Connection Pooling

for request/response style applications

Philipp S. Tiesel

TAPS

IETF 104, March 2019, Prague

Connection Pooling

- Alternate API interaction scheme for request/response style applications.
- Combine several underlying transport connections into one pooled connection.
- Automated initiation and teardown of additional underlying connections.
- Match request and responses through (local) message references.

API for QUIC

form draft-pauly-quic-interface-00

https://datatracker.ietf.org/meeting/103/materials/slides-103-taps-3a-taps-api-mappings-for-quic-00

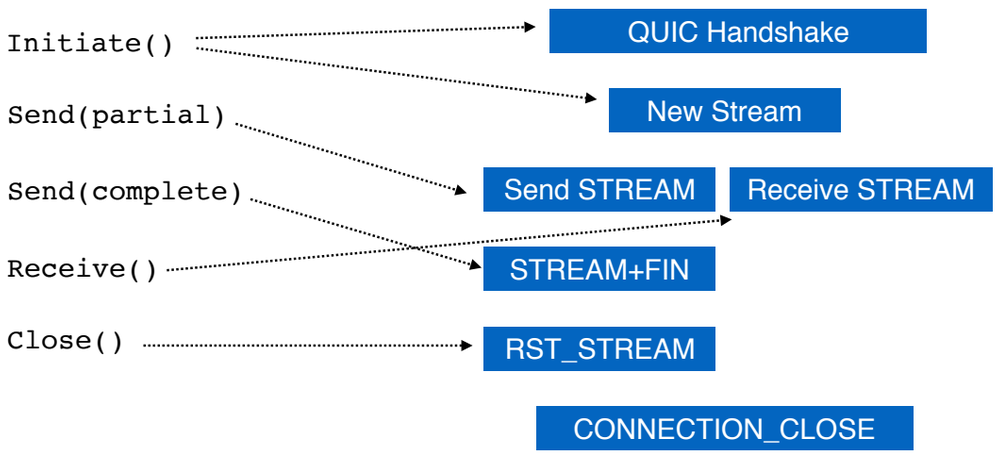
API for QUIC

form draft-pauly-quic-interface-00

https://datatracker.ietf.org/meeting/103/materials/slides-103-taps-3a-taps-api-mappings-for-quic-00

“Stream” Mode

Transport connection as QUIC stream



QUIC Interface Mapping - TAPS - T. Pauly - IETF 103 10

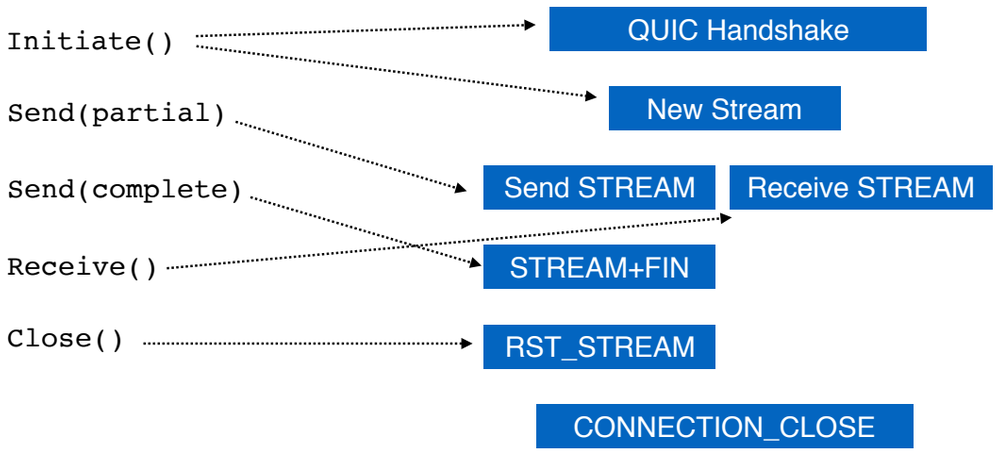
API for QUIC

form draft-pauly-quic-interface-00

https://datatracker.ietf.org/meeting/103/materials/slides-103-taps-3a-taps-api-mappings-for-quic-00

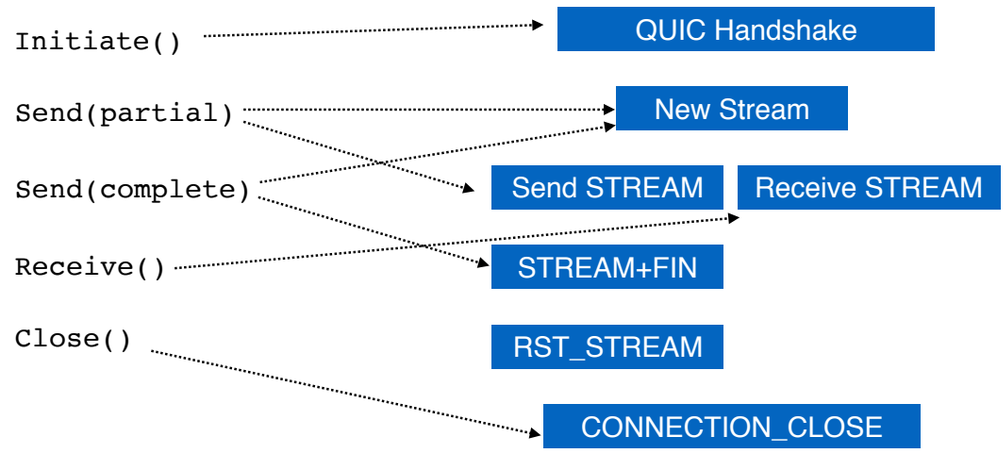
“Stream” Mode

Transport connection as QUIC stream



“Connection” Mode

Transport connection as QUIC connection



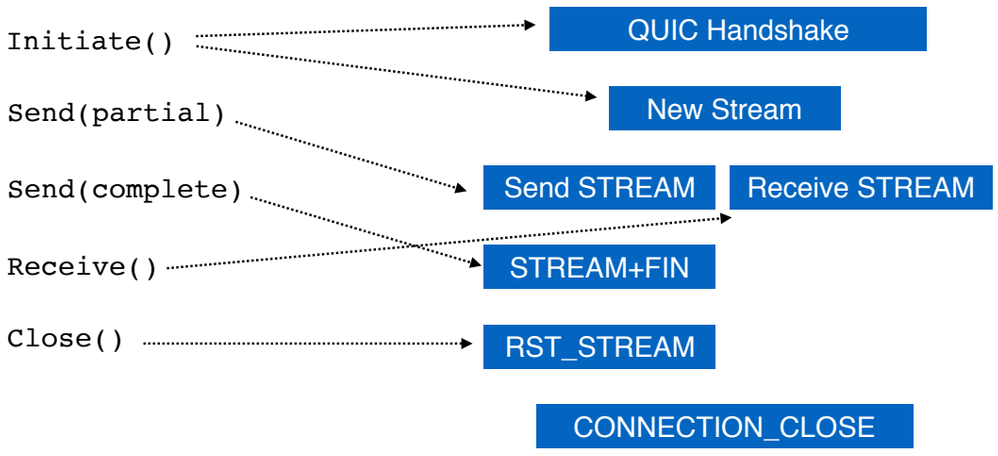
API for QUIC

form draft-pauly-quic-interface-00

https://datatracker.ietf.org/meeting/103/materials/slides-103-taps-3a-taps-api-mappings-for-quic-00

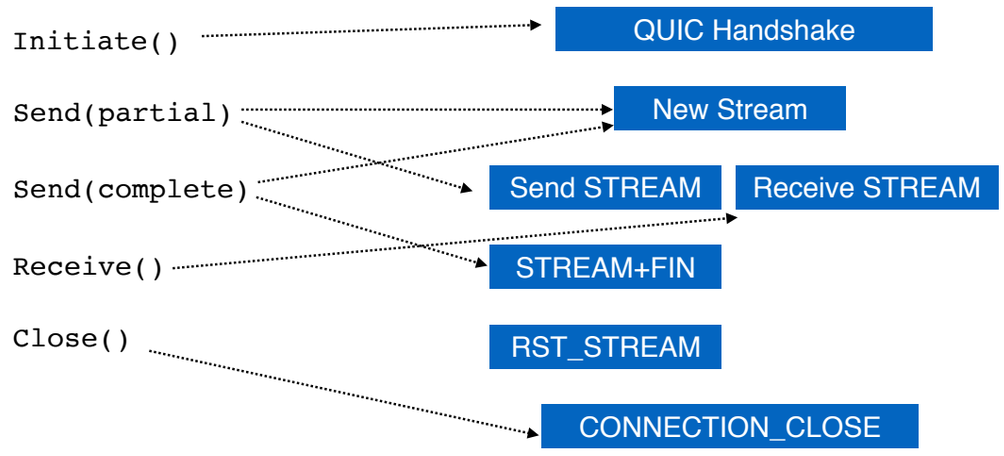
“Stream” Mode

Transport connection as QUIC stream



“Connection” Mode

Transport connection as QUIC connection



Regular Connection

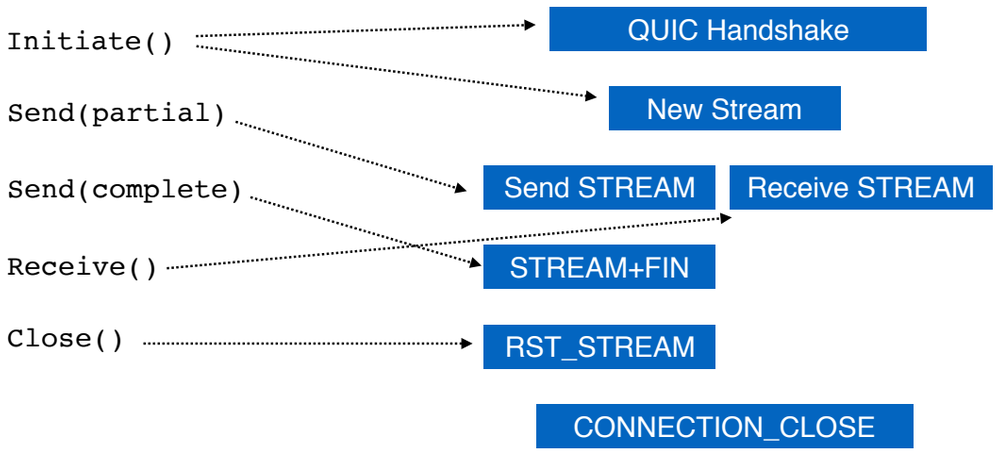
API for QUIC

form draft-pauly-quic-interface-00

https://datatracker.ietf.org/meeting/103/materials/slides-103-taps-3a-taps-api-mappings-for-quic-00

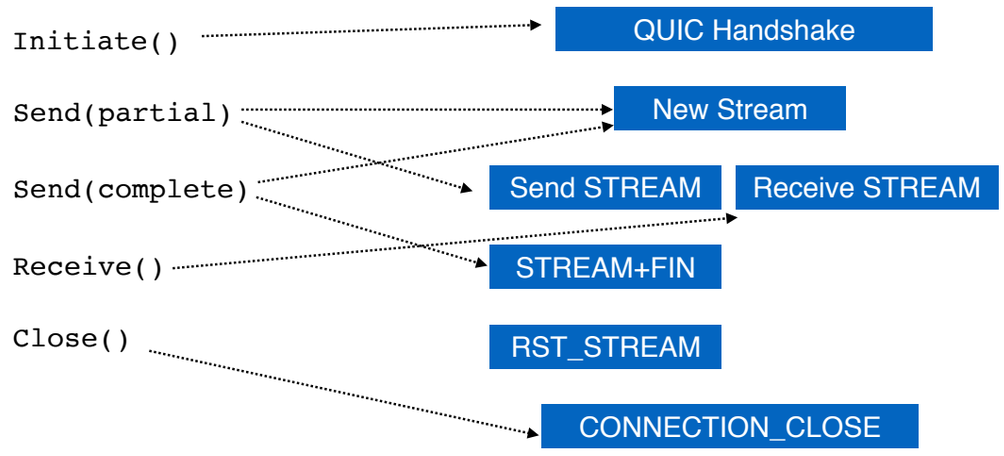
“Stream” Mode

Transport connection as QUIC stream



“Connection” Mode

Transport connection as QUIC connection



Regular Connection

Pooled Connection

Connection Pooling Functionality

- Same API for HTTP/1.1, HTTP/2, and HTTP/3, allows mixing them transparently.
- Automatic management of QUIC Streams for HTTP/3.
- Replacement for unbound UDP sockets.
- Replacement for the “one-to-many” interface for SCTP.
- Enable transparent connection migration.
- Enable per-message path selection.

Connection Pooling for Server Applications

- Same API for HTTP/1.1, HTTP/2, and HTTP/3.
- Automatic management of QUIC Streams.
-

Connection Pooling Variants

- Add Connection Pool Object – PR #295
  - Add Separate Connection Pool object.
  - A TAPS Connections always represents one underlaying transport connection.
- Add Pooled Connections – PR #298
  - Add Selection Property to enable Pooled Connections.
  - Enables a TAPS connection to represent multiple underlaying transport connections.

Connection Pool Example

```
RemoteSpecifier := NewRemoteEndpoint()
RemoteSpecifier.WithHostname("example.com")
RemoteSpecifier.WithService("https")
TransportProperties := NewTransportProperties()
TransportProperties.Require(preserve-msg-boundaries)
TransportProperties.Ignore(preserve-order)
// Security Parameters left out for brevity
```

```
Preconnection := NewPreconnection(None, RemoteSpecifier, TransportPreperties, SecurityParameters)
RequestorPool := Preconnection.RequestorPool()

// no ready event
reqRef := RequestorPool.Send(messageData:=Request, reqRef:=none)

RequestorPool.Receive()
RequestorPool -> Received(messageDataResponse, messageContext, reqRef)

RequestorPool.Stop()
```

Pooled Connection Example

```
RemoteSpecifier := NewRemoteEndpoint()
RemoteSpecifier.WithHostname("example.com")
RemoteSpecifier.WithService("https")
TransportProperties := NewTransportProperties()
TransportProperties.Require(preserve-msg-boundaries)
TransportProperties.Ignore(preserve-order)
TransportProperties.Prefer(pool-connections)
// Security Parameters left out for brevity

Preconnection := NewPreconnection(None, RemoteSpecifier, TransportPreperties, SecurityParameters)
Connection := Preconnection.Initiate()

Connection -> Ready<>
reqRef := Connection.Send(messageData:=Request, reqRef:=none)

Connection.Receive()
Connection -> Received(messageDataResponse, messageContext, reqRef)

Connection.Close()
```