Workgroup: Network Working Group
Internet-Draft: draft-schinazi-masque-02
Published: 8 January 2020
Intended Status: Experimental
Expires: 11 July 2020
Authors: D. Schinazi
Google LLC

The MASQUE Protocol

Abstract

This document describes MASQUE (Multiplexed Application Substrate over QUIC Encryption). MASQUE is a framework that allows concurrently running multiple networking applications inside an HTTP/3 connection. For example, MASQUE can allow a QUIC client to negotiate proxying capability with an HTTP/3 server, and subsequently make use of this functionality while concurrently processing HTTP/3 requests and responses.

This document is a straw-man proposal. It does not contain enough details to implement the protocol, and is currently intended to spark discussions on the approach it is taking. Discussion of this work is encouraged to happen on the MASQUE IETF mailing list masque@ietf.org or on the GitHub repository which contains the draft: https://github.com/DavidSchinazi/masque-drafts.

Table of Contents

1. Introduction

This document describes MASQUE (Multiplexed Application Substrate over QUIC Encryption). MASQUE is a framework that allows concurrently running multiple networking applications inside an

HTTP/3 connection (see [HTTP3]). For example, MASQUE can allow a QUIC client to negotiate proxying capability with an HTTP/3 server, and subsequently make use of this functionality while concurrently processing HTTP/3 requests and responses.

MASQUE Negotiation is performed using HTTP mechanisms, but MASQUE applications can subsequently leverage QUIC [QUIC] features without using HTTP.

This document is a straw-man proposal. It does not contain enough details to implement the protocol, and is currently intended to spark discussions on the approach it is taking. Discussion of this work is encouraged to happen on the MASQUE IETF mailing list masque@ietf.org or on the GitHub repository which contains the draft: https://github.com/DavidSchinazi/masque-drafts.

1.1. 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.

2. MASQUE Negotiation

In order to negotiate the use of the MASQUE protocol, the client starts by sending a MASQUE request in the HTTP data of an HTTP POST request to "/.well-known/masque/initial". The client can use this to request specific MASQUE applications and advertise support for MASQUE extensions. The MASQUE server indicates support for MASQUE by sending an HTTP status code 200 response, and can use the data to inform the client of which MASQUE applications are now in use, and various configuration parameters.

Both the MASQUE negotiation initial request and its response carry a list of type-length-value fields. The type field is a number corresponding to a MASQUE application, and is encoded as a QUIC variable-length integer. The length field represents the length in bytes of the value field, encoded as a QUIC variable-length integer. The contents of the value field or defined by its corresponding MASQUE application. When parsing, endpoints MUST ignore unknown MASQUE applications.

3. MASQUE Applications

As soon as the server has accepted the client's MASQUE initial request, it can advertise support for MASQUE Applications, which will be multiplexed over this HTTP/3 connection.

3.1. HTTP Proxy

The client can make proxied HTTP requests through the server to other servers. In practice this will mean using the CONNECT method to establish a stream over which to run TLS to a different remote destination. The proxy applies back-pressure to streams in both directions.

3.2. DNS over HTTPS

The client can send DNS queries using DNS over HTTPS [DOH] to the MASQUE server.

3.3. QUIC Proxying

By leveraging QUIC client connection IDs, a MASQUE server can act as a QUIC proxy while only using one UDP port. The server informs the client of a scheme for client connection IDs (for example, random of a minimum length or vended by the MASQUE server) and then the server can forward those packets to further web servers.

This mechanism can elide the connection IDs on the link between the client and MASQUE server by negotiating a mapping between DATAGRAM_IDs and the tuple (client connection ID, server connection ID, server IP address, server port).

Compared to UDP proxying, this mode has the advantage of only requiring one UDP port to be open on the MASQUE server, and can lower the overhead on the link between client and MASQUE server by compressing connection IDs.

3.4. UDP Proxying

In order to support WebRTC or QUIC to further servers, clients need a way to relay UDP onwards to a remote server. In practice for most widely deployed protocols other than DNS, this involves many datagrams over the same ports. Therefore this mechanism implements that efficiently: clients can use the MASQUE protocol stream to request an UDP association to an IP address and UDP port pair. In QUIC, the server would reply with a DATAGRAM_ID that the client can then use to have UDP datagrams sent to this remote server. Datagrams are then simply transferred between the DATAGRAMs with this ID and the outer server. There will also be a message on the MASQUE protocol stream to request shutdown of a UDP association to save resources when it is no longer needed.

3.5. IP Proxying

For the rare cases where the previous mechanisms are not sufficient, proxying can be performed at the IP layer. This would use a

different DATAGRAM_ID and IP datagrams would be encoded inside it without framing.

3.6. Service Registration

MASQUE can be used to make a home server accessible on the wide area. The home server authenticates to the MASQUE server and registers a domain name it wishes to serve. The MASQUE server can then forward any traffic it receives for that domain name (by inspecting the TLS Server Name Indication (SNI) extension) to the home server. This received traffic is not authenticated and it allows non-modified clients to communicate with the home server without knowing it is not colocated with the MASQUE server.

To help obfuscate the home server, deployments can use Encrypted Server Name Indication [ESNI]. That will require the MASQUE server sending the cleartext SNI to the home server.

4. Security Considerations

Here be dragons. TODO: slay the dragons.

5. IANA Considerations

This document will request IANA to register the "/.well-known/masque/" URI (expert review) https://www.iana.org/assignments/well-known-uris/well-known-uris.xhtml.

This document will request IANA to create a new MASQUE Applications registry which governs a 62-bit space of MASQUE application types.

6. References

6.1. Normative References

[HTTP3] Bishop, M., "Hypertext Transfer Protocol Version 3 (HTTP/3)", Work in Progress, Internet-Draft, draft-ietf-quic-http-24, 4 November 2019, <http://www.ietf.org/internet-drafts/draft-ietf-quic-http-24.txt>.

[QUIC] Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", Work in Progress, Internet-Draft, draft-ietf-quic-transport-24, 3 November 2019, <http://www.ietf.org/internet-drafts/draft-ietf-quic-transport-24.txt>.

[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>.

[DOH] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <https://www.rfc-editor.org/info/rfc8484>.

6.2. Informative References

[ESNI] Rescorla, E., Oku, K., Sullivan, N., and C. Wood, "Encrypted Server Name Indication for TLS 1.3", Work in Progress, Internet-Draft, draft-ietf-tls-esni-05, 4 November 2019, <http://www.ietf.org/internet-drafts/draft-ietf-tls-esni-05.txt>.

Acknowledgments

This proposal was inspired directly or indirectly by prior work from many people. The author would like to thank Nick Harper, Christian Huitema, Marcus Ihlar, Eric Kinnear, Mirja Kuehlewind, Brendan Moran, Lucas Pardue, Tommy Pauly, Zaheduzzaman Sarker, Ben Schwartz, and Christopher A. Wood for their input.

Author's Address

David Schinazi
Google LLC
1600 Amphitheatre Parkway
Mountain View, California 94043,
United States of America

Email: dschinazi.ietf@gmail.com