference "draft-zheng-netconf-fragmentation-01";
  }


      rpc get-block {
    description
      "Retrieve data from an NMDA datastore.";
    input {
      leaf source {
        type identityref {
         base ds:datastore;
        }
        mandatory true;
        description
          "Datastore from which to retrieve data.";
      }

      choice filter-spec {
        description
          "The content filter specification for this request.";

        anydata subtree-filter {
          description
            "This parameter identifies the portions of the
             target datastore to retrieve.";
          reference "RFC 6241, Section 6.";
        }
```



```
        leaf xpath-filter {
```

```
            if-feature nc:xpath;
            type yang:xpath1.0;
            description
              "This parameter contains an XPath expression
               identifying the portions of the target
               datastore to retrieve.";
          }
        }
     }
        }
        rpc discard-fragmentation {
            description
        "Discard the netconf fragmentation, if the discard parameter is conveye
               it means the operation is terminated.
               Then it doesn't need to reply the remaining fragmentations.";
    }

}
<CODE ENDS>
```

6. Security Considerations

TBD.

7. IANA Considerations

TBD.

8. Normative References

[I-D.ietf-netmod-revised-datastores]
Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture", draft-ietf-netmod-revised-datastores-10 (work in progress), January 2018.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

Appendix A. Appendix A: Examples of the Candidate Solutions

A.1. <get-block> (RPC Fragmentation) Example:

Example 1: Get the next fragment

```
  <rpc message-id="101"
       xmlns="urn:ietf:params:xml:ns:NETCONF:base:1.0">
    <get-config>
      <source>
        <running/>
      </source>
      <filter type="subtree">
        <top xmlns="http://example.com/schema/1.2/config">
          <users/>
        </top>
      </filter>
    </get-config>
  </rpc>

  <rpc-reply message-id="101"
       xmlns="urn:ietf:params:xml:ns:NETCONF:base:1.0"
       xmlns:hw=http://www.huawei.com/NETCONF/capability/base/1.0
hw:set-id="101">
     <data>
       <top xmlns="http://example.com/schema/1.2/config">
         <users>
           <user>
             <name>root</name>
             <type>superuser</type>
             <full-name>Charlie Root</full-name>
             <company-info>
               <dept>1</dept>
               <id>1</id>
             </company-info>
           </user>
           <!-- additional <user> elements appear here... -->
         </users>
       </top>
     </data>
  </rpc-reply>

<rpc message-id="102"
       xmlns="urn:ietf:params:xml:ns:NETCONF:base:1.0">
     <get-block xmlns="http://www.huawei.com/NETCONF/capability/base/1.0"
set-id="101">
```

```
       </get-block>
     </rpc>
```



```
     <rpc-reply message-id="102"
          xmlns="urn:ietf:params:xml:ns:NETCONF:base:1.0"
          xmlns:hw=http://www.huawei.com/NETCONF/capability/base/1.0
hw:set-id="101">
       <data>
         <top xmlns="http://example.com/schema/1.2/config">
           <users>
             <user>
               <name>admin</name>
               <type>commonuser</type>
               <full-name>Jim Green</full-name>
               <company-info>
                 <dept>9</dept>
                 <id>90</id>
               </company-info>
             </user>
             <!-- additional <user> elements appear here... -->
           </users>
         </top>
       </data>
     </rpc-reply>
```

Example 2: Abandon the remaining fragments

```
     <rpc message-id="103"
          xmlns="urn:ietf:params:xml:ns:NETCONF:base:1.0">
       <get-block xmlns=http://www.huawei.com/NETCONF/capability/base/1.0 set-i
         <discard-fragmentation/>
       </get-block>
     </rpc>

     <rpc-reply message-id="103"
       xmlns="urn:ietf:params:xml:ns:NETCONF:base:1.0">
       <ok/>
     </rpc-reply>
```

Authors' Addresses

Guangying Zheng
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: zhengguangying@huawei.com

Michael Wang
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: wangzitao@huawei.com