Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 3 July 2026

J. Dai
CICT
S. Yu
PCL
W. Cheng
China Mobile
M. Blanchet
Viagenie
P. Andersson
Ionio Systems

30 December 2025

NETCONF over QUIC
draft-ietf-netconf-over-quic-06

Abstract

This document specifies how to use QUIC as a secure transport for exchanging Network Configuration Protocol (NETCONF) messages. NETCONF over QUIC allows to take advantage of QUIC streams, for example, to eliminate some TCP head-of-line blocking issues. NETCONF over QUIC provides security properties similar to NETCONF over TLS.

This document also defines a YANG module which augments the ietf-netconf-client and ietf-netconf-server YANG modules.

Editorial note (to be removed by the RFC Editor

This draft contains placeholder values that need to be replaced with finalized values at the time of publication. This note summarizes all of the substitutions that are needed. No other RFC Editor instructions are specified elsewhere in this document.

Artwork in this document contains shorthand references to drafts in progress. Please apply the following replacements:

* AAAA --> the assigned RFC value for this draft

* BBBB --> the assigned RFC value for draft-ietf-netconf-netconf-client-server

* CCCC --> the assigned RFC value for draft-ietf-netconf-quic-client-server

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction 3
2. Terminology and Definitions 4
3. Connection Management 4
  3.1. Connection establishment 4
    3.1.1. Early data 4
  3.2. Connection Termination 5
    3.2.1. QUIC Connection Termination Process 5
    3.2.2. Considerations for Connection Termination 5
4. Stream mapping and usage 6
  4.1. Bidirectional Stream Between client and server 7
  4.2. Unidirectional Stream from server to client 7
  4.3. Mapping of QUIC connection, QUIC stream, and NETCONF session 7
5. Call home considerations 8
  5.1. protocol-layering perspective 8

5.2. RFC8071 Call Home Specific Case 9
6. Endpoint Authentication 9
6.1. Server Identity 9
6.2. Client Identity 9
7. Framing 9
8. Overview of YANG Module 9
8.1. The "netconf-client" augmentation 9
8.2. The "netconf-server" augmentation 10
9. YANG Module 10
10. Error codes 11
10.1. Transport Error Codes 11
10.2. Application Error Codes 11
11. Security Considerations 11
12. IANA Considerations 11
13. Acknowledgements 12
14. References 12
14.1. Normative References 12
14.2. Informative References 13
Authors' Addresses 14

1. Introduction

The Network Configuration Protocol (NETCONF) [RFC6241] defines a mechanism through which the configuration of network devices can be installed, manipulated, and deleted.

NETCONF can be conceptually partitioned into four layers: content, operation, message and security transport layers.

The Secure Transport layer provides a communication path between the client and server. NETCONF can be layered over any transport protocol that provides a set of basic requirements, such as:

1. NETCONF is connection-oriented, requiring a persistent connection between peers. This connection MUST provide reliable and sequenced data delivery. NETCONF connections are long-lived, persisting between protocol operations.

2. NETCONF connections MUST provide authentication, data integrity, confidentiality, and replay protection. NETCONF depends on the transport protocol for this capability.

The NETCONF protocol is not bound to any particular transport protocol, but allows a mapping to define how it can be implemented over any specific protocol.

However, because of the connection-oriented feature, almost all of the current secure transport protocols used by NETCONF are TCP based. As is well known, TCP has some shortcomings such as head-of-line blocking.

QUIC ([RFC9000][RFC9001]) conforms to the above requirements, therefore is also an appropriate transport protocol for NETCONF. Moreover, QUIC provides the following additional benefits not present in the other NETCONF transports:

* Single connection can be long lived and support multiple NETCONF RPC calls and responses within the same connection, using streams. This is very useful for a network management control station who is regularly monitoring devices and therefore having a long lived connection requires way less resources on both peers.

* 1 RTT initial handshake that includes TLS.

* Adaptable to more difficult environments such as those with long delays ([I-D.many-tiptop-usecase], [I-D.many-tiptop-quic-profile] .

Therefore, QUIC is a proper transport protocol for the secure transport layer of NETCONF. This document specifies how to use QUIC as the secure transport protocol for NETCONF.

2. Terminology and Definitions

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].

3. Connection Management

3.1. Connection establishment

QUIC connections are established as described in [RFC9000]. During connection establishment, support is indicated by selecting the ALPN token registered for NETCONF over QUIC (see Section 12) in the cryptographic handshake.

3.1.1. Early data

The QUIC protocol uses TLS 1.3 messages to secure the transport. This means that Early data (aka 0-RTT data) is supported. [RFC9001]

Early data (aka 0-RTT data) is a mechanism defined in TLS 1.3 [I-D.ietf-tls-rfc8446bis] that allows a client to send data ("early data") as part of the first flight of messages to a server. Note that TLS 1.3 can be used without early data as per Appendix F.5 of [I-D.ietf-tls-rfc8446bis]. In fact, early data is permitted by TLS 1.3 only when the client and server share a Pre-Shared Key (PSK), either obtained externally or via a previous handshake. The client uses the PSK to authenticate the server and to encrypt the early data.

As noted in Section 2.3 of [I-D.ietf-tls-rfc8446bis], the security properties for early data are weaker than those for subsequent TLS-protected data. In particular, early data is not forward secret, and there is no protection against the replay of early data between connections. Appendix E.5 of [I-D.ietf-tls-rfc8446bis] requires applications not use early data without a profile that defines its use. This document specifies that NETCONF over QUIC implementations MUST NOT use early data.

3.2. Connection Termination

3.2.1. QUIC Connection Termination Process

The typical QUIC connection termination process is described in [RFC9000]

3.2.2. Considerations for Connection Termination

When a NETCONF session is implemented based on a QUIC connection, the idle timeout should be set appropriately in order to keep the QUIC connection persistent even if the NETCONF session is idle. In some cases, disabling it may be a possible option.

When a NETCONF server receives a <close-session> request, it will gracefully close the NETCONF session. The server SHOULD close the associated QUIC connection.

When a NETCONF entity receives a <kill-session> request for an open session, it SHOULD close the associated QUIC connection.

When a NETCONF entity is detecting the interruption of the QUIC connection, it SHOULD send a <close-session> request to the peer NETCONF entity.

When a stateless reset event occurs, nothing needs to be done by either the client or the server.

4. Stream mapping and usage

The NETCONF protocol layers specified in [RFC6241] are presented in Figure 1.

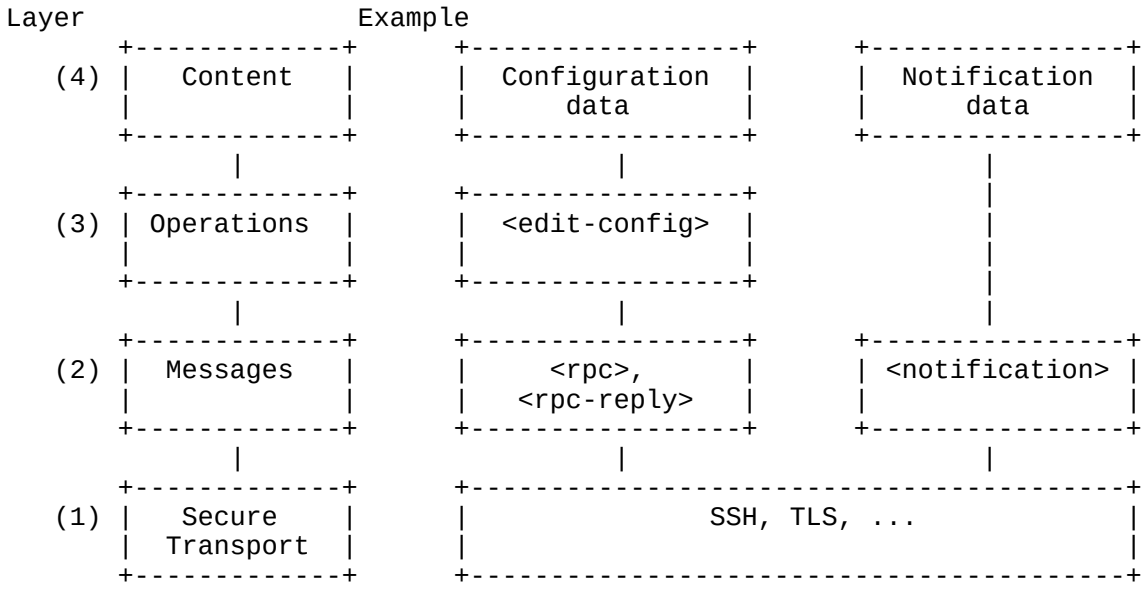

Figure 1: NETCONF Protocol Layers

Figure 1 shows that there are two kinds of main data flow exchanged between client and server:

* Configuration data from client to server.

* Notification data from server to client.

The two kinds of data flow need to be mapped into QUIC streams.

QUIC Streams provide a lightweight, ordered byte-stream abstraction to an application. Streams can be unidirectional or bidirectional meanwhile streams can be initiated by either the client or the server. Unidirectional streams carry data in one direction: from the initiator of the stream to its peer. Bidirectional streams allow for data to be sent in both directions.

QUIC uses Stream ID to identify the stream. The least significant bit (0x1) of the stream ID identifies the initiator of the stream. The second least significant bit (0x2) of the stream ID distinguishes between bidirectional streams (with the bit set to 0) and

unidirectional streams. There are four types of streams which are described in [RFC9000]. And Table 1 also describes the four types of streams

Acronym	Stream Type
C-BD	Client-Initiated, Bidirectional
S-BD	Server-Initiated, Bidirectional
C-UN	Client-Initiated, Unidirectional
S-UN	Server-Initiated, Unidirectional

Table 1: Stream Acronym Types

4.1. Bidirectional Stream Between client and server

NETCONF protocol uses an RPC-based communication model. Configuration data from client to server is exchanged based on '<rpc>' (the client initiating) and '<rpc-reply>' (sent by the server) and so on.

The messages used to exchange configuration data MUST be mapped into one bidirectional stream whose acronym is 'C-BD' according to Table 1. Since RPC processing is serialized and ordered within a session ([RFC6241] section 4.5), a bidirectional stream MUST be used for each NETCONF session.

4.2. Unidirectional Stream from server to client

There are some notification data exchanged between the client and the server. Notification is an server initiated message indicating that a certain event has been recognized by the server.

Notification messages are initiated by the server and no reply is needed from the client. So the messages used to exchange notification data MUST be mapped into one unidirectional stream whose acronym is 'S-UN' according to Table 1.

4.3. Mapping of QUIC connection, QUIC stream, and NETCONF session

The relationship among NETCONF sessions, QUIC streams and QUIC connections is illustrted as follows.

* One NETCONF session is allowed per QUIC connection.

* The NETCONF sessions, except subscriptions, runs over a QUIC bidi-stream.

* NETCONF Notifications and Subscribed Notifications runs over one QUIC uni-stream per subscription.

The notifications refer to messages corresonding to a subscription from server to client after the subscription process is over.

5. Call home considerations

5.1. protocol-layering perspective

The following diagram illustrates call home from a protocol-layering perspective based on QUIC:

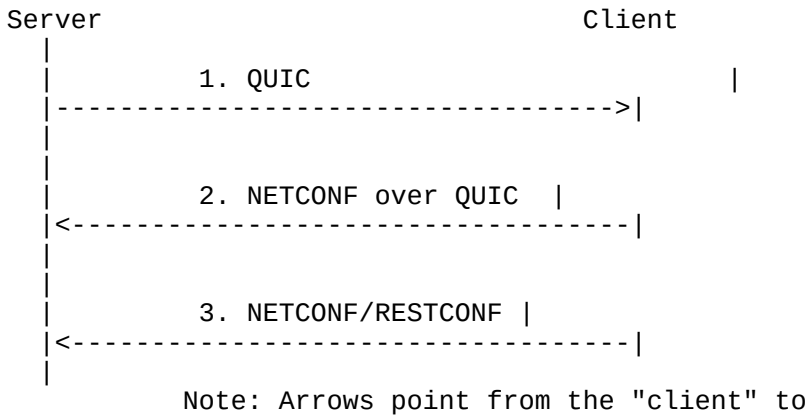

Figure 2: Call Home Sequence Diagram based on QUIC

This diagram makes the following points:

* The NETCONF/RESTCONF server begins by initiating a QUIC connection to the NETCONF/RESTCONF client.

* Using this QUIC connection, the NETCONF/RESTCONF client initiates a NETCONF/RESTCONF session to the NETCONF/RESTCONF server.

5.2. RFC8071 Call Home Specific Case

In the case of [RFC8071] Call home feature, where the NETCONF server initiates the transport connection to the NETCONF client, Table 1 will be used as follows: - the Client, referred in the Table, means the QUIC initiating party, therefore the NETCONF server and - the Server means the QUIC receiving party, therefore the NETCONF client.

6. Endpoint Authentication

Since QUIC uses TLS 1.3 this is used to verify server identity and client identity.

6.1. Server Identity

A server’s identity MUST be verified according to Section 6 of [RFC7589].

6.2. Client Identity

A client’s identity MUST be verified according to Section 7 of [RFC7589].

7. Framing

In order to mitigate delimiter injection attacks chunked framing as defined in [RFC6242] is required for NETCONF over QUIC.

The <hello> message MUST be followed by the character sequence RFC 5539 assumes that the end-of-message (EOM) sequence,]]>]]>>. Upon reception of the <hello> message, the receiving peer’s QUIC layer conceptually passes the <hello> message to the Messages layer. If the :base:1.1 capability is advertised by both peers, the chunked framing mechanism defined in Section 4.2 of [[RFC6242]] is used for the remainder of the NETCONF session. Otherwise, the old end-of-message-based mechanism (see Section 4.3 of [[RFC6242]]) is used.

8. Overview of YANG Module

This document defines one YANG module that augments the NETCONF YANG groupings [I-D.ietf-netconf-netconf-client-server] with the QUIC transport YANG groupings [I-D.ietf-netconf-quic-client-server]. This section presents an overview of the YANG Module.

8.1. The "netconf-client" augmentation

The following tree diagram [RFC8340] illustrates the augmentation of the QUIC client grouping into the NETCONF client container:

INSERT_TEXT_FROM_FILE(refs/tree-ietf-netconf-quic-client-augment.txt)

Figure 3

Comments:

* This augmentation to the "ncc:transport" container in "ietf-netconf-client.yang" adds a "quic" case with a "quic" container which uses the "quicc:quic-client" grouping.

* Note that the if-feature "quic-initiate" conditions if the "quic" container is available in the schema.

8.2. The "netconf-server" augmentation

The following tree diagram [RFC8340] illustrates the augmentation of the QUIC server grouping into the NETCONF server container:

INSERT_TEXT_FROM_FILE(refs/tree-ietf-netconf-quic-server-augment.txt)

Figure 4

Comments:

* This augmentation to the "ncs:transport" container in "ietf-netconf-server.yang" adds a "quic" case with a "quic" container which uses the "quics:quic-server" grouping.

* Note that the if-feature "quic-listen" conditions if the "quic" container is available in the schema.

9. YANG Module

This YANG module has normative references to [I-D.ietf-netconf-netconf-client-server] and [I-D.ietf-netconf-quic-client-server].

<CODE BEGINS> file "ietf-netconf-quic@YYYY-MM-DD.yang"

INSERT_TEXT_FROM_FILE(ietf-netconf-quic@YYYY-MM-DD.yang)

Figure 5

<CODE ENDS>

10. Error codes

10.1. Transport Error Codes

Error codes of secure transport layer are specified in [[RFC9000]]. There are not new transport Error Codes defined for NETCONF over QUIC.

10.2. Application Error Codes

According to [[RFC9000]], management of application error codes is left to application protocols. and application error codes can be used by RESET_STREAM Frame and STOP_SENDING Frame. Application error codes for NETCONF over QUIC are listed as follows:

* NO_NETCONF PROTOCOL ERROR (0x00): no NETCONF errors happens.

* NETCONF CLOSE SESSION_ERROR (0x01): The peer tries to close a session which is not initiated by it.

* NETCONF CLOSE STREAM ERROR (0x02): The peer tries to close a bidirectional stream when the NETCONF session is active.

11. Security Considerations

The security considerations described throughout [RFC8446] and [RFC6241] apply here as well. This document requires verification of server identity and client identity according to [RFC7589].

If invalid data or malformed messages are encountered, a robust implementation of this document MUST silently discard the message without further processing and then stop the NETCONF session.

12. IANA Considerations

This document creates a new registration for the identification of NETCONF over QUIC in the "Application Layer Protocol Negotiation (ALPN) Protocol IDs registry established in [RFC7301].

The "noq" string identifies NETCONF over QUIC:

* Protocol: NETCONF over QUIC

* Identification Sequence: 0x6e 0x6f 0x71 ("noq")

* Specification: This document

This document also requests IANA to reserve a UDP port for 'NETCONF over QUIC':

* Service Name: netconf-quic
* Transport Protocol(s): UDP
* Assignee: IESG iesg@ietf.org
* Contact: IETF Chair chair@ietf.org
* Description: NETCONF protocol over QUIC transport
* Reference: RFC AAAA
* Port number: 831
* Assignment Notes: Port 831 is currently assigned to netconf-beep, but a de-assignment is requested in [I-D.ietf-netconf-port-numbers].
* Service Name: netconf-ch-quic
* Transport Protocol(s): UDP
* Assignee: IESG iesg@ietf.org
* Contact: IETF Chair chair@ietf.org
* Description: NETCONF Call Home (QUIC)
* Reference: RFC AAAA
* Port number: 4335

13. Acknowledgements

The authors would like to acknowledge the contributors Yang Kou, Xueshun Wang, Kent Watsen, Jeffrey Haas, Bal□zs Lengyel, Robert Wilton, Huaimo Chen, Lifen Zhou, Andy Bierman, Sean Turner, and Joe Clarke for their beneficial comments.

The authors would like to acknowledge the very useful feedback from an early implementor: Adolfo Ochagavia.

14. References

14.1. Normative References

[I-D.ietf-netconf-netconf-client-server]
Watsen, K., "A YANG Data Model for NETCONF Clients and Servers", Work in Progress, Internet-Draft, draft-ietf-netconf-netconf-client-server-41, 4 December 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-netconf-client-server-41>.

[I-D.ietf-netconf-quic-client-server]
Andersson, P., "YANG Groupings for QUIC clients and QUIC servers", Work in Progress, Internet-Draft, draft-ietf-netconf-quic-client-server-03, 17 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-quic-client-server-03>.

[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>.

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

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

[RFC9001] Thomson, M., Ed. and S. Turner, Ed., "Using TLS to Secure QUIC", RFC 9001, DOI 10.17487/RFC9001, May 2021, <https://www.rfc-editor.org/info/rfc9001>.

14.2. Informative References

[I-D.ietf-netconf-port-numbers]
Boucadair, M., "Updates to NETCONF Transport Port Numbers", Work in Progress, Internet-Draft, draft-ietf-netconf-port-numbers-07, 16 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-port-numbers-07>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301, July 2014, <https://www.rfc-editor.org/info/rfc7301>.

[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, <https://www.rfc-editor.org/info/rfc7589>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8071] Watsen, K., "NETCONF Call Home and RESTCONF Call Home", RFC 8071, DOI 10.17487/RFC8071, February 2017, <https://www.rfc-editor.org/info/rfc8071>.

[I-D.many-tiptop-usecase] Blanchet, M., Eddy, W., and M. Eubanks, "IP in Deep Space: Key Characteristics, Use Cases and Requirements", Work in Progress, Internet-Draft, draft-many-tiptop-usecase-03, 18 June 2025, <https://datatracker.ietf.org/doc/html/draft-many-tiptop-usecase-03>.

[I-D.many-tiptop-quic-profile] Blanchet, M., "QUIC Profile for Deep Space", Work in Progress, Internet-Draft, draft-many-tiptop-quic-profile-01, 23 August 2025, <https://datatracker.ietf.org/doc/html/draft-many-tiptop-quic-profile-01>.

[I-D.ietf-tls-rfc8446bis] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", Work in Progress, Internet-Draft, draft-ietf-tls-rfc8446bis-14, 13 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-rfc8446bis-14>.

Authors' Addresses

Jinyou Dai
China Information Communication Technologies Group.
Gaoxin 4th Road 6#
Wuhan, Hubei 430079
China
Email: djy@fiberhome.com

Shaohua Yu
China PCL.
China
Email: yush@cae.cn

Weiqiang Cheng
China Mobile
China
Email: chengweiqiang@chinamobile.com

Marc Blanchet
Viagenie
Canada
Email: marc.blanchet@viagenie.ca

Per Andersson
Ionio Systems
Sweden
Email: per.ietf@ionio.se