

QUIC Working Group
Internet-Draft
Intended status: Experimental
Expires: February 13, 2021

M. Piraux
O. Bonaventure
UCLouvain
A. Masputra
Apple Inc.
August 12, 2020

Tunneling Internet protocols inside QUIC
draft-piraux-quic-tunnel-03

Abstract

This document specifies methods for tunneling Ethernet frames and Internet protocols such as TCP, UDP, IP and QUIC inside a QUIC connection.

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 February 13, 2021.

Copyright Notice

Copyright (c) 2020 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 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
2.	Conventions and Definitions	4
3.	Reference environment	4
4.	The lightweight mode	5
5.	The datagram mode	6
5.1.	Joining a tunneling session	7
5.1.1.	Coordinate use of the Packet Tag	8
6.	Connection establishment	9
7.	Messages format	9
7.1.	QUIC tunnel control TLVs	10
7.1.1.	Access Report TLV	10
7.1.2.	New Session TLV	11
7.1.3.	Session ID TLV	12
7.1.4.	Join Session TLV	12
8.	Security Considerations	13
8.1.	Privacy	13
8.2.	Ingress Filtering	13
9.	IANA Considerations	13
9.1.	Registration of QUIC tunnel Identification String	13
9.2.	QUIC tunnel control TLVs	14
9.2.1.	QUIC tunnel control TLVs Types	14
9.3.	QUIC tunnel control Error Codes	14
9.4.	QUIC tunnel Access Report Signal Codes	15
10.	References	15
10.1.	Normative References	15
10.2.	Informative References	15
Appendix A.	Change Log	17
A.1.	Since draft-piroux-quic-tunnel-02	17
A.2.	Since draft-piroux-quic-tunnel-01	17
A.3.	Since draft-piroux-quic-tunnel-00	17
	Acknowledgments	17
	Authors' Addresses	18

[1.](#) Introduction

Mobile devices such as laptops, smartphones or tablets have different requirements than the traditional fixed devices. These mobile devices often change their network attachment. They are often attached to trusted networks, but sometimes they need to be connected to untrusted networks where their communications can be eavesdropped, filtered or modified. In these situations, the classical approach is

to rely on VPN protocols such as DTLS or IPSec. These VPN protocols provide the encryption and authentication functions to protect those mobile clients from malicious behaviors in untrusted networks.

However, some networks have deployed filters that block these VPN protocols. When faced with such filters, users can either switch off their connection or find alternatives, e.g. by using TLS to access some services over TCP port 443. The planned deployment of QUIC [[I-D.ietf-quic-transport](#)] [[I-D.ietf-quic-tls](#)] opens a new opportunity for such users. Since QUIC will be used to access web sites, it should be less affected by filters than VPN solutions such as IPSec or DTLS. Furthermore, the flexibility of QUIC makes it possible to easily extend the protocol to support VPN services.

This document shares some goals with the MASQUE framework [[I-D.schinazi-masque](#)]. The proposed QUIC tunnel protocol contributes to the effort of defining a signaling for conveying multiple proxied flows inside a QUIC connection. While this document specifies its own protocol, further work could adapt the mechanisms presented in this proposal to use HTTP/3.

On the other hand, today's mobile devices are often multihomed and many expect to be able to perform seamless handovers from one access network to another without breaking the established VPN sessions. In some situations it can also be beneficial to combine two or more access networks together to increase the available host bandwidth. A protocol such as Multipath TCP [[RFC6824](#)] supports those handovers and allows aggregating the bandwidth of different access links. It could be combined with single-path VPN protocols to support both seamless handovers and bandwidth aggregation above VPN tunnels. Unfortunately, Multipath TCP is not yet deployed on most Internet servers and thus few applications would benefit from such a use case.

In this document, we explore how QUIC could be used to enable multihomed mobile devices to communicate securely in untrusted networks. The QUIC protocol opens up a new way to find a clean solution to this problem. First, QUIC includes the same encryption and authentication techniques as deployed VPN protocols. Second, QUIC is intended to be widely used to support web-based services, making it unlikely to be filtered in many networks, in contrast with VPN protocols. Third,

the QUIC migration mechanism enables handovers between several network interfaces.

This document is organized as follows. [Section 3](#) describes our reference environment. Then, we propose two mode of operations, explained in [Section 4](#) and [Section 5](#), that use the recently proposed datagram extension ([I-D.pauly-quic-datagram](#)) for QUIC to transport plain IP packets over a QUIC connection. [Section 6](#) specifies how a connection is established in this document proposal. [Section 7](#) details the format of the messages introduced by this document.

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.

3. Reference environment

Our first scenario is a client that uses a QUIC tunnel to send all its packets to a concentrator. The concentrator decrypts the packets received over the QUIC connection and forwards them to their final destination. It also receives the packets destined to the client and tunnels them through the QUIC connection.

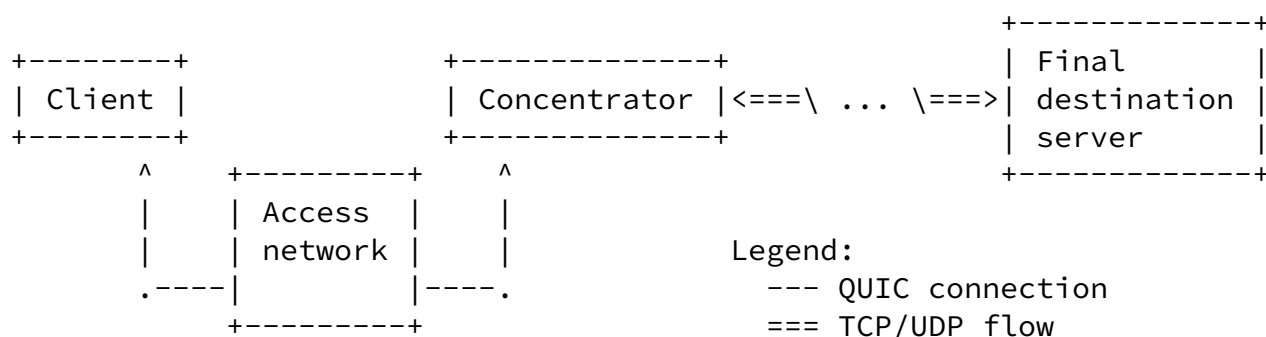


Figure 1: A client attached to a concentrator

However, there are several situations where the client is attached to

two or more access networks. This client can be multihomed, dual-stack, ... This is illustrated in Figure 2, in which a client-initiated flow is tunneled through the concentrator. We also discuss inbound connections in this document in [Section 6](#).

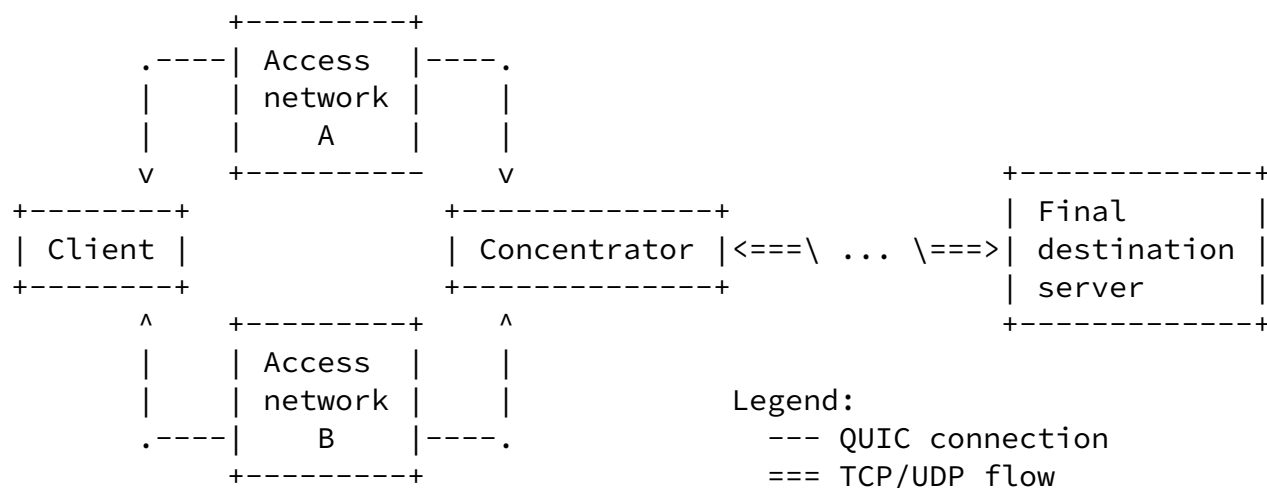


Figure 2: Example environment

Such a client would like to benefit from the different access networks available to reach the concentrator. These access networks can be used for load-sharing, failover or other purposes. One possibility to efficiently use these two access networks is to rely on the proposed Multipath extensions to QUIC [\[I-D.deconinck-quic-multipath\]](#). Another approach is to create one

QUIC connection using the single-path QUIC protocol [[I-D.ietf-quic-transport](#)] over each access network and glue these different sessions together on the concentrator. Given the migration capabilities of QUIC, this approach could support failover with a single active QUIC connection at a time.

In a nutshell, the solution proposed in this document works as follows. The client opens a QUIC connection to a concentrator. The concentrator authenticates the client through means that are outside the scope of this document such as client certificates, usernames/passwords, OAuth, ... If the authentication succeeds, the client can use the tunnel to exchange Ethernet frames and IP packets with the concentrator over the QUIC session. If the client uses IP, then the concentrator can allocate an IP address to the client at the end of the authentication phase. The client can then send packets via the concentrator by tunneling them through the concentrator. The concentrator captures the IP packets destined to the client and tunnels them over the QUIC connection. Our solution is intended to provide a similar service as the one provided by IPsec tunnels or DTLS.

[4.](#) The lightweight mode

Our first mode of operation is very simple. It leverages the recently proposed QUIC datagram extension [[I-D.pauly-quic-datagram](#)]. In a nutshell, to send a packet to a remote host, the client simply

encodes the entire packet inside a QUIC DATAGRAM frame sent over the QUIC connection established with the concentrator.

This QUIC DATAGRAM frame is then encrypted and authenticated in a QUIC packet. This transmission is subject to congestion control, but the frame that contains the packet is not retransmitted in case of losses as specified in [[I-D.pauly-quic-datagram](#)].

This mode adds a minimal byte overhead for packet encapsulation in QUIC. It does not define ways of indicating the protocol of the conveyed packets, which can be useful in deployments for which out-of-band signalling can be used.

[5.](#) The datagram mode

Our second mode of operation, called the datagram mode in this document, enables the client and the concentrator to exchange packets of several network protocols through the QUIC tunnel connection at the same time. It also leverages the QUIC datagram extension [[I-D.pauly-quic-datagram](#)].

This document specifies the following format for encoding packets in QUIC DATAGRAM frame. It allows encoding packets from several protocols by identifying the corresponding protocol of the packet in each QUIC DATAGRAM frame. Figure 3 describes this encoding.

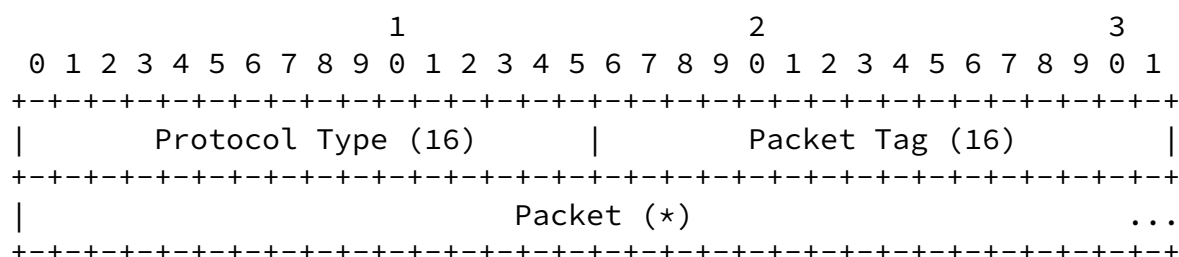


Figure 3: Encoding packets in QUIC DATAGRAM frame

This encoding defines three fields.

- o Protocol Type: The Protocol Type field contains the protocol type of the payload packet. The values for the different protocols are defined as "ETHER TYPES" in [[IANA-ETHER-TYPES](#)]. A QUIC tunnel that receives a Protocol Type representing an unsupported protocol MAY drop the associated Packet. QUIC tunnel endpoints willing to exchange Ethernet frames can use the value 0x6558 for [[Transparent-Ethernet-Bridging](#)].
- o Packet Tag: An opaque 16-bit value. The QUIC tunnel application is free to decide its semantic value. For instance, a QUIC tunnel

endpoint MAY encode the sending order of packets in the Packet Tag, e.g. as a timestamp or a sequence number, to allow reordering on the receiver.

- o Packet: The packet conveyed inside the QUIC tunnel connection.

```

,->+-----+
|  |      IP  |

```

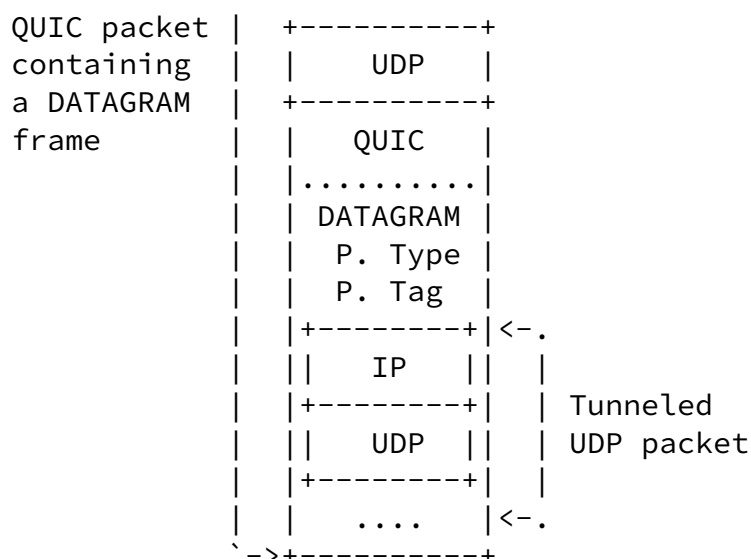


Figure 4: QUIC packet sent by the client when tunneling a UDP packet

Figure 4 illustrates how a UDP packet is tunneled using the datagram mode. The main advantage of the datagram mode is that it supports IP and any protocol above the network layer. Any IP packet can be transported using the datagram extension over a QUIC connection. However, this advantage comes with a large per-packet overhead since each packet contains both a network and a transport header. All these headers must be transmitted in addition with the IP/UDP/QUIC headers of the QUIC connection. For TCP connections for instance, the per-packet overhead can be large.

5.1. Joining a tunneling session

If the client is multihomed, it can use Multipath QUIC [[I-D.deconinck-quic-multipath](#)] to efficiently use its different access networks. This version of the document does not elaborate in details on this possibility. If the concentrator does not support Multipath QUIC, then the client creates several QUIC connections and joins them at the application layer. This works as illustrated in figure Figure 5. Each message is exchanged over a dedicated unidirectional QUIC stream. Their format is detailed in [Section 7](#). When the client opens the first QUIC connection with the concentrator, (1) it can request a QUIC tunnel session identifier.

(2) The concentrator replies with a variable-length opaque value that

identifies the QUIC tunneling session. When opening a QUIC connection over another access network, (3) the client can send this identifier to join the QUIC tunneling session. The concentrator matches the session identifier with the existing session with the client. It can then use both sessions to reach the client and received tunneled packets from the client.

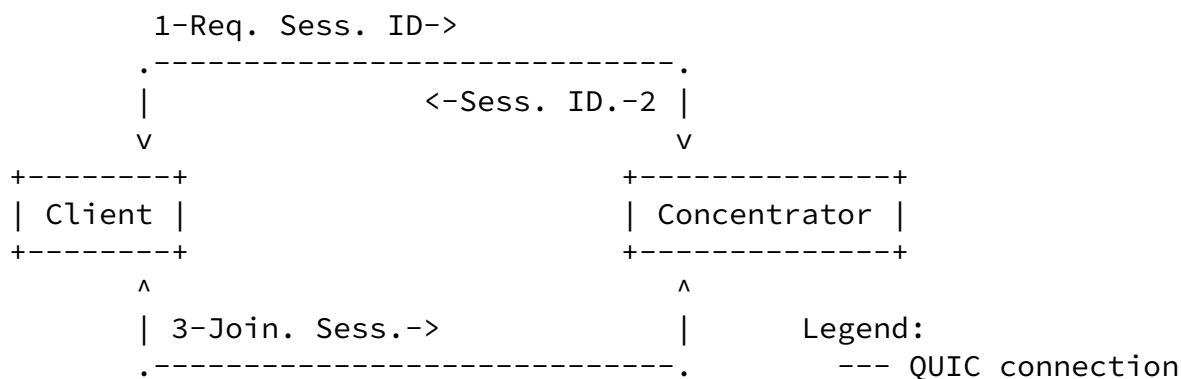


Figure 5: Creating sessions over different access networks

Joining a tunneling session allows grouping several QUIC connections to the concentrator. Each endpoint can then coordinate the use of the Packet Tag across the tunneling session as presented in [Section 5.1.1](#).

Both QUIC tunnel endpoints open their first unidirectional stream (i.e. stream 2 and 3), hereafter named the QUIC tunnel control stream, to exchange these messages. A QUIC tunnel endpoint **MUST NOT** close its unidirectional stream and **SHOULD** provide enough flow control credit to its peer.

The messages format used for this purpose are described in [Section 7](#). The client initiates the procedure and **MAY** either start a new session or join an existing session. This negotiation **MUST NOT** take place more than once per QUIC connection.

[5.1.1](#). Coordinate use of the Packet Tag

When using the datagram mode, each packet is associated with a 16-bit value called Packet Tag. This document leaves defining the meaning of this value to implementations. This section provides some examples on how it can be used to implement packet reordering across several QUIC tunnel connections grouped in a tunneling session.

A first simple example of use is to encode the timestamp at which the datagram was sent. Using a millisecond precision and encoding the 16

lower bits of the timestamp makes the value wrapping around in a bit more than 65 seconds.

Another example of use is to maintain a value counting the datagrams sent over all QUIC tunnel connections of the tunneling session. The 16-bit value allows distinguishing at most 32768 packets in flight.

The QUIC tunnel receiver can then distinguish, buffer and reorder packets based on this value. Mechanisms for managing the datagram buffer and negotiating the use of the Packet Tag are out of scope of this document.

6. Connection establishment

During connection establishment, the QUIC tunnel datagram mode (resp. lite mode) support is indicated by setting the ALPN token "qt" (resp. "qt-lite") in the TLS handshake. Draft-version implementations MAY specify a particular draft version by suffixing the token, e.g. "qt-00" (resp. "qt-lite-03") refers to the first (resp. fourth) version of this document.

After the QUIC connection is established, the client can start using the datagram or the stream mode. The client may use PCP [[RFC6887](#)] to request the concentrator to accept inbound connections on their behalf. After the negotiation of such port mappings, the concentrator can start sending packets containing inbound connections in QUIC DATAGRAM frame.

7. Messages format

In the following sections, we specify the format of each message introduced in this document. They are encoded as TLVs, i.e. (Type, Length, Value) tuples, as illustrated in Figure 6. All TLV fields are encoded in network-byte order.

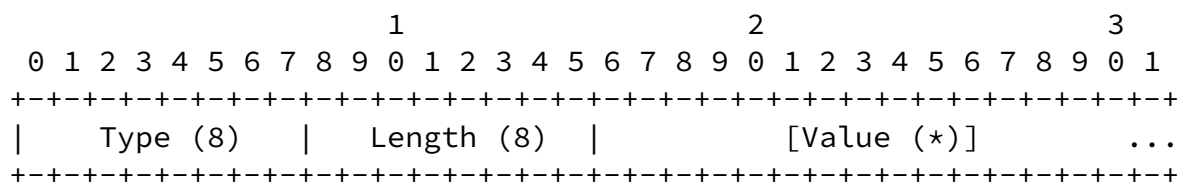


Figure 6: QUIC tunnel TLV Format

The Type field is encoded as a byte and identifies the type of the TLV. The Length field is encoded as a byte and indicate the length of the Value field. A value of zero indicates that no Value field is

present. The Value field is a type-specific value whose length is determined by the Length field.

7.1. QUIC tunnel control TLVs

This document specifies the following QUIC tunnel control TLVs:

Type	Size	Sender	Mode	Name
0x00	4 bytes	Client	all	Access Report TLV
0x01	2 bytes	Client	datagram	New Session TLV
0x02	Variable	Concentrator	datagram	Session ID TLV
0x03	Variable	Client	datagram	Join Session TLV

Figure 7: QUIC tunnel control TLVs

The Access Report TLV is sent by the client to periodically report on access networks availability. Each Access Report TLV MUST be sent on a separate unidirectional stream. When the datagram mode is in use, this separate stream MUST be other than the QUIC tunnel control stream.

The New Session TLV is used by the client to initiate a new tunneling session. The Session ID TLV is used by the concentrator to communicate to the client the Session ID identifying this tunneling session. The Join Session TLV is used to join a given tunneling session, identified by a Session ID. All QUIC these tunnel control TLVs MUST NOT be sent on other streams than the QUIC tunnel control streams.

7.1.1. Access Report TLV

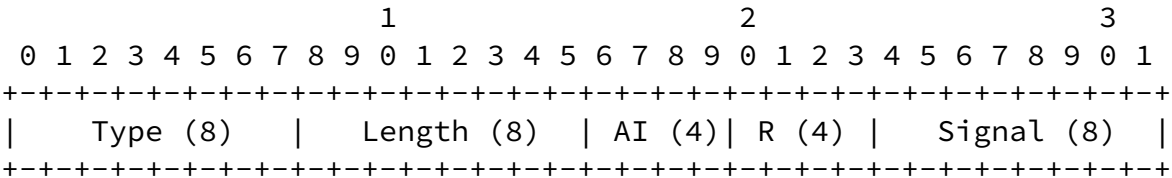


Figure 8: Access Report TLV

The Access Report TLV contains the following:

- o AI (Access ID) - a four-bit-long field that identifies the access network, e.g., 3GPP (Radio Access Technologies specified by 3GPP) or Non-3GPP (accesses that are not specified by 3GPP) [[TS23501](#)]. The value is one of those listed below (all other values are invalid and the TLV that contains it MUST be discarded):

+-----+-----+	
Access ID	Description
+-----+-----+	
1	3GPP Network
2	Non-3GPP Network
+-----+-----+	

- o R (Reserved) - a four-bit-long field that MUST be zeroed on transmission and ignored on receipt.
- o Signal - a one-octet-long field that identifies the report signal, e.g., available or unavailable. The value is supplied to the QUIC tunnel through some mechanism that is outside the scope of this document. The value is one of those listed in [Section 9.4](#).

The client that includes the Access Report TLV sets the value of the Access ID field according to the type of access network it reports on. Also, the client sets the value of the Signal field to reflect the operational state of the access network. The mechanism to determine the state of the access network is outside the scope of this specification.

The client MUST be able to cancel the sending of an Access Report TLV that is pending delivery, i.e. by resetting its corresponding unidirectional stream. This can be used when the information contained in the TLV is no longer relevant, e.g. the access network availability has changed. The time of canceling is based on local policies and network environment.

Reporting the unavailability an access network to the concentrator can serve as an advisory signal to preventively stop sending packets over this network while maintaining the QUIC tunnel connection. Upon

reporting of the availability of this network, the concentrator can quickly resume sending packets over this network.

[7.1.2.](#) New Session TLV

The New Session TLV does not contain a value. It initiates a new tunneling session at the concentrator. The concentrator **MUST** send a Session ID TLV in response, with the Session ID corresponding to the tunneling session created. After sending a New Session TLV, the client **MUST** close the QUIC tunnel control stream.

The concentrator **MUST NOT** send New Session TLVs.

[7.1.3.](#) Session ID TLV

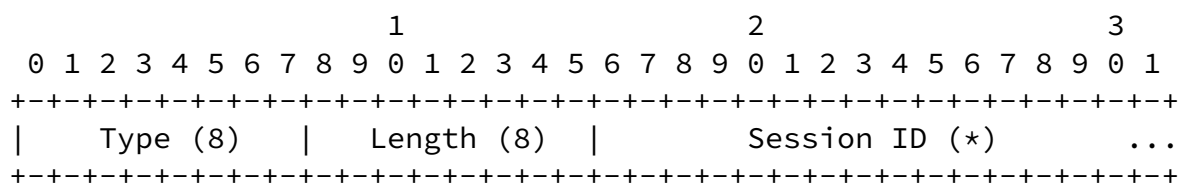
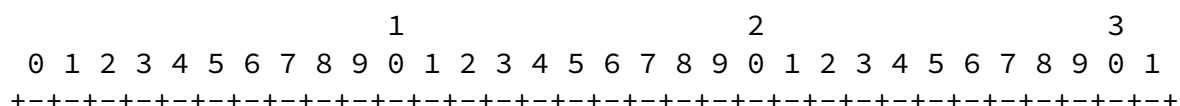


Figure 9: Session ID TLV

The Session ID TLV contains an opaque value that identifies the current tunneling session. It can be used by the client in subsequent QUIC connections to join them to this tunneling session. The concentrator **MUST** send a Session ID TLV in response of a New Session TLV, with the Session ID corresponding to the tunneling session created.

The client **MUST NOT** send a Session ID TLV. The concentrator **MUST** close the QUIC tunnel control stream after sending a Session ID TLV.

[7.1.4.](#) Join Session TLV



Type (8)	Length (8)	Session ID (*)	...
+-+-+-----	+-+-+-----	+-+-+-----	+-+-+-----

Figure 10: Join Session TLV

The Join Session TLV contains an opaque value that identifies a tunneling session to join. The client can send a Join Session TLV to join the QUIC connection to a particular tunneling session. The tunneling session is identified by the Session ID. After sending a Join Session TLV, the client MUST close the QUIC tunnel control stream.

The concentrator MUST NOT send Join Session TLVs. After receiving a Join Session TLV, the concentrator MUST use the Session ID to join this QUIC connection to the tunneling session. Joining the tunneling session implies merging the state of this QUIC tunnel connection to the session. A successful joining of connection is indicated by the closure of the QUIC tunnel control stream of the concentrator.

In cases of failure when joining a tunneling session, the concentrator MUST send a RESET_STREAM with an application error code discerning the cause of the failure. The possible codes are listed below:

- o UNKNOWN_ERROR (0x0): An unknown error occurred when joining the tunneling session. QUIC tunnel endpoints SHOULD use more specific error codes when applicable.
- o UNKNOWN_SESSION_ID (0x1): The Session ID used in the Join Session TLV is not a valid ID. It was not issued in a Session ID TLV or refers to an expired tunneling session.
- o CONFLICTING_STATE (0x2): The current state of the QUIC tunnel connection could not be merged with the tunneling session.

8. Security Considerations

8.1. Privacy

The Concentrator has access to all the packets it processes. It MUST be protected as a core IP router, e.g. as specified in [\[RFC1812\]](#).

[8.2.](#) Ingress Filtering

Ingress filtering policies MUST be enforced at the network boundaries, i.e. as specified in [[RFC2827](#)].

[9.](#) IANA Considerations

[9.1.](#) Registration of QUIC tunnel Identification String

This document creates two new registrations for the identification of the QUIC tunnel protocol in the "Application Layer Protocol Negotiation (ALPN) Protocol IDs" registry established in [[RFC7301](#)].

The "qt" string identifies the QUIC tunnel protocol datagram mode.

Protocol: QUIC Tunnel

Identification Sequence: 0x71 0x74 ("qt")

Specification: This document

The "qt-lite" string identifies the QUIC tunnel protocol lightweight mode.

Protocol: QUIC Tunnel lightweight mode

Identification Sequence: 0x71 0x74 0x55 0x6c 0x69 0x74 0x65 ("qt-lite")

Specification: This document

[9.2.](#) QUIC tunnel control TLVs

IANA is requested to create a new "QUIC tunnel control Parameters" registry.

The following subsections detail new registries within "QUIC tunnel control Parameters" registry.

[9.2.1.](#) QUIC tunnel control TLVs Types

IANA is request to create the "QUIC tunnel control TLVs Types" sub-

registry. New values are assigned via IETF Review ([Section 4.8 of \[RFC8126\]](#)).

The initial values to be assigned at the creation of the registry are as follows:

Code	Name	Reference
0	Access Report TLV	[This-Doc]
1	New Session TLV	[This-Doc]
2	Session ID TLV	[This-Doc]
3	Join Session TLV	[This-Doc]

[9.3.](#) QUIC tunnel control Error Codes

This document establishes a registry for QUIC tunnel control stream error codes. The "QUIC tunnel control Error Code" registry manages a 62-bit space. New values are assigned via IETF Review ([Section 4.8 of \[RFC8126\]](#)).

The initial values to be assigned at the creation of the registry are as follows:

Code	Name	Reference
0	UNKNOWN_ERROR	[This-Doc]
1	UNKNOWN_SESSION_ID	[This-Doc]
2	CONFLICTING_STATE	[This-Doc]

[9.4.](#) QUIC tunnel Access Report Signal Codes

This document establishes a registry for QUIC tunnel Access Report Signal codes. The "QUIC tunnel Access Report Signal Code" registry

manages a 62-bit space. New values are assigned via IETF Review ([Section 4.8 of \[RFC8126\]](#)).

The initial values to be assigned at the creation of the registry are as follows:

Code	Name	Reference
1	Access Available	[This-Doc]
2	Access Unavailable	[This-Doc]

[10.](#) References

[10.1.](#) Normative References

- [RFC1701] Hanks, S., Li, T., Farinacci, D., and P. Traina, "Generic Routing Encapsulation (GRE)", [RFC 1701](#), DOI 10.17487/RFC1701, October 1994, <<https://www.rfc-editor.org/info/rfc1701>>.
- [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>>.
- [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>>.
- [TS23501] 3GPP (3rd Generation Partnership Project), "Technical Specification Group Services and System Aspects; System Architecture for the 5G System; Stage 2 (Release 16)", 3GPP TS23501, 2019.

[10.2.](#) Informative References

- [I-D.deconinck-quic-multipath] Coninck, Q. and O. Bonaventure, "Multipath Extensions for QUIC (MP-QUIC)", [draft-deconinck-quic-multipath-04](#) (work in progress), March 2020.

[I-D.ietf-quic-tls]

Thomson, M. and S. Turner, "Using TLS to Secure QUIC", [draft-ietf-quic-tls-29](#) (work in progress), June 2020.

[I-D.ietf-quic-transport]

Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", [draft-ietf-quic-transport-29](#) (work in progress), June 2020.

[I-D.pauly-quic-datagram]

Pauly, T., Kinnear, E., and D. Schinazi, "An Unreliable Datagram Extension to QUIC", [draft-pauly-quic-datagram-05](#) (work in progress), November 2019.

[I-D.schinazi-masque]

Schinazi, D., "The MASQUE Protocol", [draft-schinazi-masque-02](#) (work in progress), January 2020.

[IANA-ETHER-TYPES]

"IANA ETHER TYPES", <https://www.iana.org/assignments/ieee-802-numbers/ieee-802-numbers.txt> , n.d..

[RFC1812] Baker, F., Ed., "Requirements for IP Version 4 Routers", [RFC 1812](#), DOI 10.17487/RFC1812, June 1995, <<https://www.rfc-editor.org/info/rfc1812>>.

[RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", [BCP 38](#), [RFC 2827](#), DOI 10.17487/RFC2827, May 2000, <<https://www.rfc-editor.org/info/rfc2827>>.

[RFC6824] Ford, A., Raiciu, C., Handley, M., and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses", [RFC 6824](#), DOI 10.17487/RFC6824, January 2013, <<https://www.rfc-editor.org/info/rfc6824>>.

[RFC6887] Wing, D., Ed., Cheshire, S., Boucadair, M., Penno, R., and P. Selkirk, "Port Control Protocol (PCP)", [RFC 6887](#), DOI 10.17487/RFC6887, April 2013, <<https://www.rfc-editor.org/info/rfc6887>>.

[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>>.

Internet-Draft

QUIC Tunnel

August 2020

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", [BCP 26](#), [RFC 8126](#), DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.

[Transparent-Ethernet-Bridging]
Hanks, S., Li, T., Farinacci, D., and P. Traina, "Generic Routing Encapsulation (GRE)", [RFC 1701](#), DOI 10.17487/RFC1701, October 1994, <<https://www.rfc-editor.org/info/rfc1701>>.

[Appendix A](#). Change Log

[A.1](#). Since [draft-piroux-quic-tunnel-02](#)

- o Add the lightweight mode

[A.2](#). Since [draft-piroux-quic-tunnel-01](#)

- o Add the Access Report TLV for reporting access networks availability
- o Add a section with examples of use of the Packet Tag

[A.3](#). Since [draft-piroux-quic-tunnel-00](#)

- o Separate the document in two and put the stream mode in another document
- o Remove TCP Extended TLV
- o Add a mechanism for joining QUIC connections in a QUIC tunneling session
- o Add a format for encoding any network-layer protocol packets and Ethernet frames in QUIC DATAGRAM frames

Acknowledgments

Thanks to Quentin De Coninck and Francois Michel for their comments

and the proofreading of the first version of this document. Thanks to Gregory Vander Schueren for his comments on the first version of this document.

Piroux, et al.

Expires February 13, 2021

[Page 17]

Internet-Draft

QUIC Tunnel

August 2020

Authors' Addresses

Maxime Piroux
UCLouvain

Email: maxime.piroux@uclouvain.be

Olivier Bonaventure
UCLouvain

Email: olivier.bonaventure@uclouvain.be

Adi Masputra
Apple Inc.

Email: adi@apple.com

