NETCONF Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 14, 2017

Z. Wang
G. Zheng
Huawei Technologies

March 13, 2017

Network Configuration Protocol (NETCONF) Proxy
draft-wangzheng-netconf-proxy-00

Abstract

This document presents Network Configuration Protocol (NETCONF) Proxy through which NETCONF requests can be forwarded to a target host. It would be useful when a client does not have direct network access to a target host.

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 http://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 September 14, 2017.

Copyright Notice

Table of Contents

1. Introduction

This document proposes a NETCONF Proxy mechanism. The mechanism extends the NETCONF protocol [RFC6241] which provides a standard way to set up NETCONF session. At its core, the mechanism defined here introduces a set of new operations which allow a client to forward NETCONF requests to a target host through an intermediary NETCONF proxy server, especially in case where client would otherwise not have direct network access to a target host. The document also includes YANG data model which extend the model and RPCs defined within [RFC6241].

1.1. Motivation

NETCONF provide a RPC-based mechanism to facilitate communication between a client and a server. The client can be a script or application typically running as part of a network manager. The server is typically a network device [RFC6241]. However, the network manager may not have direct network access to the target network devices. For example, some target network devices may locate in a network with private addresses behind a NAT device or a firewall. Thus, network manager cannot direct communicate with these target devices.

NETCONF Call Home [RFC8071] provides a mechanism that allows NETCONF Servers to initiate a connection with a NETCONF client, reversing the normal direction of NETCONF session setup. This allows a NETCONF Server, e.g. a networking device that needs to be managed, to reach out to a NETCONF Client, e.g. an Operations Support System of an SDN controller, in order to be managed. By reversing the direction in

which NETCONF sessions are normally set up, problems such as establishing connectivity with devices behind a firewall can be alleviated. However, NETCONF Call Home requires that the server knows its client by way of configuration or discovery. It does not address the scenarios as presented below:

1. In some NFV scenarios, VNF instances are running in a private network. To reduce the management resources (like IP resources, bandwidth, etc) of large-scale management activities, these VNF instances may not be assigned IP addresses. Then the element management system (EMS), which located in public network, cannot be aware of the addresses of VNF instances. Therefore, the element management system (EMS) is difficult to manage these VNF instances via NETCONF protocol.

2. And in some cloud network scenarios, the gateway network element (GNE) and non-gateway network elements (N-GNEs) communicate with each other using some private protocol. And these non-gateway network elements (N-GNEs) may not IP devices. Therefore, the cloud centre EMS (element management system) cannot be aware of the addresses of N-GENs. Thus, the element management system (EMS) is difficult to manage these N-GNE devices via NETCONF protocol.

To solve that problem, this document proposes a NETCONF Proxy mechanism. The proxy can acts as an intermediary between manager and target device, therefore the client can set up a NETCONF session to a target through a NETCONF Proxy.

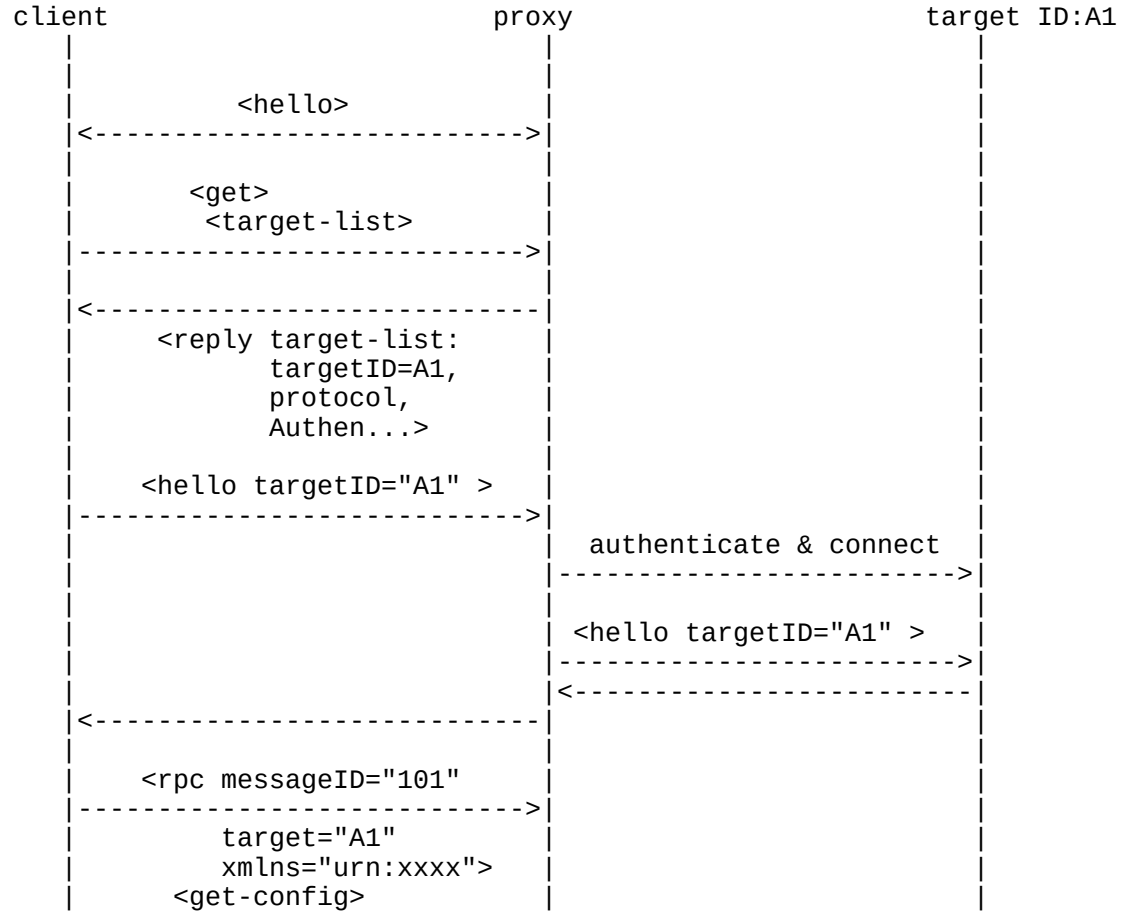
The mechanism allows the client to subsequently direct NETCONF requests to the server, to receive responses, and to subscribe to notifications from the server. While the NETCONF Proxy can be used to traverse NAT boundaries, it should be noted that it does not apply NAT mappings for contents carried as part of the NETCONF payload; specifically, it does not substitute IP address information that is carried as part of data nodes.

1.2. Requirements 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 RFC 2119 [RFC2119].

2. Solution Overview

The diagram below illustrates how the client can set up a NETCONF session to a target through the NETCONF proxy.



This diagram makes the following points:

1. The client initiates the connection using the SSH/TLS transport protocol. When the NETCONF session is established, the client and proxy MUST send a <hello> element containing a list of that peer's capabilities. The proxy SHOULD send at least the "netconf" and "proxy" capabilities. And other rules of capabilities exchange described in section 8 of [RFC6241].

2. The client sends a <get> RPC to proxy to retrieve the "target-list" of the proxy.

3. The proxy responds with a <get-reply> RPC which containing "target-list" attributes. The "target-list" attributes includes

the target's information such as target-id, protocol, authentication, etc.

4. The client receive a <get-reply> RPC from the proxy, and looks up the value of "target-list". Then the client constructs a <hello> message according to the received "target-list". This <hello> massage SHOULD contain at least a "target-id" attribute. And then client sends this <hello> message to proxy and waits for a reply.

5. The proxy receives the <hello> message and checks the value of "target-id" attribute:

   If the target is not found, then an "invalid-target" error will be returned.

   If the target can be found, then the proxy initiates a connection to corresponding target. And then proxy forwards the <hello> message, which received from client, to corresponding target.

6. The target receives the <hello> message and then responds a <hello> message containing a list of capabilities. The rules of capabilities exchange described in section 8 of [RFC6241].

7. Now, client established a NETCONF session to a target through NETCONF Proxy. Subsequently, the client can direct NETCONF requests to the target, to receive responses, and to subscribe to notifications from the target.

3. The NETCONF Client

The term "client" is defined in [RFC6241], Section 1.1 "client". In the context of network management, the NETCONF client might be a network management system for example a EMS (element management system).

The client operation describes as follows:

1. The client initiates a connection to proxy using the SSH/TLS transport protocol [RFC6242]. How to establish an SSH/TLS transport connection is described in [RFC6242]

2. When the NETCONF session is established, the client sends a <hello> message to proxy, then waits for a reply. This <hello> message contains a list of client's capabilities.

3. After capabilities exchange, the client sends a <get> RPC to proxy to retrieve the "target-list" of the proxy, then waits for a reply.

4. The client receive a <get-reply> RPC from the proxy, looks up the value of "target-list", and then constructs a <hello> message according to the received "target-list". This <hello> massage SHOULD contain at least a "target-id" attribute. For example, if the client received "target-list" containing "target-id=A1", then the client constructs <hello> message:

```
C: <hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
C:   <target-id>A1</target-id>
C:   <capability>foo<capability>
C: </hello>
```

And then client sends this <hello> message to proxy and waits for a reply.

5. If the reply presents the "capabilities" of target, the proxy connection is established. Subsequently, the client can direct NETCONF requests to the target, to receive responses, and to subscribe to notifications from the target.

6. If the reply contains the "invalid-target" error, the process turn to step (4) or aborts.

7. Otherwise, the client interprets the error and aborts.

4. The Proxy

The Proxy should ensure that requests given by client for a particular target device should reach the target device and the operations should be executed on that target device and the response should be given back to the client.

The proxy operation describes as follows:

1. When the NETCONF session is established, the proxy sends a <hello> element containing a list of proxy’s capabilities. The proxy SHOULD send at least the "netconf" and "proxy" capabilities. And other rules of capabilities exchange described in section 8 of [RFC6242].

2. The proxy receives the <get> RPC and then responds with a <get-reply> RPC which containing "target-list" attributes. The "target-list" attributes SHOULD includes the target's information such as target-id, protocol, authentication, etc. The following example shows a "target-list":

```
<target-list>
 <target-id>A1</target-id>
 <protocol>protocol-foo</protocol>
 <authentication>authentication-foo</authentication>
</target-list>
```

3. The proxy receives the <hello> message and checks the value of "target-id" attribute:

   If the target is not found in target-list, then an "invalid-target" error will be returned.

   If the target can be found in target-list, the proxy checks the corresponding "protocol" and "authentication" of the "target-id". And then, the proxy uses corresponding protocol to establish a connection to the target. After session established, the proxy forwards the <hello> message, which received from client, to corresponding target.

4. If the reply presents the "capabilities" of target, the proxy connection is established. Subsequently, the proxy transports the messages received from the client to the target and vice versa.

5. The Target

The term "target" is equivalent to the term "server" which is defined in [RFC6242], Section 1.1 "server". In the context of network management, the target is typically a network device.

The target operations describes as follows:

1. If the connection between the proxy and the target established. And target receives the <hello> message from the proxy, and then responds a <hello> message containing a list of capabilities. The rules of capabilities exchange described in section 8 of [RFC6242].

2. After client established a NETCONF session to a target through NETCONF Proxy. The client sends a series of one or more RPC request messages, which cause the server to respond with a corresponding series of RPC reply messages.

6. New attribute: target-id

A proxy can be used by a client to connect to several servers and to maintain multiple NETCONF sessions. A client may use the proxy even to maintain multiple NETCONF sessions with the same NETCONF server. When issuing a NETCONF request, a client must therefore differentiate between NETCONF sessions. To solve this problem, a new attribute "target-id" is defined. This attribute allow the proxy to forward RPC to corresponding target.

For example:

The following <rpc> element invokes the NETCONF <get> method and includes the "target-id" attribute:

```
<rpc message-id="101"
     target-id="A1"
     xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get/>
</rpc>

<rpc-reply message-id="101"
     target-id="A1"
     xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <!-- contents here... -->
  </data>
</rpc-reply>
```

7. YANG DATA MODEL

7.1. Overview

The YANG data model for NETCONF Proxy is depicted in the following figure. Following Yang tree convention in the depiction, brackets enclose list keys, "rw" means configuration, "ro" operational state data, "?" designates optional nodes, "*" designates nodes that can have multiple instances. A "+" at the end of a line indicates that the line is to be concatenated with the subsequent line.

```
module: ietf-netconf-proxy
    +--rw proxy {proxy}?
       +--rw proxy-name?    string
       +--rw target-list* [target-id]
          +--rw target-id         string
          +--rw protocol?         string
          +--rw authentication?   string
```

7.2. YANG Module

```
<CODE BEGINS> file "ietf-netconf-proxy@2017-03-09.yang"
module ietf-netconf-proxy {

  namespace "urn:ietf:params:xml:ns:yang:ietf-netconf-proxy";

  prefix np;

  organization
    "IETF NETCONF (Network Configuration) Working Group";

  contact
    "WG Web:   http://tools.ietf.org/wg/netconf
     WG List:  netconf@ietf.org

     WG Chair: Mehmet Ersue
               mehmet.ersue@nsn.com

     Editor:   zitao wang
                   wangzitao@huawei.com";

  description
    "NETCONF Protocol Data Types and Protocol Operations.

     Copyright (c) 2011 IETF Trust and the persons identified as
     the document authors.  All rights reserved.

     Redistribution and use in source and binary forms, with or
     without modification, is permitted pursuant to, and subject
     to the license terms contained in, the Simplified BSD License
     set forth in Section 4.c of the IETF Trust's Legal Provisions
     Relating to IETF Documents
     (http://trustee.ietf.org/license-info).

     This YANG module describe how to define a netconf proxy";

  revision 2017-03-09 {
    description
```

```
        "Initial revision";
      reference
        "draft-wang-netconf-proxy";
    }

    feature proxy {
      description
        "Netconf proxy";
    }

    container proxy {
     if-feature proxy;
     leaf proxy-name{
      type string;
      description
          "Proxy name";
     }
     list target-list {
      key "target-id";
          leaf target-id{
           type string;
           description
           "Target ID";
          }
          leaf protocol {
           type string;
           description
           "Support protocols";
          }
          leaf authentication {
           type string;
           description
           "Authentication";
          }
          description
          "List for target information";
     }
     description
     "Container for NETCONF Proxy";
    }

  }
  <CODE ENDS>
```

8. Security Considerations

The security considerations described in [RFC6242] and [RFC7589], and by extension [RFC4253], [RFC5246] apply here as well.

9. IANA Considerations

TBD

10. Normative References

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

[RFC4253] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Transport Layer Protocol", RFC 4253, DOI 10.17487/RFC4253, January 2006, <http://www.rfc-editor.org/info/rfc4253>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[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, <http://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <http://www.rfc-editor.org/info/rfc6242>.

[RFC7589] Badra, M., Luchuk, A., and J. Schoenwaelder, "Using the NETCONF Protocol over Transport Layer Security (TLS) with Mutual X.509 Authentication", RFC 7589, DOI 10.17487/RFC7589, June 2015, <http://www.rfc-editor.org/info/rfc7589>.

[RFC793] Postel, J., "TRANSMISSION CONTROL PROTOCOL", STD 7, September 1981, <https://www.ietf.org/rfc/rfc793.txt>.

Authors' Addresses

Zitao Wang
Huawei Technologies
101 Software Avenue, Yuhua District
Nanjing
China

EMail: wangzitao@huawei.com

Guangying Zheng
Huawei Technologies
101 Software Avenue, Yuhua District
Nanjing
China

EMail: zhengguangying@huawei.com