MASQUE CONNECT-UDP Listener

draft-schinazi-connect-udp-listen

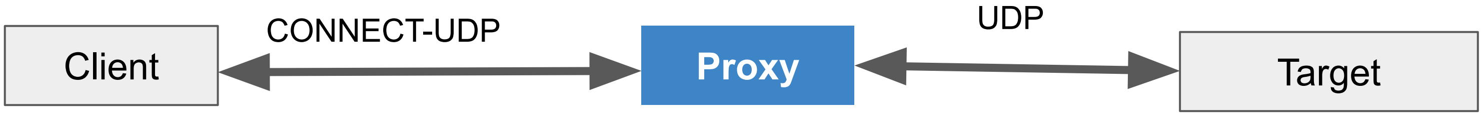
IETF 117 – San Francisco – 2022-07-24

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

CONNECT-UDP as it stands..

Exclusively allows using a single 5-tuple

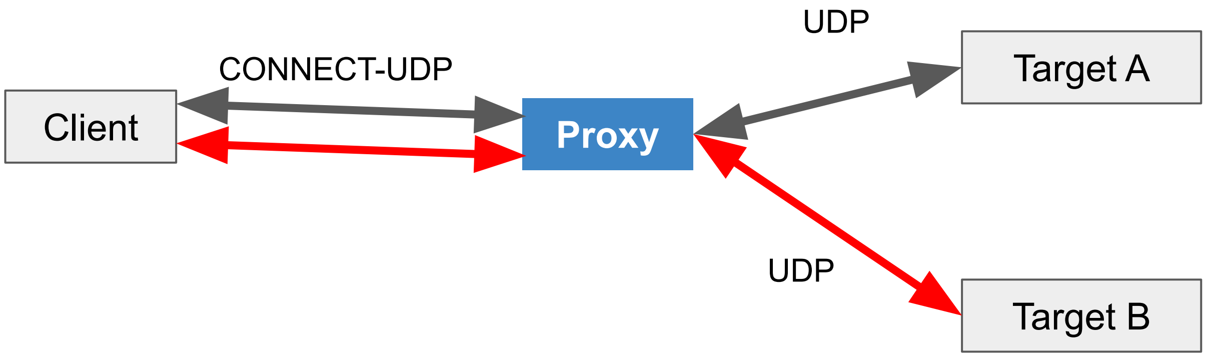
(Connected sockets only)



```
HEADERS
   :method = CONNECT
   :protocol = connect-udp
   :scheme = https
   :path = /.well-known/masque/udp/192.0.2.6/443/
   :authority = example.org
   capsule-protocol = ?1
```

Limitations

- Similarity to address-and-port dependent mapping on NATs

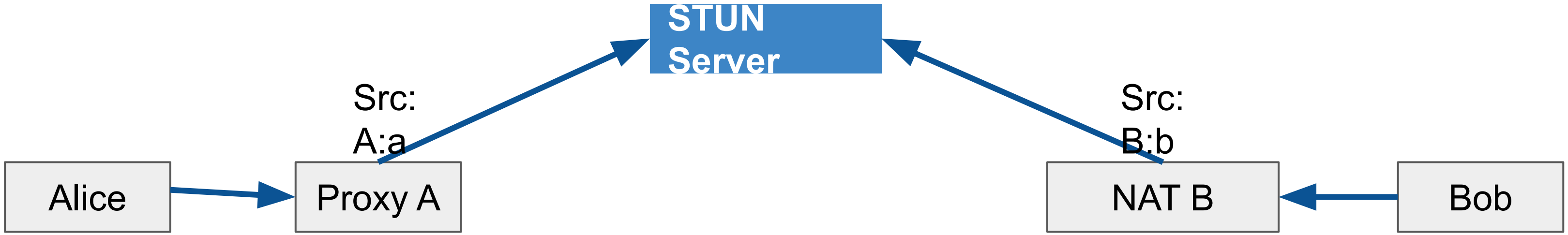


Need one connection to the proxy per target

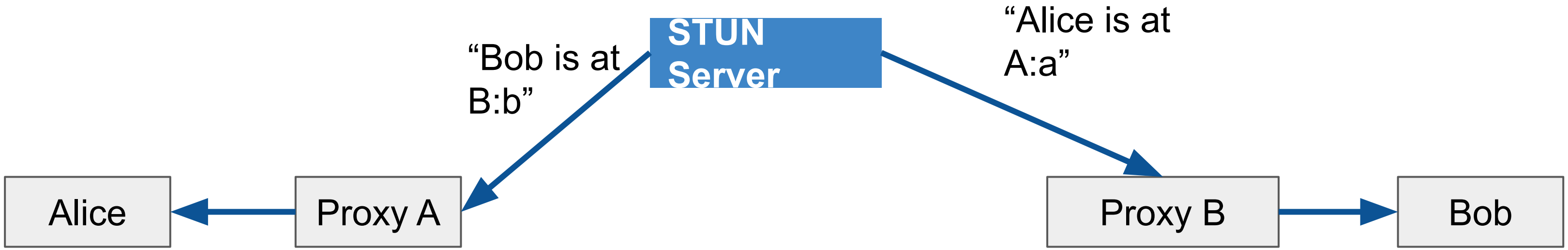
Target A and B will see the Client as different ports and even different IPs (due to HTTP semantics)

The case of two clients

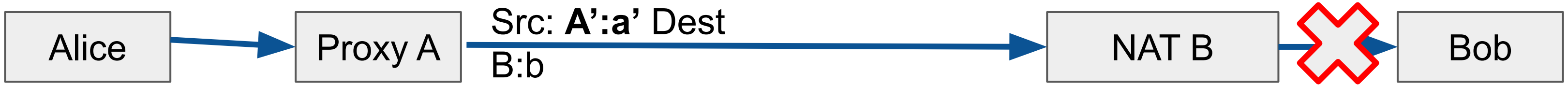
Bob and Alice want to establish peer to peer connections, Bob is behind a address-port dependent NAT and Alice is behind a CONNECT-UDP proxy



They receive each other’s address information

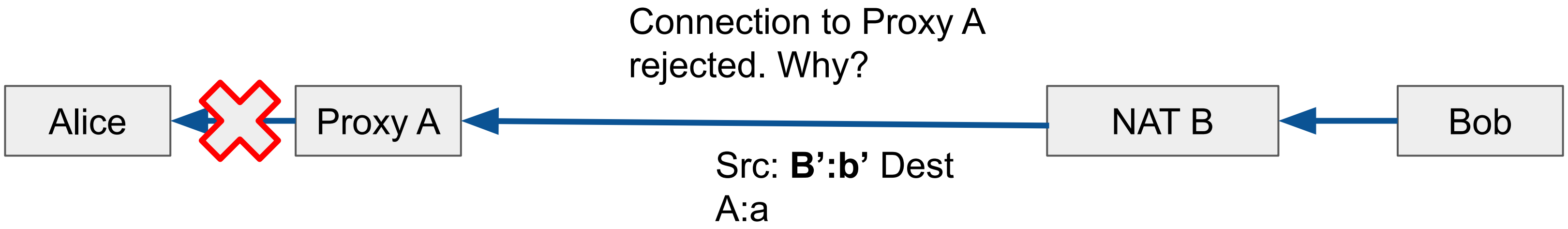


Now Alice uses the information it learned to try to connect to Bob



Connection to NAT B rejected,
but creates a “hole” for Source B:b packets
Notice that the source is **A’:a’** not, A:a

Now Bob tries to connect to Alice.



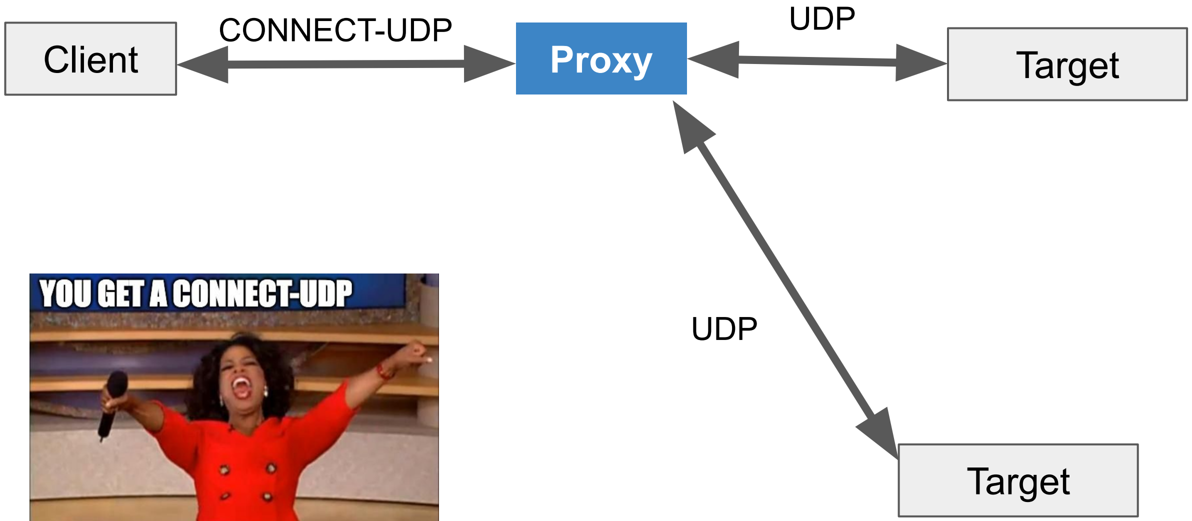
Proxy A has permission for B:b to pass through but the source from which it receives this packet is **B’:b’**.
The information they learned about each other’s addresses is essentially useless and a P2P connection is impossible.

Lessons from the world of STUN, TURN and NATs

- As long as, at least one of the two clients isn’t behind an address dependent NAT, P2P communication can be established.
- If both are behind address dependent NATs, a relay must be used.
- In our case, both our client and target can each be behind the CONNECT-UDP proxies, (or one behind the proxy and the other behind a address dependent NAT), hence WebRTC applications would be forced to run a relay. Seems like a waste when we already have a proxy (or two!) running.

CONNECT-UDP - with Listener support!

Now with ∞ more 5-tuples! With just one CONNECT-UDP connection!



Client

CONNECT-UDP

Proxy

UDP

Target

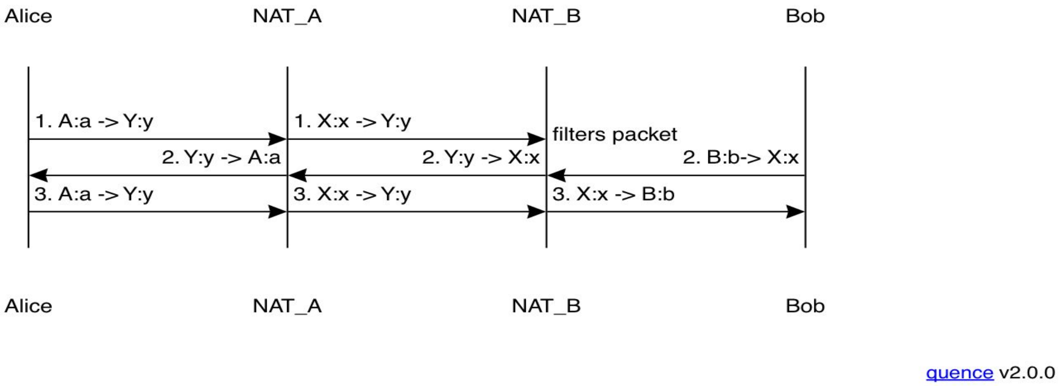
UDP

Target

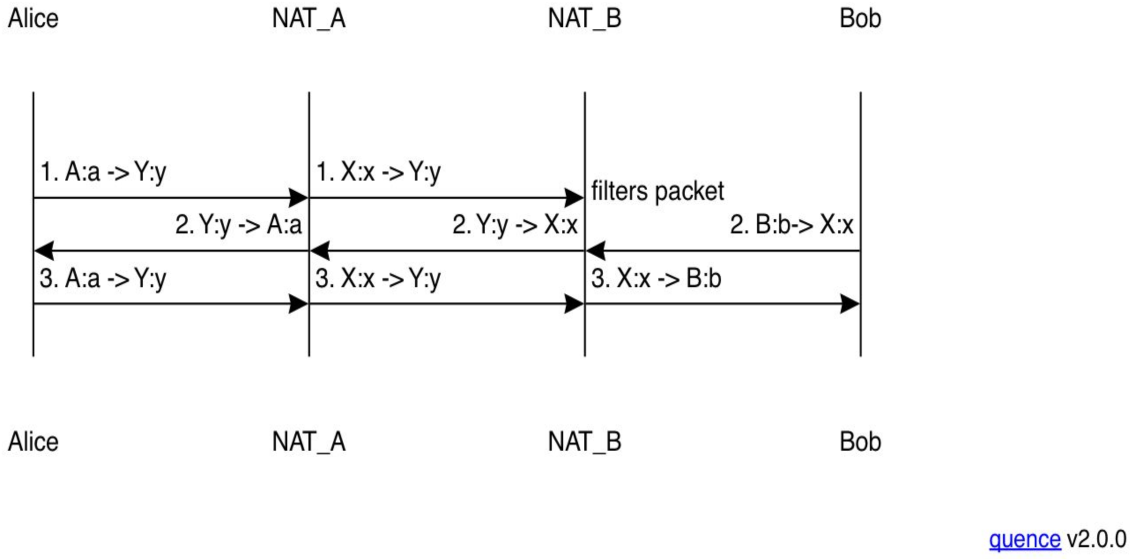
Now we’re a better kind of NAT

- Endpoint independent mapping and Address-port dependent filtering: Use the same connection for multiple clients. Easier to deal with, no relays required

From Eric Rescorla’s blog: establishing P2P connection between two EIM:APF ↔ EIM:APF clients



A Closer look



Notice here how even though Alice’s first connection got rejected, it created a hole in the NAT/Proxy A for Bob, to pass through. And subsequently, Alice will be able to connect to Bob too.

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
```

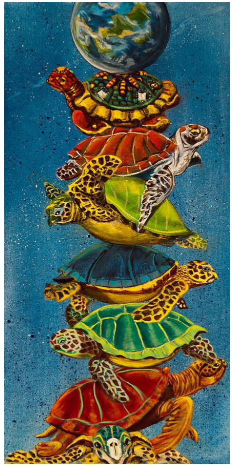
```
DATAGRAM QUIC Frame {
  Type (i) = 0x30..0x31,
  [Length (i)],
  Quarter Stream ID (i),
  Context ID (i) = 42,
  IP Version (8),
  IP Address (32..128),
  UDP Port (16),
  UDP Payload (..),
}
```

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

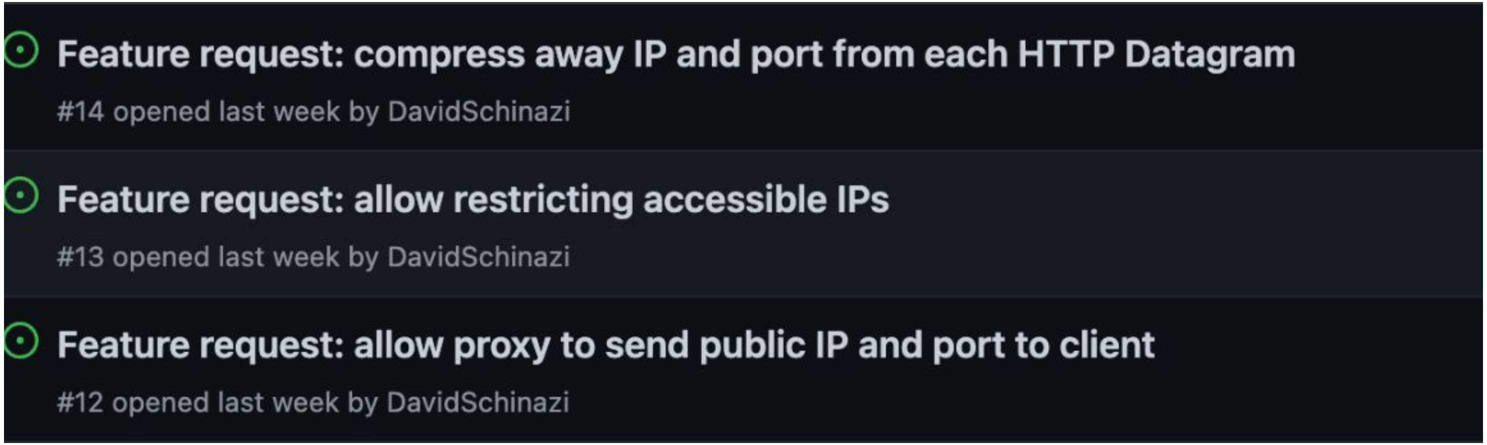
proxy -> client

Source IP/Port PER PAYLOAD

Shall we validate source packets?

Open Issues

https://github.com/DavidSchinazi/draft-schinazi-connect-udp-listen/issues



Feature request: compress away IP and port from each HTTP Datagram

#14 opened last week by DavidSchinazi

Feature request: allow restricting accessible IPs

#13 opened last week by DavidSchinazi

Feature request: allow proxy to send public IP and port to client

#12 opened last week by DavidSchinazi

Feature request: allow proxy to send public IP and port to client #12

https://github.com/DavidSchinazi/draft-schinazi-connect-udp-listen/issues/12

- Ability to retrieve the proxy’s public IP and port for the client.
- WebRTC uses STUN servers to find its reflexive address. Why not have the ability to ask the proxy directly?

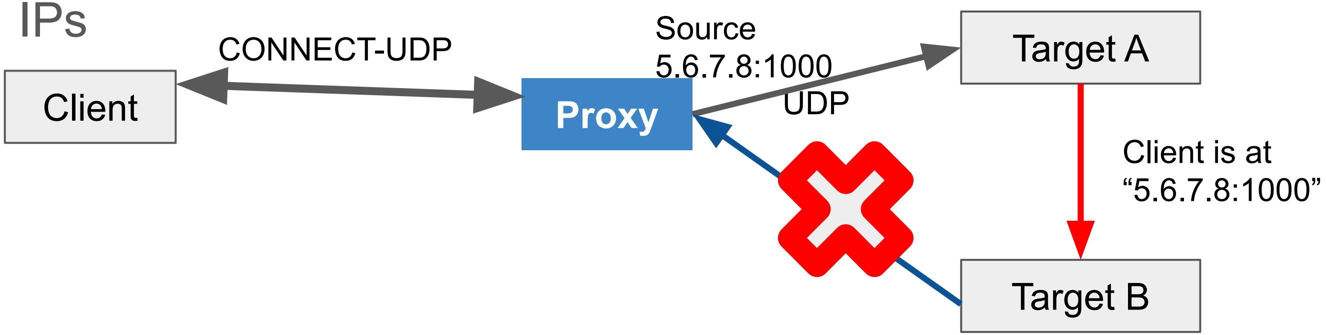
Feature request: allow restricting accessible IPs #13

https://github.com/DavidSchinazi/draft-schinazi-connect-udp-listen/issues/13

- **Question:** Do we want the ability to allow all IPs through? Become a publicly reachable server. Bug or feature?
- Adding support for an allowlist of remote IPs that are allowed to send through the proxy to the client. Similar to “Adding entries to the NAT”
- In TURN, this prevents the TURN server to be used as a server, and instead, simply as a mechanism to enable p2p connections

Address-and-port dependent filtering

- Target B can’t reach Client using information from Target A without an entry in the proxy “firewall” allowlist already.
- Client must initiate communication first with Target B to create an entry in this “firewall”. Or we could create another message to create to specify allowed IPs



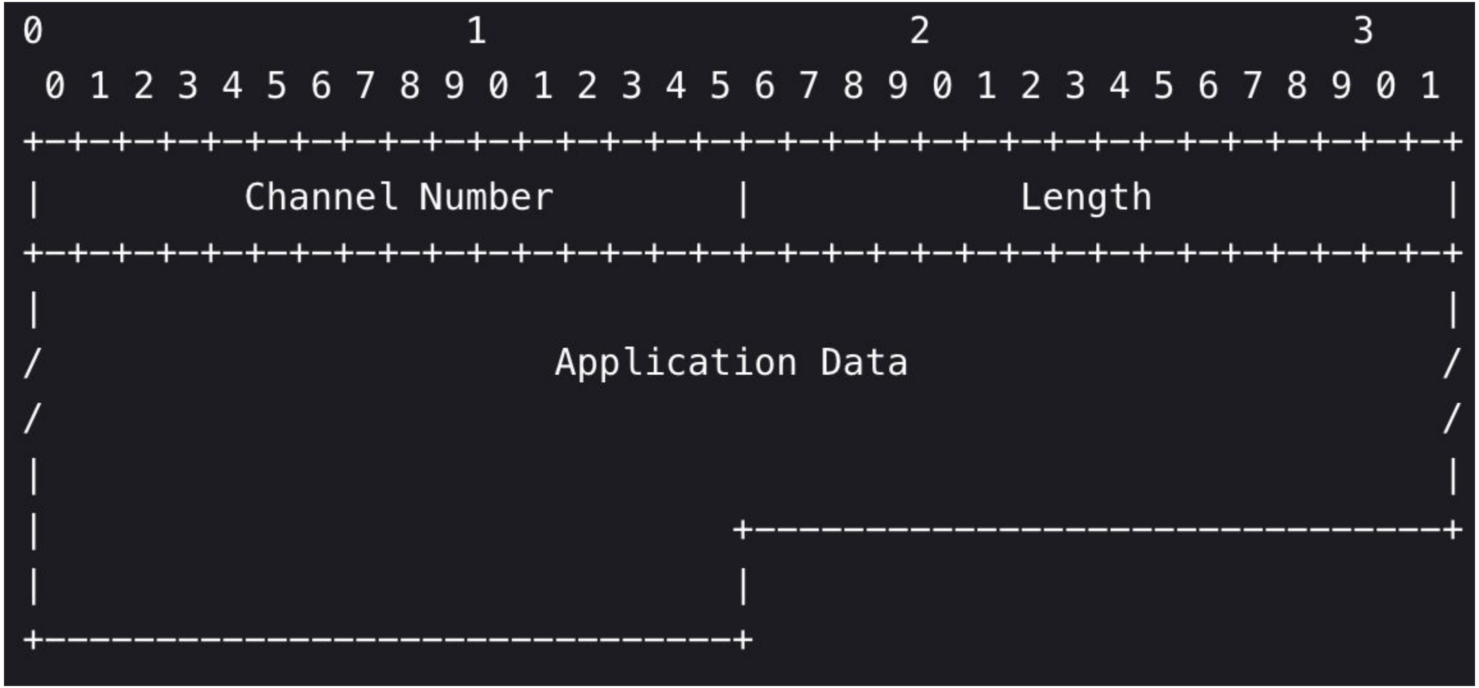
Feature request: compress away IP and port from each HTTP Datagram #14

https://github.com/DavidSchinazi/draft-schinazi-connect-udp-listen/issues/14

- Our proposed header adds 19 bytes of overhead per packet
- For compressed audio formats, it may represent >50% of net bandwidth.
- Need header compression. TURN uses channels for this purpose.

- Could be implemented e.g. by registering a context ID with associated 2-tuple

Example: TURN Channels



```
0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Channel Number        |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
/                       Application Data                        /
/                                                               /
|                                                               |
|                               +-------------------------------+
|                               |
+-------------------------------+
```

MASQUE CONNECT-UDP Listener

draft-schinazi-connect-udp-listen

IETF 117 – San Francisco – 2022-04-24

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