

NETCONF Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 10, 2019

Q. Wu
C. Feng
Huawei
March 9, 2019

NMDA protocol operation Backwards-Compatibility with Legacy Devices
draft-wu-netconf-nmda-compatibility-01

Abstract

NMDA architectural framework eliminates the need to duplicate data structures to provide separate configuration and operational state sections and uses different datastores and new protocol operations to distinct configuration from operation state. However when non-NMDA clients communicate with NMDA aware server, it is not clear whether the NMDA aware server can use existing operation to return the same results with <rpc-reply> as non-NMDA-aware server does.

This document identifies some of the major challenges, and provides solutions that are able to mitigate those challenges and smooth the migration towards NMDA deployment.

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 September 10, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of

Internet-Draft

NMDA Backwards-Compatibility

March 2019

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	2
1.1.	Terminology	3
2.	Problems	4
2.1.	NMDA Client vs Non-NMDA Client	4
2.2.	NETCONF Server Back-Compatibility	4
2.3.	Feed System Configuration Back into Running Datastore	4
3.	Solution	5
3.1.	Client NMDA Support	5
3.2.	Default handling Behaviour	5
3.2.1.	Default-Handling Basic Modes	5
3.2.2.	get/get-config Operation	6
3.2.3.	get-data Operation	7
3.3.	Protocol Operation Clarification	7
3.3.1.	<get>	7
3.3.2.	<get-config>	7
3.3.3.	<edit-config>	8
3.3.4.	<get-data>	8
3.3.5.	<edit-data>	8
4.	IANA Considerations	9
5.	Security Considerations	9
6.	Acknowledgements	9
7.	Contributors	9
8.	References	9
8.1.	Normative References	9
8.2.	Informative References	10
	Authors' Addresses	10

[1.](#) Introduction

NMDA architectural framework introduces additional datastores for systems that support more advanced processing chains converting configuration to operational state. It eliminates the need to duplicate data structures to provide separate configuration and operational state sections and uses different datastores and new

protocol operations (e.g., <get-data>, <edit-data> to distinct configuration from operation state).

However when a NMDA client communicates with NMDA aware server, it is not clear whether the NMDA aware server can return the same results

with <rpc-reply> to non-NMDA clients as non-NMDA-aware server does since the system configuration and default configuration originally part of conventional configuration datastores have been explicitly separated and moved to operational state datastore under NMDA architecture. Also it is not clear whether the server should maintain protocol operation backwards-compatibility when the NMDA aware server is upgraded from non-NMDA-aware server to NMDA aware server.

NMDA Transition Guidelines in [section 4.23.3 of \[RFC8407\]](#) only provides guidelines to transform non-NMDA compliant model into NMDA compatible model, but doesn't provide guidelines on whether existing NETCONF protocol operations such as get/get-config/edit-config change semantics when they are exchanged between the non NMDA client and the NMDA aware server.

This document identifies some of the major challenges, and provides solutions that are able to address those challenges which provide smooth migration towards NMDA deployment.

- o Use protocol operation to indicate whether the client NMDA support.
- o An extension to Default handling Behaviour.
- o Protocol Operation Clarification.

[1.1](#). Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [BCP 14 \[RFC2119\]](#) [\[RFC8174\]](#) when, and only when, they appear in all capitals, as shown here.

The following terms are defined in [\[RFC8342\]](#) and are not redefined

here:

- o startup configuration datastore
- o candidate configuration datastore
- o running configuration datastore
- o intended configuration datastore
- o operational state datastore

The following terms are defined in this document:

Server Protocol Operation Backwards-Compatibility: The client can use the same protocol operation to get the same results from both NMDA compliant server and Non NMDA server.

[2.](#) Problems

[2.1.](#) NMDA Client vs Non-NMDA Client

When a server is upgraded to NMDA aware server and needs to support both NMDA client and non-NMDA clients, there is no standards-based way for the server to know whether the client supports NMDA.

[2.2.](#) NETCONF Server Back-Compatibility

When a server is upgraded to NMDA server and needs to support both NMDA client and non-NMDA clients, without NETCONF server protocol operation backwards-compatibility, a NMDA server can return the results with <rpc-reply> to non-NMDA clients different from non-NMDA-aware server does. Since in some cases ('report-all' Retrieval Mode or client set Explicit Mode) default configuration originally part of conventional configuration datastores have been explicitly separated and moved to operational state datastore, so does the system configuration. For non-NMDA client, the configuration retrieved by <get-config> on the running datastore will be reduced without including system configuration and default configuration set by the server. Non-NMDA client also has no way to retrieve system configuration without new operations support on operational state

datstore.

[2.3.](#) Feed System Configuration Back into Running Datastore

As we know, the system configuration and default configuration originally part of conventional configuration datastores before NMDA is introduced have been separated and moved to operational state datastore. When further configuration is needed within the system configuration, e.g., configure IP address of the interface after such interface configuration (i.e., system configuration) is auto-created, such auto-created interface configuration needs to be set by the client since it doesn't exist in the conventional configuration datastore. The effect is the same as feeding auto-created interface configuration into running datastore and making it become client set configuration. After the interface configuration is applied, it will be merged with the other existing system configuration in the operational state datastore.

[3.](#) Solution

[3.1.](#) Client NMDA Support

When a NMDA aware server needs to support both NMDA client and non-NMDA clients, server NMDA support can be advertised to the client via capability identifier `:yang-library:1.1` to the client. Client NMDA support should be indicated by protocol operations. If `<get>/<get-config>/<edit-config>` operation is received from the client, the server should assume the client is Non-NMDA client. If `<get-data>/<edit-data>` operation is received from the client, the server should assume the client is NMDA client.

Editor-Note: There are three ways to indicate client support on NMDA:

1. Define capability identifier for client NMDA support and advertising this capability identifier to the server;
2. Use new or old protocol operation to indicate client NMDA support;
3. Use whether module type is NMDA aware to indicate client NMDA

support ;

Either advertising capability identifier to the server or using module type to indicate client NMDA support adds server implementation complexity. We argue to use protocol operation to indicate whether the client support NMDA.

[3.2.](#) Default handling Behaviour

With-default capability defined in [[RFC6243](#)] is designed for conventional configuration datastore. As described in [I-D.ietf-netconf-nmda-netconf], semantics for "with-defaults" in <edit-data> operations is the same as for <edit-config>, as described in [section 4.5.2 of \[RFC6243\]](#). However when the NMDA is introduced, the default configuration is clearly separated from conventional configuration datastore, the semantics of "with-defaults" parameter have made a few changes.

[3.2.1.](#) Default-Handling Basic Modes

A NMDA aware server still supports three basic modes defined in [[RFC6243](#)] for handling default data. In addition to 'report-all' Retrieval Mode, the 'report-all' basic mode should be also treated in the same way as 'explicit' basic model in case of NMDA since the default configuration has been moved to operational state datastore and therefore the NMDA aware server should not consider the default

configuration is part of conventional configuration datastore unless it is explicitly set by the client.

[3.2.1.1.](#) 'report-all' Basic Mode Retrieval

When the data is retrieved from a server using the 'report-all' basic mode, and the <with-defaults> parameter is not present, data nodes MUST be reported if explicitly set by the client, even if they contain the schema default value.

[3.2.1.2.](#) 'report-all' <edit-config> and <copy-config> Behaviour

For backwards compatibility consideration, the server consider the default data part of conventional configuration datastore. A valid 'create' operation attribute for a data node that has been set by a

client to its schema default value MUST fail with a 'data-exists' error-tag. A valid 'delete' operation attribute for a data node that has been set by a client to its schema default value MUST succeed.

[3.2.1.3](#). 'report-all' <edit-data> Behaviour

If the 'with-defaults' capability is supported by the server, the 'report-all' basic mode, defined in [section 3.2.1.1](#), is supported for <edit-data> operations that target conventional configuration datastores.

A valid 'create' operation attribute for a data node that has been set by a client to its schema default value MUST succeed. A valid 'delete' operation attribute for a data node that has been set by a client to its schema default value MUST fail with a 'data-missing' error-tag.

Since the default configuration doesn't exist in the conventional configuration datastore, the default configuration MUST be created without error being returned, irrespective of "with-defaults" parameter being set to basic-mode, trim or report-all.

[3.2.2](#). get/get-config Operation

For backwards compatibility consideration, when the basic mode is set to 'report-all' or 'with-defaults' capability parameter is set to report all, the NMDA aware server should return all the data based on filtering selection criteria including all the data from conventional configuration datastore and default configuration from operational state datastore.

[3.2.3](#). get-data Operation

When the basic mode is set to report-all or 'with-defaults' capability parameter is set to report all, the NMDA aware server should return all the data based on filtering selection criteria including all the data from conventional configuration datastore, but not include default configuration from operational state datastore unless they are explicitly set by the client.

[3.3.](#) Protocol Operation Clarification

[3.3.1.](#) <get>

As described in [[RFC6241](#)], the NETCONF <get> operation returns the contents of <running> together with the operational state.

If the client supports NMDA and sends <get> request to the NMDA aware server, the server should assume the client is non-NMDA client and retrieve specified configuration from running configuration and system configuration from operational state datastore and return it together with the system configuration to the client in <rpc-reply>.

If the client doesn't support NMDA and sends <get> request to the NMDA aware server, the server should retrieve running configuration and system configuration from operational state datastore and return the same results as it retrieves from non-NMDA aware server.

For default handling basic modes, please refer to [Section 3.2.2.](#)

[3.3.2.](#) <get-config>

As described in [[RFC6241](#)], the NETCONF <get-config> operation can be used to retrieve all or part of a specified configuration datastore.

If the client supports NMDA and sends <get-config> request to the NMDA aware Server, the server should assume the client is non-NMDA client and retrieve specified configuration from <running> together with the system configuration.

If the client doesn't support NMDA and send <get-config> request to the NMDA aware server, the server should retrieve <running> configuration together with the system configuration and return the same result as it retrieves from non-NMDA aware server.

For default handling basic modes, please refer to [Section 3.2.2.](#)

[3.3.3.](#) <edit-config>

As described in [[RFC6241](#)], the NETCONF <edit-config> operation can be used to load all or part of a specified configuration to the specified target configuration datastore.

If the client sends <edit-config> request to the NMDA aware server and wants to have further configuration based on the system configuration, (e.g., configure IP address within auto-created physical interface configuration), the server should create IP address configuration corresponding to specific physical interface without error to be returned to the client as long as IP address configuration is validated. The effect is the same as the physical interface has already been part of conventional configuration datastore. If the system configuration is set by the client sending <edit-config> operation request, the error should be returned as if the system configuration has already been part of conventional configuration datastore.

For default handling basic modes, please refer to [Section 3.2.1.2](#).

[3.3.4.](#) <get-data>

As described in [[I-D.ietf-netconf-nmda-netconf](#)], the <get-data> operation retrieves data from a specific NMDA datastore, similar to NETCONF's <get-config> operation defined in [[RFC6241](#)].

If the client sends <get-data> request with specified target configuration datastore set to the running datastore, the server should assume the client is NMDA client and retrieve specified configuration from <running> without system configuration set by the server since the system configuration is separated from conventional configuration datastore.

For default handling basic modes, please refer to [Section 3.2.3](#).

[3.3.5.](#) <edit-data>

As described in [[I-D.ietf-netconf-nmda-netconf](#)], the NETCONF <edit-data> operation can be used to load all or part of a specified configuration to the specified target configuration datastore.

For the NMDA client sending <edit-data> operation request with specified target configuration datastore set to the configuration datastore such as running datastore, since the system configuration is separated from conventional configuration datastore, if the NMDA client wants to use system configuration or configure other parameter (e.g., IP address) within system configuration (e.g., auto-created

interface configuration), Explicitly creating the system configuration by the client MUST be allowed without error being returned.

[4.](#) IANA Considerations

There is no IANA action in this document.

[5.](#) Security Considerations

This document does not introduce any security vulnerability besides one defined in [[RFC6241](#)] [[I-D.ietf-netconf-nmda-netconf](#)].

[6.](#) Acknowledgements

Thanks Robert Wilton,Guangying Zheng,Shouchuan Yang,Dan Qu, Ye Niu to discuss NMDA comability issues on existing protocol operation and provide important input to this document.

[7.](#) Contributors

Michael Wang
Huawei Technologies, Co., Ltd
101 Software Avenue, Yuhua District
Nanjing 210012
China

Email: wangzitao@huawei.com

[8.](#) References

[8.1.](#) 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, <<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>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", [RFC 7950](#), DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.

Internet-Draft

NMDA Backwards-Compatibility

March 2019

- [RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", [RFC 8342](#), DOI 10.17487/RFC8342, March 2018, <<https://www.rfc-editor.org/info/rfc8342>>.
- [RFC8407] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", [BCP 216](#), [RFC 8407](#), DOI 10.17487/RFC8407, October 2018, <<https://www.rfc-editor.org/info/rfc8407>>.

[8.2.](#) Informative References

- [I-D.ietf-netconf-nmda-netconf]
Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "NETCONF Extensions to Support the Network Management Datastore Architecture", [draft-ietf-netconf-nmda-netconf-08](#) (work in progress), October 2018.

Authors' Addresses

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: bill.wu@huawei.com

Chong Feng
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: frank.fengchong@huawei.com

