Delay/Disruption Tolerant Networking
Internet-Draft
Intended status: Standards Track
Expires: 20 September 2026

R. Taylor
E. Kline
Aalyria Technologies

19 March 2026

# DTN QUIC Bundle Protocol Convergence Layer (qubicle)
# draft-ek-dtn-qubicle-01

## Abstract

This document specifies a minimal convergence layer protocol for transferring Bundle Protocol version 7 (BPv7) bundles over QUIC. The protocol leverages QUIC's native capabilities for reliable streaming, connection management, and security, requiring no application-layer framing for reliable transfers. Unreliable transfers use the Bundle Transfer Protocol - Unidirectional (BTP-U) over QUIC datagrams.

## About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://ekline.github.io/draft-dtn-qubicle/draft-ek-dtn-qubicle.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ek-dtn-qubicle/.

Discussion of this document takes place on the Delay/Disruption Tolerant Networking Working Group mailing list (mailto:dtn@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/dtn/. Subscribe at https://www.ietf.org/mailman/listinfo/dtn/.

Source for this draft and an issue tracker can be found at https://github.com/ekline/draft-dtn-qubicle.

## 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/.

## Copyright Notice



## Table of Contents

## 1. Introduction

Bundle Protocol version 7 (BPv7) [RFC9171] requires Convergence Layer Adapters (CLAs) to transfer bundles between nodes. This document specifies the QUIC Bundle Protocol Convergence Layer (QBCL or "qubicle"), a minimal CLA using QUIC [RFC9000] that embraces QUIC's native capabilities rather than layering additional protocol machinery.

The design philosophy is simple: QUIC already provides reliable streams, multiplexing, flow control, congestion control, and integrated security. This specification adds only what is strictly necessary to transfer bundles.

The protocol provides two services:

Reliable Service: Bundles are transferred on QUIC streams with guaranteed delivery.

Unreliable Service: Bundles are transferred via QUIC datagrams [RFC9221] using [BTP-U] framing.

## 2. Conventions 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.

Client: The Qubicle peer that initiates the QUIC connection. This is a connection-level role and does not imply any restriction on bundle transfer direction.

Server: The Qubicle peer that accepts the QUIC connection. This is a connection-level role and does not imply any restriction on bundle transfer direction.

Qubicle Session: The period during which a QUIC connection is

established between two Qubicle peers. A session begins when the QUIC handshake completes and ends when the QUIC connection closes. Both client and server are equal peers for the purpose of bundle transfer.

## 3. Applicability Statement

QBCL SHOULD NOT be used in deployments where the QUIC transport may not perform well. It is primarily targeted to deployments where round-trip times remain under a few seconds, making QUIC's 1-RTT handshake overhead negligible relative to data transfer time. For extremely high-delay or distrupted environments such as deep space communications (e.g., Earth-Mars links with multi-minute RTTs), any handshake may represent significant absolute delay, and specialized protocols like LTP ({?RFC5326}) may be more appropriate.

Similarly, the SVCB-based DNS service discovery mechanism ({<dns-example}) SHOULD NOT be used in environments where DNS itself might not perform well. DNS-based discovery is NOT RECOMMENDED for use in DTN environments where DNS infrastructure is unavailable, network disruptions cause failed lookups or stale cached records, DNSSEC validation fails due to a mismatch between query RTT and valid signature lifetimes, or DNS query overhead is significant relative to available bandwidth. For such environments, implementations SHOULD support alternative CL provisioning mechanisms including manual configuration with pre-planned contact schedules, contact graph routing protocols that maintain topology independently of DNS, or out-of-band metadata distribution through mission management plane channels. A hybrid approach is RECOMMENDED for nodes bridging Internet and deep-space networks: use QBCL with DNS discovery for Internet-side connections, and use alternate mission management planes for space-side connections.

## 4. Protocol Overview

### 4.1. Connection Establishment

A Qubicle session is established by initiating a QUIC connection to a peer. The QUIC handshake provides mutual authentication via TLS 1.3 [RFC9001].

The ALPN identifier for Qubicle is qbcl.

### 4.2. Reliable Bundle Transfer

For reliable transfer, each bundle is sent on a dedicated QUIC unidirectional stream:

1. The sender creates a new unidirectional stream.

2. The sender writes the complete bundle (CBOR-encoded per [RFC9171]) to the stream.

3. The sender closes the stream by sending a STREAM frame with the FIN bit set.

The bundle is implicitly framed by the stream boundaries. No length prefix or application-layer framing is required.

The receiver reads data from the stream until FIN is received, then delivers the complete bundle to the BPA. Receipt of more than one Bundle on a given stream is a protocol error, and the receiver MUST abort the connection with QBCL_PROTOCOL_ERROR.

QUIC guarantees reliable, in-order delivery of stream data. No application-layer acknowledgment is required; the sender can consider the transfer complete when QUIC confirms the stream data has been acknowledged by the peer. Completed transfer at the convergence layer does not guarantee successful receipt at the receiving Bundle Protocol Agent, so this signal alone does not suffice to indicate when a Bundle can be deleted from the sender. Additional information at the Bundle Protocol layer is required to confirm successful transfer.

#### 4.2.1. Bidirectional Bundle Flow

Both peers can send bundles simultaneously. Each peer creates unidirectional streams to send its bundles. QUIC stream IDs inherently separate client-initiated streams (IDs 2, 6, 10...) from server-initiated streams (IDs 3, 7, 11...), ensuring no collision between the two directions of bundle flow.

#### 4.2.2. Stream Selection and Priority

Senders MAY use QUIC stream priorities to expedite higher-priority bundles. The mapping of bundle priority to QUIC stream priority is an implementation matter.

#### 4.2.3. Stream Exhaustion

QUIC stream identifiers are 62-bit values, providing an effectively unlimited number of streams per connection. The MAX_STREAMS transport parameter limits concurrent streams, not the total number of streams over a connection’s lifetime.

If an implementation reaches practical limits on stream creation, it SHOULD close the connection and establish a new one.

### 4.3. Unreliable Bundle Transfer

For unreliable transfer, bundles are sent using QUIC datagrams [RFC9221] with [BTP-U] framing.

Each QUIC datagram contains one or more [BTP-U] messages. The [BTP-U] specification defines segmentation, reassembly, transfer identification, and optional repetition for probabilistic reliability.

Implementations MUST negotiate the QUIC max_datagram_frame_size transport parameter to enable datagram support.

The mapping of bundle priority to [BTP-U] transfer interleaving is an implementation matter.

### 4.4. Connection Termination

To terminate a session, a peer closes the QUIC connection using CONNECTION_CLOSE. Application-specific error codes are defined in Section 5.

A peer MAY close the connection at any time. In-flight reliable transfers on incomplete streams will fail; the BPA is notified of the failure.

### 4.5. Transfer Cancellation

TODO(ek): describe how a receiver can cancel a transfer via STOP_SENDING.

### 4.6. Keepalive

Qubicle relies on QUIC’s native idle timeout mechanism. Peers negotiate the max_idle_timeout transport parameter during connection establishment.

If application-layer liveness detection is required, implementations MAY send QUIC PING frames.

## 5. Error Codes

The following application error codes are defined for use with QUIC CONNECTION_CLOSE:

| Code | Name | Description |
|---|---|---|
| 0x00 | QBCL_NO_ERROR | Graceful closure, no error |
| 0x01 | QBCL_PROTOCOL_ERROR | Qubicle protocol error encountered |

Table 1: Qubicle Error Codes

## 6. Security Considerations

### 6.1. Transport Security

QUIC mandates TLS 1.3 for all connections, providing confidentiality, integrity, and authentication. Qubicle inherits these security properties.

Implementations SHOULD require peer certificate authentication. The Node ID in the transport parameter SHOULD match an identity in the peer's certificate. The BundleEID OtherName form defined in [RFC9174], Section 4.4.2 provides a standard mechanism for embedding DTN Node IDs in X.509 certificates. Automated certificate provisioning is available via the ACME extensions defined in [RFC9891].

### 6.2. Bundle Security

Transport security protects bundles in transit between adjacent nodes. For end-to-end bundle security, implementations SHOULD use BPSec [RFC9172].

### 6.3. Denial of Service

QUIC provides built-in protection against many denial-of-service attacks, including address validation and amplification prevention.

Implementations SHOULD apply rate limiting on bundle reception to prevent resource exhaustion.

### 6.4. 0-RTT Considerations

QUIC 0-RTT data is subject to replay attacks. Implementations that enable 0-RTT SHOULD only send bundles that are safe to replay (e.g., bundles with replay protection at the bundle layer).

## 7. Operational Considerations

### 7.1. Version Negotiation

Qubicle endpoints wishing to combat various ossification vectors are RECOMMENDED to support version negotiation and the same Bundle transfer operations described in this memo over QUIC v2 [RFC9369].

### 7.2. Convergence Layer Fallback

As noted in [RFC9308], some networks block UDP traffic such that Qubicle connections cannot be established. Bundle Protocol Agents that employ Qubicle are RECOMMENDED to support additional Convergence Layers, e.g. TCPCLv4 [RFC9174].

### 7.3. Coexistence With Other UDP-based Convergence Layers

It is RECOMMENDED that Qubicle implementations use a dedicated UDP port for operational simplicity.

Bundle Protocol Agents that employ Qubicle and other UDP-based Convergence Layers on the same UDP port MUST be able to disambiguate received datagrams in order to route them to the correct CLA. For UDP CLs that use DTLS, [RFC9443] provides the required guidance to disambiguate QUIC traffic from DTLS-encapsulated CL traffic.

### 7.4. Finding a Qubicle Endpoint Via DNS

{#dns-example}

Qubicle senders may be manually provisioned with a hostname (or IP addresses) and UDP port corresponding to the listening Qubicle endpoint for a peer Bundle Protocol Agent. If only a hostname is known but a port is not, [RFC9460] SVCB Resource Records may be looked up to find a listening UDP port and confirm expected ALPN configuration.

Consider this zone file for example.:

```
;; zone: example.
;
$ORIGIN example.
_dtn-bundle._tcp.mars-orbiter IN SRV 10 20 4556 cloud-agent.example.
_qbcl.mars-orbiter IN SVCB 0 cloud-agent.example.

cloud-agent IN A    192.0.2.1
cloud-agent IN AAAA 2001:db8::1
cloud-agent IN SVCB 10 . (
    ipv4hint=192.0.2.1
    ipv6hint=2001:db8::1
    port=1234 alpn="qbcl")
```

A BPA supporting both [RFC9174] may attempt to resolve an SRV record for the _dtn-bundle._tcp prefixed hostname. A BPA that support Qubicle might also issue DNS SVCB queries for the [AttrLeaf] prefix "_qbcl". The sample above indicates that mars-orbiter.example. has an SVCB record in AliasMode referring to cloud-agent.example. The SVCB record associated with cloud-agent.example. contains all required QUIC transport rendezvous information.

# 8. IANA Considerations

## 8.1. ALPN Identifier

IANA is requested to register the following ALPN identifier in the "TLS Application-Layer Protocol Negotiation (ALPN) Protocol IDs" registry:

| Protocol | Identification Sequence | Reference |
|---|---|---|
| Qubicle | 0x71 0x62 0x63 0x6C ("qbcl") | This document |

Table 2: ALPN Registration

## 8.2. AttrLeaf Node Name

Per [AttrLeaf], IANA is request to add the following entry to the DNS "Underscored and Globally Scoped DNS Node Names" registry:

| RR Type | _NODE NAME | Reference |
|---|---|---|
| SVCB | _qbcl | this document |

Table 3: AttrLeaf Registration

### 8.3. Application Error Codes

IANA is requested to create a new registry "Qubicle Error Codes" with the following initial values:

| Code | Name | Reference |
|---|---|---|
| 0x00 | QBCL_NO_ERROR | This document |
| 0x01 | QBCL_PROTOCOL_ERROR | This document |
| 0x02-0xEF | Unassigned | |
| 0xF0-0xFF | Reserved for Private Use | This document |

Table 4: Error Code Registry

## 9. References

### 9.1. Normative References

[AttrLeaf] Crocker, D., "Scoped Interpretation of DNS Resource Records through "Underscored" Naming of Attribute Leaves", BCP 222, RFC 8552, DOI 10.17487/RFC8552, March 2019, <https://www.rfc-editor.org/rfc/rfc8552>.

[BTP-U] Taylor, R., "Bundle Transfer Protocol - Unidirectional", Work in Progress, Internet-Draft, draft-ietf-dtn-btpu-02, 17 February 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-dtn-btpu-02>.

[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/rfc/rfc2119>.

[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/rfc/rfc8174>.

[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/rfc/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/rfc/rfc9001>.

[RFC9171] Burleigh, S., Fall, K., and E. Birrane, III, "Bundle Protocol Version 7", RFC 9171, DOI 10.17487/RFC9171, January 2022, <https://www.rfc-editor.org/rfc/rfc9171>.

[RFC9172] Birrane, III, E. and K. McKeever, "Bundle Protocol Security (BPSec)", RFC 9172, DOI 10.17487/RFC9172, January 2022, <https://www.rfc-editor.org/rfc/rfc9172>.

[RFC9174] Sipos, B., Demmer, M., Ott, J., and S. Perreault, "Delay-Tolerant Networking TCP Convergence-Layer Protocol Version 4", RFC 9174, DOI 10.17487/RFC9174, January 2022, <https://www.rfc-editor.org/rfc/rfc9174>.

[RFC9221] Pauly, T., Kinnear, E., and D. Schinazi, "An Unreliable Datagram Extension to QUIC", RFC 9221, DOI 10.17487/RFC9221, March 2022, <https://www.rfc-editor.org/rfc/rfc9221>.

[RFC9369] Duke, M., "QUIC Version 2", RFC 9369, DOI 10.17487/RFC9369, May 2023, <https://www.rfc-editor.org/rfc/rfc9369>.

[RFC9443] Aboba, B., Salgueiro, G., and C. Perkins, "Multiplexing Scheme Updates for QUIC", RFC 9443, DOI 10.17487/RFC9443, July 2023, <https://www.rfc-editor.org/rfc/rfc9443>.

[RFC9460] Schwartz, B., Bishop, M., and E. Nygren, "Service Binding and Parameter Specification via the DNS (SVCB and HTTPS Resource Records)", RFC 9460, DOI 10.17487/RFC9460, November 2023, <https://www.rfc-editor.org/rfc/rfc9460>.

## 9.2. Informative References

[RFC9308] K端hlewind, M. and B. Trammell, "Applicability of the QUIC Transport Protocol", RFC 9308, DOI 10.17487/RFC9308, September 2022, <https://www.rfc-editor.org/rfc/rfc9308>.

[RFC9891] Sipos, B., "Automated Certificate Management Environment (ACME) Delay-Tolerant Networking (DTN) Node ID Validation Extension", RFC 9891, DOI 10.17487/RFC9891, November 2025, <https://www.rfc-editor.org/rfc/rfc9891>.

## Authors' Addresses

Rick Taylor
Aalyria Technologies
Email: rtaylor@aalyria.com

Erik Kline
Aalyria Technologies
Email: ek.ietf@gmail.com