MASQUE CONNECT-UDP Bind

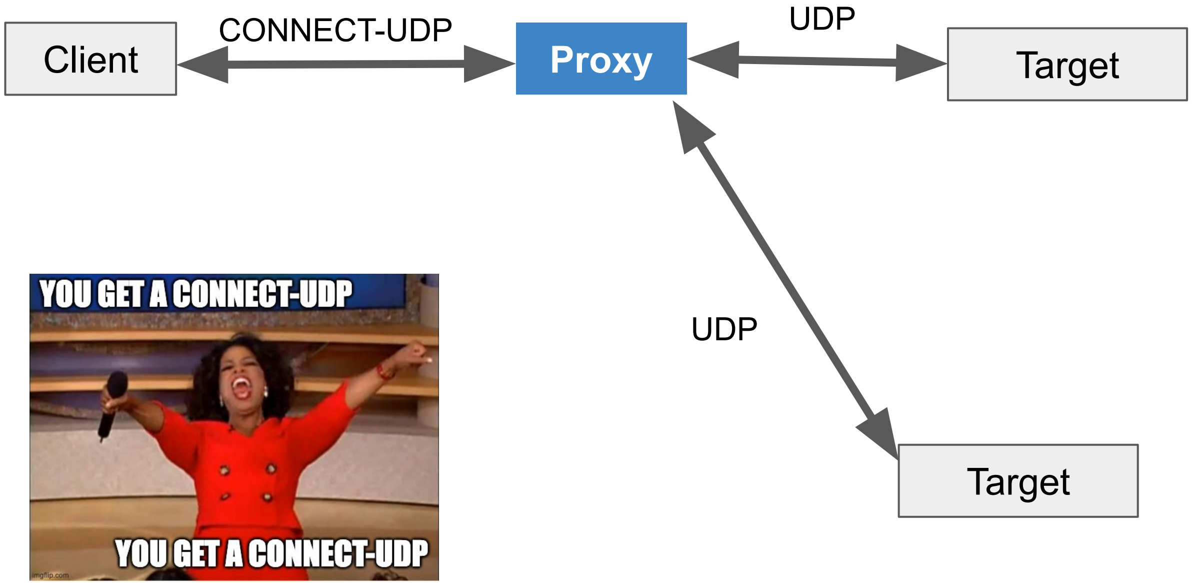
draft-ietf-masque-connect-udp-listen

IETF 119 – Brisbane– 2024-03-19

David Schinazi – dschinazi.ietf@gmail.com
Abhi Singh - abhisinghietf@gmail.com

CONNECT-UDP - with Binding support

Infinite 5-tuples using just one CONNECT-UDP connection.



How does it work?

```
HEADERS
  :method = CONNECT
  :protocol = connect-udp
  :scheme = https
  :path = /masque/udp/*/*/
  :authority = proxy.org
  capsule-protocol = ?1
  connect-udp-listen = 42
```

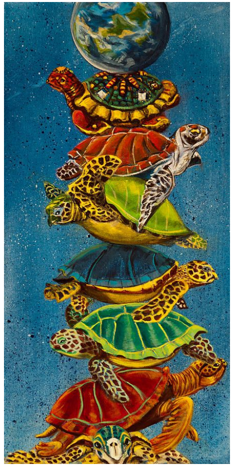
```
DATAGRAM QUIC Frame {
  Type (i) = 0x30..0x31,
  [Length (i)],
  Quarter Stream ID (i),
  Context ID (i) = 42,
  IP Version (8),
  IP Address (32..128),
  UDP Port (16),
  UDP Payload (..),
}
```

QUIC

HTTP/3

CONNECT-UDP

CONNECT-UDP-Listen



Context ID registered by header – payload then contains IP & port

More about the IP fields

```
IP Version (8),
IP Address (32..128),
UDP Port (16),
```

These Fields reflect:

client -> proxy

Target IP/Port PER PAYLOAD

proxy -> client

Source IP/Port PER PAYLOAD

Shall we validate source packets?

Changes since last time (Closed issues)

1) Rename CONNECT-UDP Listener to CONNECT-UDP Bind. (As agreed in the mailing list and 118 WG meeting)

https://github.com/ietf-wg-masque/draft-ietf-masque-connect-udp-listen/issues/1

2) Register "connect-udp-bind" as "Connect-UDP-Bind" instead (H1 style field name compliant to https://www.iana.org/assignments/http-fields/http-fields.xhtml)

https://github.com/ietf-wg-masque/draft-ietf-masque-connect-udp-listen/issues/20

Pending PRs

1) Add IP and port allocated to the client in the response header of the CONNECT UDP listen request

   https://github.com/ietf-wg-masque/draft-ietf-masque-connect-udp-listen/pull/18

2) Compress away IP and Port using Context IDs

   https://github.com/ietf-wg-masque/draft-ietf-masque-connect-udp-listen/pull/19

Add IP and Port to response header

1) Add IP(upto one v6 and one v4) and port allocated to the client in the response header of the CONNECT UDP listen request

   The following response header is added by the proxy when it accepts the CONNECT UDP Bind Request:

<-------- HEADERS

:status = 200

capsule-protocol = ?1

proxy-public-address = 192.0.2.45:54321, [2001:db8::1234]:54321

Compress away IP and Port using Context IDs

Achieves 2 goals at once

1) COMPRESSION: Save bytes from sending IP:Port per datagram using context IDs
2) IP RESTRICTION: Remove context IDs to reject traffic from a given target or all unregistered targets. Create them to poke holes in the “firewall”

New Capsule types

1) The COMPRESSION_ASSIGN request capsule:
    - Request the use of Context ID to compress a single target
    - Specify target IP version(4/6), IP address and port in assign Capsule
    - Can be done by either client or server with even and odd Context IDs respectively

Uncompressed COMPRESSION_ASSIGN request

- Set IP Version = 0 to define an uncompressed context ID
- When established, all targets not registered to other context IDs are matched to this Context ID i.e. accept and forward ALL traffic
- Only the client can allocate this kind of Context ID
- Datagrams with this context ID will retain the target IP and Port information per datagram

```
Capsule {
        Type COMPRESSION_ASSIGN,
        Length (i),
        Target Information,
}

Target Information {
        Context ID (i),
        IP Version (8),
        IP Address (32..128),
        UDP Port (16),
}
```

To accept a COMPRESSION_ASSIGN, echo back the COMPRESSION_ASSIGN to the sender

Now Datagrams can be sent on this context ID.
- If it is a compressed context, client and server omit IP-Port information and identify it using the Context ID
- If uncompressed, the client and server must specify target IP Port information per datagram

- Context IDs can be preemptively used, neither party absolutely needs to wait for confirmation but datagrams may get lost

Reject a context request with COMPRESSION_CLOSE

- Reject a COMPRESSION_ASSIGN request with this capsule as a response

- The client or proxy may reject an assign request if for example, they do not want to commit more memory towards the CONNECT-UDP Bind connection.

```
Capsule {
  Type COMPRESSION_CLOSE,
  Length (i),
  Context ID (i),
}
```

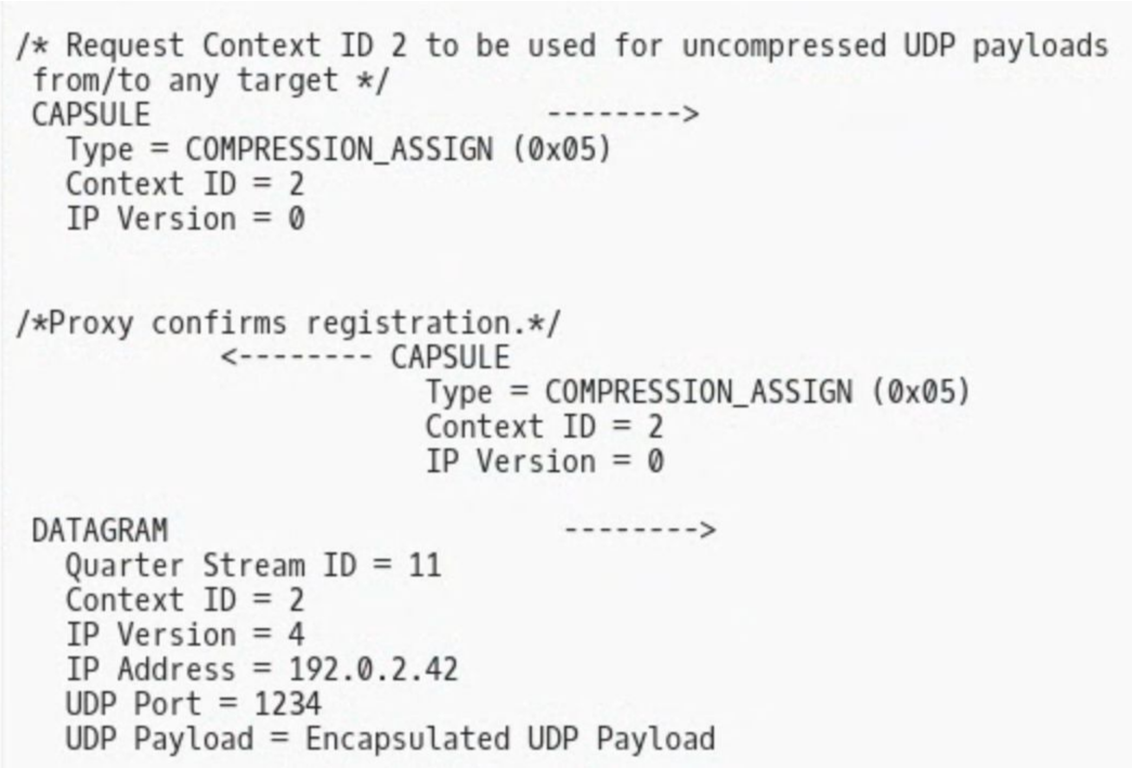
COMPRESSION_CLOSE for IP restriction

- The client or proxy can send a COMPRESSION_CLOSE to unregister an already established Context ID.

  Benefits:
- The proxy may use it to free up memory, if the client hasn’t communicated with the given Context ID mapped target in a long time.

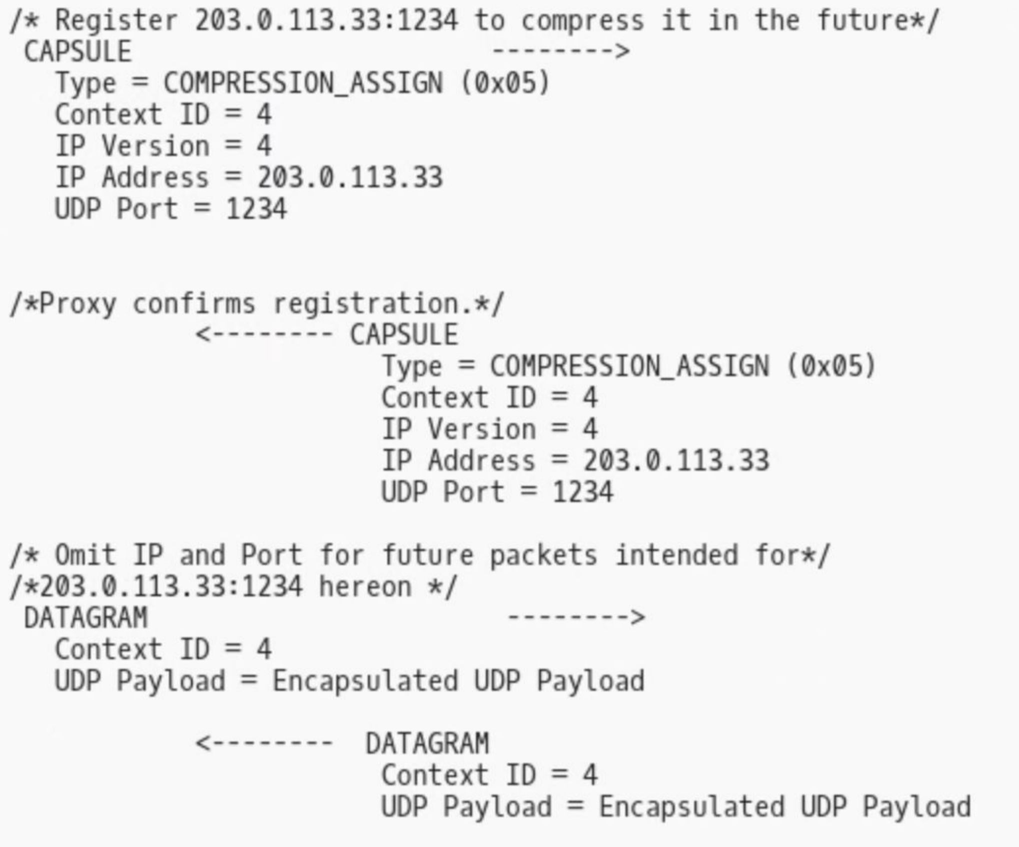
- IP Restriction: The client may use it to delete the Context ID used for uncompressed packets, establishing a firewall that allows only Context ID registered targets through.

- Uncompressed Context IDs have target Information per datagram. Example:



```
/* Request Context ID 2 to be used for uncompressed UDP payloads
 from/to any target */
 CAPSULE                       -------->
   Type = COMPRESSION_ASSIGN (0x05)
   Context ID = 2
   IP Version = 0


/*Proxy confirms registration.*/
            <-------- CAPSULE
                        Type = COMPRESSION_ASSIGN (0x05)
                        Context ID = 2
                        IP Version = 0

 DATAGRAM                       -------->
   Quarter Stream ID = 11
   Context ID = 2
   IP Version = 4
   IP Address = 192.0.2.42
   UDP Port = 1234
   UDP Payload = Encapsulated UDP Payload
```

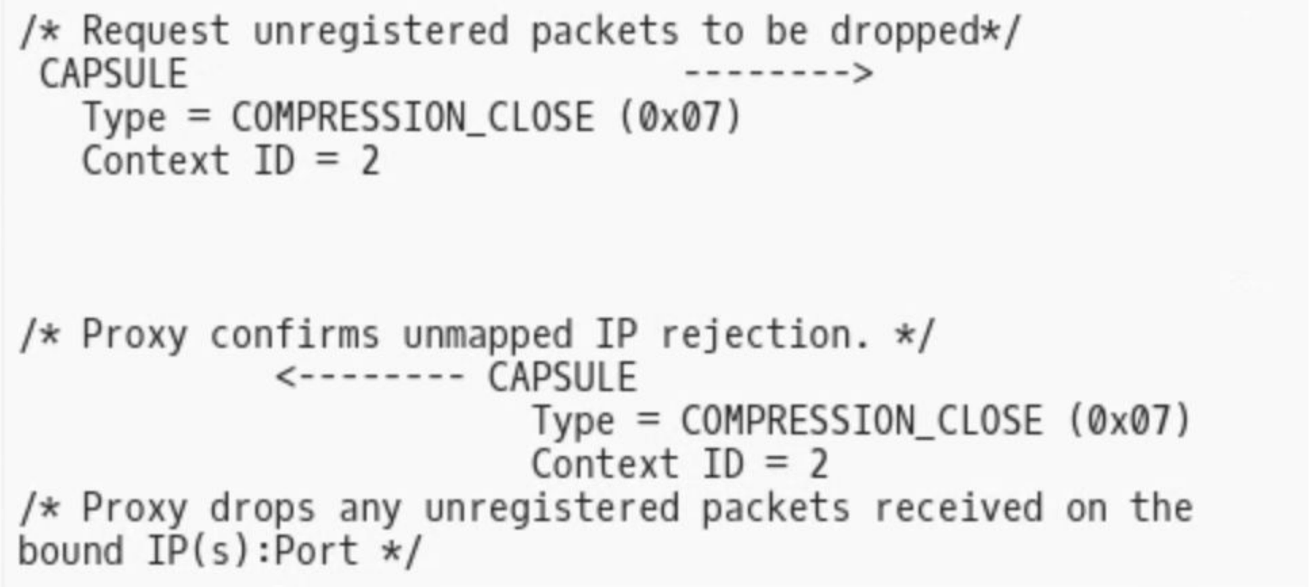
- Compressed Contexts omit this information.



```
/* Register 203.0.113.33:1234 to compress it in the future*/
 CAPSULE                       -------->
   Type = COMPRESSION_ASSIGN (0x05)
   Context ID = 4
   IP Version = 4
   IP Address = 203.0.113.33
   UDP Port = 1234


/*Proxy confirms registration.*/
            <-------- CAPSULE
                        Type = COMPRESSION_ASSIGN (0x05)
                        Context ID = 4
                        IP Version = 4
                        IP Address = 203.0.113.33
                        UDP Port = 1234

/* Omit IP and Port for future packets intended for*/
/*203.0.113.33:1234 hereon */
 DATAGRAM                       -------->
   Context ID = 4
   UDP Payload = Encapsulated UDP Payload

            <--------  DATAGRAM
                        Context ID = 4
                        UDP Payload = Encapsulated UDP Payload
```

A Context ID can be removed at any point.

Here we delete the context ID where uncompressed packets were being sent/received



```
/* Request unregistered packets to be dropped*/
 CAPSULE                       -------->
   Type = COMPRESSION_CLOSE (0x07)
   Context ID = 2



/* Proxy confirms unmapped IP rejection. */
            <-------- CAPSULE
                        Type = COMPRESSION_CLOSE (0x07)
                        Context ID = 2
/* Proxy drops any unregistered packets received on the
bound IP(s):Port */
```

MASQUE CONNECT-UDP Bind

draft-ietf-masque-connect-udp-listen

IETF 119 – Brisbane – 2024-03-19

David Schinazi – dschinazi.ietf@gmail.com
Abhi Singh - abhisinghietf@gmail.com