SIDROPS
Internet-Draft
Intended status: Standards Track
Expires: 31 August 2026

S. Yue
China Mobile
C. Lin
New H3C Technologies
J. Roy
HPE Networking
D. Ma
ZDNS
Y. Liu
China Mobile

27 February 2026

RPKI to Router Protocol over QUIC
draft-liu-sidrops-rpki-rtr-over-quic-03

Abstract

The Resource Public Key Infrastructure (RPKI) to Router Protocol provides a simple but reliable mechanism to receive cryptographically validated RPKI prefix origin data and router keys from a trusted cache. RPKI to Router (RTR) Protocol can be carried over various transports such as TCP, SSH or else. QUIC provides practical and secure semantics for the RTR protocol, particularly fast connection establishment and multi-stream carrying, thereby reducing the time required to complete RTR data synchronization. This document describes how to use RTR Protocol over the QUIC transport protocol, named RTRoQUIC.

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 31 August 2026.

Table of Contents

1. Introduction 2
2. Terminology and Definitions 4
3. Connection Management 5
3.1. Connection Establishment 5
3.2. Connection Termination 6
3.2.1. QUIC Connection Termination Process 6
3.2.2. RTRoQUIC Considerations for Connection Termination . 6
4. Stream Mapping and Usage 6
4.1. Multiple Stream Usage 7
4.1.1. Control Channel 7
4.1.2. Data Channel 7
4.1.3. Use Case 9
5. Endpoint Authentication 10
6. Operational Considerations 10
7. IANA Considerations 10
8. Security Considerations 11
9. References 11
9.1. Normative References 11
9.2. Informative References 12
Authors' Addresses 12

1. Introduction

In order to verifiably validate the origin Autonomous Systems (ASes) and AS paths of BGP announcements, the Resource Public Key Infrastructure (RPKI) to Router Protocol [RFC8210] provides a simple but reliable mechanism to receive cryptographically validated RPKI [RFC6480] prefix origin data and router keys from a trusted cache.

The transport-layer session between a router and a cache carries the binary Protocol Data Units (PDUs) in a persistent session. To prevent cache spoofing and DoS attacks by illegitimate routers, it is

highly desirable that the router and the cache be authenticated to each other. Integrity protection for payloads is also desirable to protect against Man-in-the-Middle (MITM) attacks.

The RPKI to Router (RTR) Protocol is not bound to any particular transport protocol, but allows a mapping to define how it could be implemented over any specific transport protocol. At present, some secure transport protocols are defined to carry RTR Protocol: TCP-A0 transport [RFC5925], Secure Shell version 2 (SSHv2) transport [RFC4252], TCP MD5 transport [RFC5925], TCP over IPsec transport [RFC4301], and Transport Layer Security (TLS) transport [RFC8446]. However, because of the connection-oriented feature, almost all of the current secure transport protocols used by RTR Protocol is TCP based. As is well known, TCP has some shortcomings such as head-of-line blocking.

QUIC [RFC9000] is a UDP-based multiplexed and secure transport protocol that provides connection-oriented and stateful interaction between a client and server. It can provide low latency and encrypted transport with resilient connections.

QUIC uses multiple simultaneous streams to carry data in one direction. Each stream is a separate unidirectional or bidirectional channel consisting of an ordered stream of bytes. In Addition, each stream has its own flow control, which limit bytes sent on a stream, together with flow control of the connection.

Compared with the current secure transport protocols used by RTR Protocol, QUIC offers the following advantages:

* Higher-speed connection establishment and restoration.

  QUIC integrates TLS 1.3 into its core. For servers with which it has previously connected, the client can carry application data (such as an RTR session request) in the first packet, achieving "0-RTT" recovery. Even for the first connection, only one RTT is needed to complete the cryptographic handshake. In contrast, TCP+TLS requires at least 2-3 RTTs.

  This allows RTR caching servers or routers to almost instantly resume synchronization with the RPKI verification database after restarting or switching networks, greatly reducing the "blank period" of routing verification information caused by synchronization delays and improving the resilience and response speed of routing security.

* Completely eliminate TCP head-of-line blocking and improve transmission efficiency.

QUIC implements multiple independent "streams" on top of UDP. In an RTR context, different PDUs can be mapped to different QUIC streams. Packet loss in one stream only affects the retransmission of that stream; data in other streams can continue to be processed and parsed by the application layer.

In network environments with a certain packet loss rate, the overall completion time for RTR data synchronization will be shorter and more predictable.

* Enhanced connectivity migration and network resilience.

  QUIC connections use a connection ID instead of the traditional four-tuple (source/destination IP, port) for identification. When the client's network address changes (e.g., during router failover or mobile network switching), the QUIC connection can remain intact as long as the connection ID remains the same.

  RTR sessions can continue uninterrupted after a network layer failover, avoiding TCP connection drops and re-handshakes caused by IP address changes, thus further ensuring the continuity of RPKI data synchronization.

* Built-in security.

  QUIC's encryption and authentication are mandatory and built-in. This eliminates any risk of misconfiguration leading to plaintext transmission.

Therefore, QUIC is the appropriate and secure transport protocol choice for the RTR protocol message transmission mechanism. This document specifies how to use QUIC as the secure transport protocol for RTR Protocol.

2. Terminology and Definitions

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.

In this document, the terms "client" and "server" are used to refer to the two ends of the QUIC connection. The client actively initiates the QUIC connection. The terms "router" and "cache" are used to refer to the two ends of the RTR Protocol session. A router establishes and keeps open a connection to one or more caches, generally, a "router" is a "client" meanwhile a "cache" is a "server".

* Client: The endpoint that initiates a QUIC connection, the RTR router.

* Server: The endpoint that accepts a QUIC connection, the RTR cache.

3. Connection Management

3.1. Connection Establishment

QUIC connection establishment is described in [RFC9000]. During establishing connection, RTRoQUIC support is indicated by selecting the Application-Layer Protocol Negotiation (ALPN) [RFC7301] token as listed in the IANA Section 7 in the TLS handshake.

The router MUST also act as the client meanwhile the cache must also act as the server.

The router should be the initiator of the QUIC connection to the cache meanwhile the cache acts as a connection acceptor.

The QUIC protocol uses TLS 1.3 messages to secure the transport. TLS 1.3 supports Early Data (also known as 0-RTT data) [RFC9001], and TLS 1.3 can be used without Early Data. Early Data provides weaker security than standard TLS, lacking forward secrecy and replay attack protection. Because RPKI data is used to verify the legitimacy of BGP route advertisements, any tampering, forgery, or leakage could lead to large-scale route hijacking, traffic disruption, or eavesdropping. Therefore, RPKI data security requirements are extremely high.

This document specifies that RTRoQUIC implementations MUST NOT utilize Early Data (0-RTT). Clients MUST NOT include early_data extensions in ClientHello messages, and servers MUST reject such extensions if presented. Implementations MUST configure their TLS 1.3 stacks to disable 0-RTT functionality.

3.2. Connection Termination

3.2.1. QUIC Connection Termination Process

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

3.2.2. RTRoQUIC Considerations for Connection Termination

When a RTR session is implemented based on a QUIC connection, the idle timeout should be disabled or the QUIC max_idle_timeout should be set appropriately in order to keep the QUIC connection persistent even if the RTR session is idle.

When the cache and router support different versions, the checker should close the RTR session and the associated QUIC connection.

When a router or cache is detecting the interruption of the QUIC connection, it SHOULD terminate the connection.

4. Stream Mapping and Usage

Currently, there are eleven kinds of RTR Protocol Data Units (PDUs) exchanged between the cache and the router, namely Serial Notify PDU, Serial Query PDU, Reset Query PDU, Cache Response PDU, IPv4 Prefix PDU, IPv6 Prefix PDU, End of Data PDU, Cache Reset PDU, Router Key PDU, Error Report PDU and ASPA PDU [I-D.ietf-sidrops-8210bis]. The eleven kinds of RTR PDUs need to be mapped into QUIC streams.

QUIC [RFC9000] is a UDP-based multiplexed and secure transport protocol that provides connection-oriented and stateful interaction between a client and server. It can provide low latency and encrypted transport with resilient connections.

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 (client with the bit set to 0). The second least significant bit (0x2) of the stream ID distinguishes between bidirectional streams (with the bit set to 0) and unidirectional streams [RFC9000].

Since there are PDUs from Cache to Router and PDUs from Router to Cache, all RTR PDUs can be simply mapped into one bidirectional QUIC stream whose stream type is 0x0 according to section 2.1 of [RFC9000]. The bidirectional stream SHOULD be created immediately by the Router after the connection is established between the Router and the Cache. And the bidirectional stream MUST be destroyed when the connection is terminated.

Additionally, to improve transmission efficiency, RTR PDUs also can be mapped into multiple QUIC streams, with the stream type determined by the transmitted PDUs.

4.1. Multiple Stream Usage

According to the functional characteristics of RTR PDUs, RTR PDUs can be divided into two categories: Data PDUs and Control PDUs. Therefore, the transmission of RTR PDUs can occur through two types of channels: Data Channel and Control Channel.

4.1.1. Control Channel

The Control PDUs currently include Serial Notify PDU, Serial Query PDU, Reset Query PDU, Cache Reset PDU and Error Report PDU. Some of these PDUs are initiated by the cache and some are sent by the router. So this PDUs MAY be mapped into one bidirectional stream whose stream type is 0x0 according to section 2.1 of [RFC9000]. The bidirectional control stream between Cache and Router is called the Control Channel. The Control Channel always uses QUIC stream 0, which is a client-initiated bidirectional stream.

The Control Channel MUST be created immediately by the Router, after the connection establishment is done between the Router and the Cache. The Control Channel also MUST be destroyed when the connection is closed.

4.1.2. Data Channel

Among all RTR PDUs, Cache Response PDU is always followed by payload PDUs and an End of Data PDU. And the payload PDUs include IPv4 Prefix PDU, IPv6 Prefix PDU, Router Key PDU and ASPA PDU. So Data PDUs include the Cache Response PDU, the payload PDUs and the End of Data PDU, which are from the cache (server) to the router (client), and these PDUs are ordered. The stream for transmitting these Data PDUs is called the Data Channel. For the order of Data PDUs, Cache Response PDU and End of Data PDU MUST be sent to all Data Channels (streams).

According to above information, All Data PDUs are initiated by the cache and no reply is needed from the router, so Data PDUs can be mapped into one or more unidirectional stream whose stream type is 0x3 according to section 2.1 of [RFC9000]. As shown in the figure 1, these unidirectional data channels SHOULD be created by the Cache, before the Cache sends the Cache Response PDU which is the first of all data channels. And the number of data channel SHOULD be controlled by the Cache through configuration for actual performance requirements. Furthermore, the data channels MUST be closed when the connection is terminated.

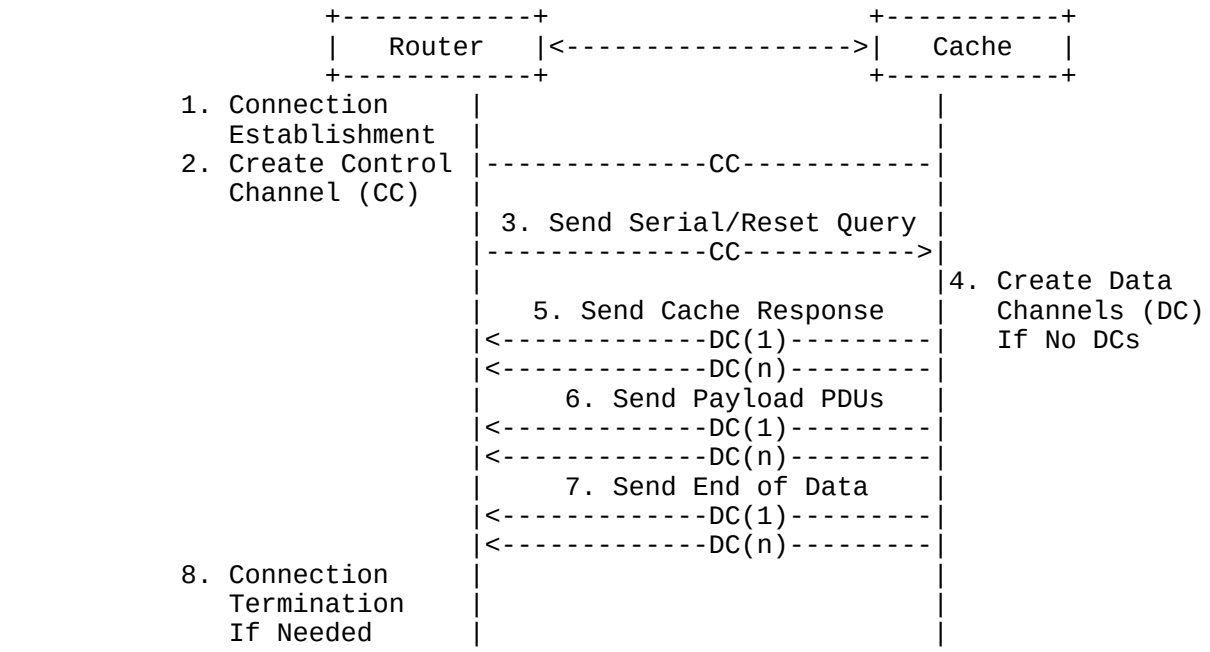

Figure 1: Usage of Unidirectional Data Channels

When receiving Data PDUs via n unidirectional Data Channels as shown in the figure 1, the first received Cache Response PDU is considered a valid PDU, and the last received End of Data PDU is considered a valid PDU.

To improve operational flexibility, the Data PDUs also can be mapped into one or more bidirectional stream which is created by the Router, and the stream type is 0x0 according to section 2.1 of [RFC9000]. As shown in the figure 2, these bidirectional data channels SHOULD be created before the Router sends the Serial or Reset Query over the Control Channel. And the number of data channel also SHOULD be configured by the Router according to actual performance

requirements. In addition, the bidirectional data channels could be closed when the Router receives the End of Data which is the last of all Data Channels.

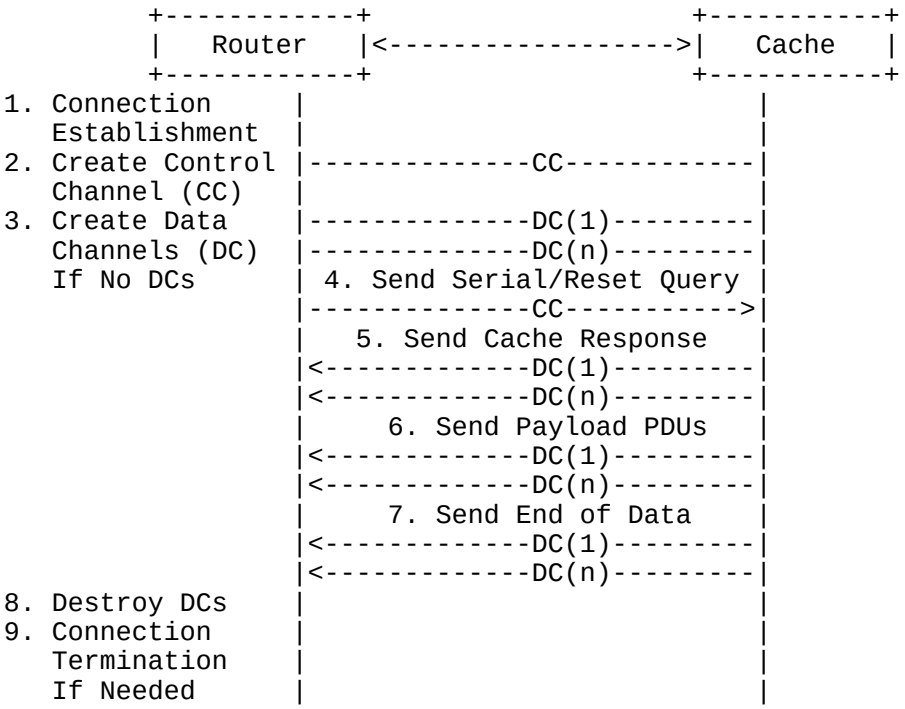

Figure 2: Usage of Bidirectional Data Channels

Also when receiving Data PDUs via n bidirectional Data Channels as shown in the figure 2, the first received Cache Response PDU is considered a valid PDU, and the last received End of Data PDU is considered a valid PDU.

4.1.3. Use Case

Since there are four types of payload PDUs, namely IPv4 Prefix PDU, IPv6 Prefix PDU, Router Key PDU and ASPA PDU, four data channels (streams) can be created to carry these four types of PDUs respectively, as shown in the figure 3.

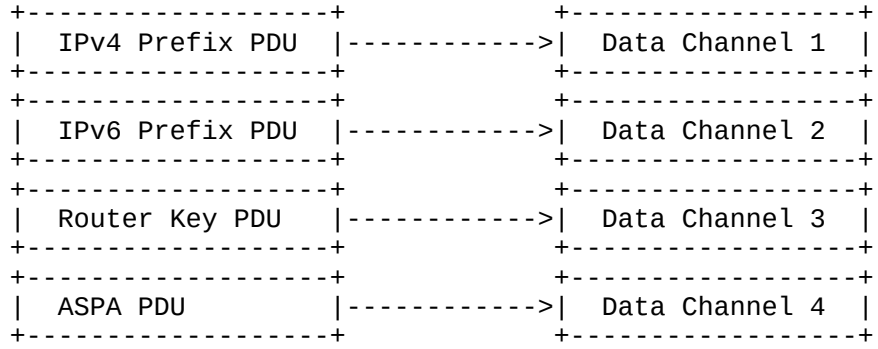

Figure 3: Use Case

5. Endpoint Authentication

RTRoQUIC uses QUIC which uses TLS version 1.3 or greater. Therefore, the TLS handshake process can be used for RTRoQUIC endpoint authentication. A third-party authentication mechanism can also be applied for RTRoQUIC endpoint authentication, such as a TLS client certificate.

6. Operational Considerations

The decision to use RTRoQUIC instead of the TCP-based mechanism in [RFC8210] is an operational decision, and an implementation MUST provide a configuration mechanism to enable RTRoQUIC on the RTR session.

Some connectivity problems (such as blocking UDP) could result in a failure to establish a QUIC connection. When this happens, the router SHOULD attempt to establish a TCP-based RTR session.

7. IANA Considerations

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

The "RTRoQ" string identifies RTRoQUIC:

* Protocol: RTRoQUIC

* Identification Sequence: 0x52 0x54 0x52 0x6f 0x51 ("RTRoQ")

* Specification: This document

8. Security Considerations

This document replaces the transport protocol layer of RTR from TCP to QUIC. The basic protocol specification of RTR is not modified, and therefore the new security risks are not introduced to the basic RTR protocol. RTRoQUIC enhances transport-layer security for RTR session according to [RFC9000].

This document does not require to support third-party authentication (e.g., backend Authentication) due to the fact that TLS does not specify this way of authentication. If third-party authentication is needed, TLS client certificates are recommended to be used here.

9. References

9.1. Normative References

[I-D.ietf-sidrops-8210bis]
Bush, R., Austein, R., and T. Harrison, "The Resource Public Key Infrastructure (RPKI) to Router Protocol, Version 2", Work in Progress, Internet-Draft, draft-ietf-sidrops-8210bis-24, 5 February 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-sidrops-8210bis-24>.

[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>.

[RFC4252] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Authentication Protocol", RFC 4252, DOI 10.17487/RFC4252, January 2006, <https://www.rfc-editor.org/info/rfc4252>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <https://www.rfc-editor.org/info/rfc4301>.

[RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, DOI 10.17487/RFC5925, June 2010, <https://www.rfc-editor.org/info/rfc5925>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8210] Bush, R. and R. Austein, "The Resource Public Key Infrastructure (RPKI) to Router Protocol, Version 1", RFC 8210, DOI 10.17487/RFC8210, September 2017, <https://www.rfc-editor.org/info/rfc8210>.

[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>.

[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>.

9.2. Informative References

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480, February 2012, <https://www.rfc-editor.org/info/rfc6480>.

[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>.

Authors' Addresses

Shengnan Yue
China Mobile
China
Email: yueshengnan@chinamobile.com

Changwang Lin
New H3C Technologies
Beijing
China
Email: linchangwang.04414@h3c.com

Jishnu Roy
HPE Networking
1133 Innovation Way
Sunnyvale, CA 94089
United States of America
Email: jishnu.roy@hpe.com

Di Ma
ZDNS
Floor 21, Block B, Greenland Center
Chaoyang Beijing, 100102
China
Email: madi@zdns.cn

Yisong Liu
China Mobile
Beijing
China
Email: liuyisong@chinamobile.com