4.1 HTTP/3 拡張仕様と応用、はじめに

この章では、HTTP/3の拡張仕様と応用の説明をします。

拡張仕様とは、HTTP/3に機能追加を行うものです。HTTP/3はHTTP/2同様拡張性を持っています。HTTP/2でも、議論中のものからすでにRFCになっている拡張仕様まであります。

また、議論中のHTTP/3ではありますが、応用としてHTTP/3を活用するプロトコルの議論が行われています。ここではそのうちいくつかを紹介します。

4.2 HTTP/2の拡張フレーム

HTTP/2の拡張仕様ですでにRFCになっているもののうち、HTTP/3でも引き続き使うことが検討されているものについて紹介します。

例えば、下記の拡張仕様があります

- RFC7838 HTTP Alternative Services
  - クライアントがつないでいるWebサービスを別のホスト・ポート・プロトコルで提供できることを通知するのに使う仕様。
- RFC8336 The ORIGIN HTTP/2 Frame
  - サーバが提供できるオリジンを通知する。これによりクライアントどのホストへのリクエストならこのコネクションを再利用できるか認識出るようになる。

これらの仕様ではそれぞれ、ALTSVCフレーム、ORIGINフレームといった独自のフレームやSETTINGSパラメータを定義しています。

これらのフレームをHTTP/3でもそのまま使えるようにフレームタイプ値を確保する提案仕様がでています。

Existing HTTP/2 Extensions in HTTP/3
- https://tools.ietf.org/html/draft-bishop-httpbis-altsvc-quic

4.3 DATAGRAMフレーム

QUICはUDPを利用するプロトコルで、HTTP/3はQUICを利用しています。QUIC上で送受信されるアプリケーションプロトコルのデータは、パケットロスにより欠損しても再送されます。仕様上、パケット番号順にデータの並び直しや、再送を必要としないアプリケーションデータの送信はできません。

しかし、ユースケースとしてそのようなことを行いたい場合があります。このような通信を実現するために、QUICのフレームとしてDATAGRAMフレームという拡張仕様が提案されています。

なお、DATAGRAフレームはQUICのフレームとして定義されていますが、HTTP/3で使う場合はそこにFlow Identifierという識別子を含めることになっており、それを定義する仕様が別途提出されています。（HTTP/3のフレームとしてDATAGRAMフレームが定義されているわけで花ありません）

An Unreliable Datagram Extension to QUIC
- https://tools.ietf.org/html/draft-pauly-quic-datagram

Using QUIC Datagrams with HTTP/3
- https://tools.ietf.org/html/draft-schinazi-quic-h3-datagram

DATAGRAMではトランスポートパラメータとしてmax_datagram_frame_sizeを送りこのパラメータは受信できるDATAGRAMフレームの最大相手に通知します。このトランスポートパラメータは、相手にこの拡張仕様に対応していることを伝える役割も持ちます。

DATAGRAMフレーム自体はデータとその長さのみを持つ単純なフレームです。なお、ストリームには属しません。
- Length
- Datagram Data

DATAGRAMフレーム自体はストリームに属しませんが、HTTP/3上でこのフレームを使う場合は、複数のフローを扱うためにFlow Identifierが付きます。これにより単

ーコネクション上で、複数のフローとしてデータ整合性のないデータを送受信できるようになります。

4.4 WebTransport

WebTransportはWebSocketのようなクライアント・サーバ間の双方向通信においても、QUIC・HTTP/3を活用するために登場した技術です。まだまだ比較的議論が始まったばかりの技術であり、実装としては、GoogleがChromeにおいて実験実装を進めている段階です。

4.4.1 背景と概要

HTTPはHTTPリクエストから開始されHTTPレスポンスが返されます。Webの利用用途が広がる中で、例えばゲームといった双方向にデータを頻繁にやりとりするような用途にHTTPは向いていませんでした。そのようなユースケースをカバーするために、クライアントとサーバ間でコネクションを確立したのちHTTP上で任意のタイミングでどちらからでもメッセージを送信可能にするWebSocketという技術が登場しました。

WebSocketは主にAPIとプロトコルの仕様からなります。APIは、ブラウザ上で動作するJavaScriptからコネクションの確立や読み書きするためのAPI仕様であり、W3Cで定義されていましたが、現在はWhatWGのHTML仕様に含まれています。プロトコルはHTTPのコネクションをWebSocketの通信に切り替える手順及び、その後のメッセージについて定義しています。

API
- HTML Web sockets
  https://html.spec.whatwg.org/multipage/web-sockets.html

プロトコル
- RFC6455 The WebSocket Protocol
- RFC8441 Bootstrapping WebSockets with HTTP/2

その後、QUICやHTTP/3の標準化が進められていく中で、Webの双方向メッセージングにおいてもQUICやHTTP/3を活用したいという話が出てきました。例えば、先述のDATAGRAMフレームを活用し、パケットロスしたアプリケーションデータを再送不要な通信を行いたいなどです。そこでWebTransportとして、JavaScript API、プロトコルを新しく定義するという流れにつながっていきます。

現在の仕様は、API仕様はW3CのWeb Incubator Community Groupで、プロトコルはIETFのWebTransport WGで連携をとりながら議論が進められています。なお、WebTransport WGは2020年に出来たばかりの新しいWGになります。

プロトコルについては現在、QUIC、HTTP/3を使うもの、フォールバックとしてTCPのHTTP/2を使うものが提案仕様として提出されています。
API

- https://wicg.github.io/web-transport/

プロトコル

- OverView
  https://tools.ietf.org/html/draft-ietf-webtrans-overview
- WebTransport over HTTP/3
  https://tools.ietf.org/html/draft-vvv-webtransport-http3-01
- WebTransport over QUIC
  https://tools.ietf.org/html/draft-vvv-webtransport-quic-01
- WebTransport using HTTP/2
  https://tools.ietf.org/html/draft-kinnear-webtransport-http2-00

もちろんどのQUIC, HTTP/3, HTTP/2のどれを使うかによって、機能には差が出てきます。例えば、HTTP/2ではTCPなのですべてのアプリケーションデータは再送・整列されます。OverViewのドキュメントではWebTransportの特性としては4つをあげており、使う下位プロトコルによってその特性をサポートするか否かが異なります

- Stream independence: ストリームの独立性。ストリーム間でhead of line blockingがない特性
- Partial reliability: 部分的な信頼性。パケットの欠損や並び替えが起こっても、アプリケーションデータの再送および整列は行わないという特性
- Pooling support: コネクションプーリング。複数の通信で1つのコネクションを共用することができる特性
- Connection mobility: コネクションのモビリティ。コネクションマイグレーションを行える特性

ここでは、プロトコルについて深ぼりしていきます。ただ、まだWGとして正式に採用されていない仕様になりますので、ここから大きな変更が加えられる可能性はある点注意ください。

4.4.2 WebTransport over QUIC

QUICを利用するWebTransportで、QUICコネクション上でアプリケーションデータをやりとりします。WebTransportの特性の観点では、QUICを直接利用する場合はPooling supportが無しとなります。

- Stream independence: あり
- Partial reliability: あり
- Pooling support: なし
- Connection mobility: 実装依存

WebTransport over HTTP/3では1つのコネクション上で、一部のストリームはHTTP/3用、また一部のストリームはWebTransport用に使用することができます。WebTransport over QUICでは、確立されたQUICコネクションはWebTransport専用になります(ALPNとしてwq, draft版ではwq-vvv-01が使用されます)。双方向ストリーム・単方向ストリームでもアプリケーションデータを送受信できます。また、ストリームに属さないDATAGRAMフレームも送信できます。

WebTransport over QUICの場合は、ブラウザで動作するJavaScriptからWebTransportを利用す際は下記のようにURIを指定します。

```
quic-transport://host:port/path
```

アプリケーションデータをやり取りする前に準備として、コネクションが確立した直後、ストリームID2上でClient Indicationというメッセージをクライアントからサーバに送信します。この Client Indicationには2つの情報をサーバに伝達します。

- Path: アプリケーションから指定された、URIのpath。特に利用用途は指定されていないが、アプリケーションが独自に利用して良い。
- Origin: このWebTransportを利用したWebページのオリジンが入る。CORS相当の処理を行うために送られる。

例えば、https://example.comからquic-transport://wq.example.com:4433/room_1に対して接続を開始した場合、接続を開始したoriginとしてhttps://example.com:443サーバに通知されます。こうすることで第三者のWebページにWebTransport over QUICサーバを勝手に利用されるということはなくなります。

このClient Indicationが送信されたあとは、QUICのストリーム上でDATAフレーム、もしくはDATAGRAMフレームでアプリケーションデータの送受信が行われるようになります。

4.3.3 WebTransport over HTTP/3

TODO

4.5 MASQUE

TODO

4.6 RIPT

TODO