https://github.com/flano-yuki/http3-note
ver 0.0.1 (2020/06/01)

1.1 はじめのはじめに

HTTP/3という新しいHTTPバージョンの標準化が進められています。2020年、すでにブラウザやミドルウェアの実験実装が進んでおり、すでに利用できる段階がすぐ目の前まで来ています。

HTTP/1.1から始まりHTTP/2へ、そしてHTTP/3とHTTPメッセージをより効率よく送受信できるようにプロトコルが改良され、バージョンが進んできています。新しいバージョンであるところのHTTP/3はもちろん、さまざまな改良が含まれております。

大きな特徴としてトランスポートプロトコルとしてQUICを利用しますが、その利点を享受できるように様々な工夫がされています。

もちろん、ユーザはブラウザが対応すれば気づくことさえなくHTTP/3を利用できますし、サービス提供者側もクラウドサービスやミドルウェアが対応すればその詳細をきにすることなく利用できます。

しかし、HTTP/3及びQUICの話は奥が深く興味深いものです、おもにプロトコル部分にフォーカスして自分の理解を整理しつつ文章をまとめていこうとおもいます。なお、各種サンプル実装のクライアントやサーバのログをみたり、Wiresharkで通信を復号したり、qlogといった可視化ツールを利用することをおすすめします。

> INFO
>
> 現状、HTTP/3およびQUICの仕様は Draftレベルであり正式にRFCが出るまでに変更される可能性があります。本文書は執筆中の仕様を参考にしています。
>
> 最新仕様は下記から参照できます。
> - https://github.com/quicwg/base-drafts

1.2 HTTPのセマンティクスとバージョンの話

HTTP/3の解説の前に、HTTPとバージョンについて説明します。

HTTPの仕様にはいくつかのバージョンがあります。主に使われているのは HTTP/1.1, HTTP/2, HTTP/3があり、これらの違いはHTTPメッセージの伝え方が異なるという点です。逆に言えば、HTTPメッセージのセマンティクス(意味)は変わっていないということです。

HTTPメッセージとは、HTTPリクエストとHTTPレスポンスの事です。HTTPリクエストにはメソッドやHTTPリクエストヘッダがあり、POSTメソッドの場合には HTTPボディもあります。HTTPレスポンスにはステータスコードがあって、HTTP レスポンスヘッダ、HTTPボディがあります。

これらのHTTPメッセージの意味はHTTPバージョンに関わらず一緒です。この HTTPメッセージをより効率よくやりとりするために、各バージョンでは送り方が異なるということです(一部例外はあります)。
(細かい用語は以後の章で説明されます)

- HTTP/1.1ではTCP上でASCII文字のまま送受信されます
- HTTP/2ではコネクション上の仮想的な通信単位であるストリームでフレームという形式で送受信されます
- HTTP/3ではUDP上で動作するQUICトランスポートを使い、そのストリームでフレームという形式で送受信されます。

このように転送の形式(かたち)のみが異なり、送受信するHTTPリクエストおよび HTTPレスポンスの意味は変わっていません。そのためバージョンを意識することなく、ブラウザ上で動作するJavaScriptや、サーバ上で動作するアプリケーションは今まで通りHTTPリクエストやHTTPレスポンスを処理することができます。

仕様上の話をすると、HTTPのセマンティクス定義は現状HTTP/1.1の仕様の一部として定義されています。

- RFC 7230 - Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing
- RFC 7231 - Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content
- RFC 7232 - Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests
- RFC 7233 - Hypertext Transfer Protocol (HTTP/1.1): Range Requests
- RFC 7234 - Hypertext Transfer Protocol (HTTP/1.1): Caching
- RFC 7235 - Hypertext Transfer Protocol (HTTP/1.1): Authentication

これらの仕様は以下の3つの仕様に再整理する議論を現在行っています。

- HTTP Semantics
    - https://tools.ietf.org/html/draft-ietf-httpbis-semantics
- HTTP/1.1 Messaging
    - https://tools.ietf.org/html/draft-ietf-httpbis-messaging
- HTTP Caching
    - https://tools.ietf.org/html/draft-ietf-httpbis-cache

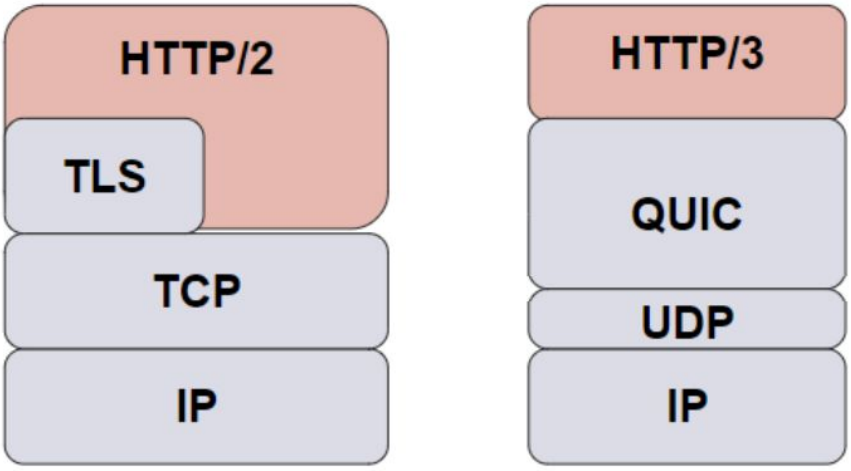
これにより、HTTPセマンティクスと、HTTP/1.1のシンタックスの仕様は分離されます。HTTP/3の仕様からは整理されたHTTPセマンティクスの仕様を参照するかたちとなるでしょう。

1.3 HTTP/3の概要

ここではHTTP/3について簡単な概要を説明します。聞き慣れない用語が出てきますが、後の章で説明されます。

HTTP/3はトランスポートプロトコルとしてQUICというプロトコルを使用しています。QUICはUDP上で動作するプロトコルであり、TCPのようなデータの整合性(パケットロスしたデータの回復、データ順番)を担保し、TLSのようにアプリケーションプロトコルレイヤの暗号化を行います。

HTTP/2とHTTP/3を比較してスタック図とすると下記のようになります。QUICでは暗号化しない通信方法は定義されておらず、必然的にHTTP/3では通信は暗号化され、https:// のURLのときに使用されます。また、使用する番号は決まっていませんが、一般的にはUDP 443番が使用されます。



HTTP/2とHTTP/3のスタック図

先述の通り、QUICはUDP上で動作します。このQUICを使う一つのメリットとして画像Aと画像Bをダウンロードする例で説明します。

HTTP/2では一つのTCPコネクション上で画像AのHTTPリクエストと画像BのHTTPリクエストを並列的に投げることができました。これは輻輳制御上複数のTCPコネクションを利用するより、より効率的にネットワーク待機を利用できるようになりました。

しかし、例えばこのとき画像Bのレスポンスデータを含むパケットがロスした場合、OSレイヤで欠損したパケットを回復してからアプリケーションに渡されます。仮に後続の画像Aのレスポンスデータを含むパケットを受信できていたとしても、ロスしたパケットが再送されるまでアプリケーションでは処理することができません。

HTTP/3では、UDP上で動作するQUICを使用します。UDPパケットをアプリケーションレイヤで扱いますので、画像Aと画像Bを並列的にリクエストした際に、画像Bのレスポンスデータを含むパケットをロスした場合でも、後続の届いたパケットが処理可能であれば処理をすすめることができます。

このようにHTTP/3では届いたパケットが処理可能であれば処理するという特徴があります。

その他にもHTTP/3は、HTTP/2で出てきた仕組みを受け継ぎつつ、ストリームの管理やウィンドウ制御といった一部の機能はQUICレイヤで吸収されています。

また、HTTP/2ではTCPを利用していたので送信者が送信した順番に受信者が処理をすすめるという前提がありました。例えば、HTTP/2のもつHTTPヘッダ圧縮の仕組みHPACKや、優先度制御機能はそのような前提のもと動作していました。HTTP/3ではパケットロスやパケットの入れ替わりがあるため、そのような前提のもと動作すると、それらが回復されるまで待ち状態になってしまします。先述のように届いたパケットから処理可能となるように、例えばHPACKはQPACKという新しい仕組みで起き直されています。

このように、HTTP/3では、トランスポートプロトコルとしてQUICを効率よく使うための特徴を多く持っています。第3章では、HTTP/3の以下の項目について説明していきます。

HTTP/3
- QUIC ストリームの利用
- HTTP/3フレーム
- コネクションの開始・切断
- ヘッダ圧縮 (QPACK)

- サーバプッシュ
- 優先度制御の扱い
- エラー処理
- 拡張性

また、HTTP/3の機能はQUICの理解が必要不可欠ですので、関連する部分に限定して2章で説明します。QUICについてはHTTP/3と併せて読むことをおすすめします。適宜、2章と3章を往復すると読みやすいと思います。

QUIC
- QUICの特徴・機能
- QUICコネクションおよびその確立
- ストリームとフレーム

4章ではその他、QUICやHTTP/3を利用したプロトコルや拡張仕様について触れていいきます。
- 拡張フレーム
- DATAGRAM
- WebTransport

1.4 HTTP/3 と呼ばれるまでの道のり

HTTP/3の標準化は、IETF(Internet Engineering Task Force)のQUIC WGで行われています。QUIC WGではQUICの標準化を進めているワーキンググループです。QUIC自体はトランスポートプロトコルであり、アプリケーションプロトコルはHTTPに限定されませんが、まずはHTTP/3の標準化にフォーカスして進められています。

少々このHTTP/3という名称と、標準化までの流れをざっと振り返ってみたいと思います。

大まかに分けて大まかに下記のように進んできました。
- 2013年~2016年 Google QUIC
- 2016年~2018年 HTTP over QUIC
- 2018年~ HTTP/3

もともとQUICはGoogle社が実装・実験していたプロトコルですが、IETF版のQUICとは多くの点で異なっています。ここでは、明確に区別するためにGoogle QUIC、IETF QUICと呼び分けることにします。なお、Google QUICは当時の資料をみると、Quick UDP Internet Connections の略語としてQUICとしていましたが、IETF QUICでは「QUIC is a name, not an acronym」と、何かの略語ではないと書かれています。

Google QUICは過渡期のものであり、Google社もQUICの標準化には積極的に参加しており、彼らの利用しているGoogle QUICも徐々にIETF QUIC版にしていくとコメントしています。実際にその作業は進められ標準化されるHTTP/3のサポートを目指しています。

1.4.1 Google QUICの実験

Googleは2013年頃より、QUICというプロトコルを実装・実験していました。これがGoogle QUICと呼ばれているものです。Google QUICもUDPを使用して、HTTPのメッセージを効率よくやり取りするためのプロトコルです。

ただし、今現在IETFで標準化されているIETF QUICとは異なるもので、トランスポートレイヤの機能から暗号化、HTTPレイヤまでをひとくるめにしたプロトコルでした。

GoogleはGoogle Chrome及びGoogleのサービスで実験を行い、その効果を確かめていました。その後、2015年に開催されたIETFのオフラインミーティング (IETF93)会期中に、有志で開催された非公式の集まり(BarBoF)にて、その成果が発表されました。当時の発表資料は下記URLから閲覧できます。
https://docs.google.com/presentation/d/15e1bLKYeN56GL1oTJSF9OZiUsI-rcxisLo9dEyDkWQs/edit#slide=id.gaf8802b44_0_10

その後、このQUIC (Google QUIC)というプロトコルは多くの興味を惹きつけました。

翌2016年には、提案仕様という形で下記のDraftが提出されています。

- QUIC: A UDP-Based Multiplexed and Secure Transport
    - https://tools.ietf.org/html/draft-hamilton-quic-transport-protocol-00
- HTTP/2 Semantics Using The QUIC Transport Protocol
    - https://tools.ietf.org/html/draft-shade-quic-http2-mapping-00

この段階では、まだHTTP/3とは呼ばれていませんでした。これらが標準化を行っていくかの議論を開始するための初期草案であり、のちのちIETF QUIC、HTTP/3と呼ばれることになります。

1.4.2 HTTP over QUIC、標準化の開始

2016年7月に行われた IETF 96では、QUIC BoFというQUICに興味を持つ人があつまる正式なセッションが開催され、QUICの標準化をするモチベーションが確認され

ました。2016年11月のIETF 97では、QUICの標準を進めるためQUIC WGが設立されました。

このときに、IETF QUICはトランスポートプロトコルであり、その上に乗せるアプリケーションプロトコルとしてまずHTTPにフォーカスする事になりました。

戦術の初期草案はWGの正式な取り扱い仕様(WG Draft)となり、HTTP over QUICと呼ばれるようになりました。

- Hypertext Transfer Protocol (HTTP) over QUIC
  - https://tools.ietf.org/html/draft-ietf-quic-http-00

ここからQUIC、HTTP over QUICの標準化の議論が進められていきます。

1.4.3 HTTP/3への改称

2018年11月に行われたIETF103において、HTTP over QUICをHTTP/3と改称する議論が行われました。これは、名称の変更のみで、プロトコルとして大きな変更はありません（一部パラメータ名の変更はありました）。

この議論での意見をいくつか紹介すると下記のとおりです
- もともとGoogle QUICがHTTPレイヤも含めのプロトコルであったこともありQUICという名称に関して紛らわしさがある
- HTTP/2 over QUICではないのでそう呼ばないでほしい
- 明確にHTTP/2よりも新しいことを示す名前にしたい

(https://github.com/quicwg/wg-materials/blob/master/ietf103/http.pdf)

議論の後、QUIC WGの中でHTTP/3と呼ぶことに関してコンセンサスが得られました。2018年12月に提出された 17版目のDraftでは名称がHTTP/3になっていることが確認できます。

Hypertext Transfer Protocol Version 3 (HTTP/3)
- https://tools.ietf.org/html/draft-ietf-quic-http-17

1.5 標準化動向を追うために

QUICやHTTP/3のRFCはまだ出ていません、大枠はすでに固まっているとはいえ引き続き細かい修正が行われています。また、実装や様々な実験を通して仕様側にフィードバックが行われています。

仕様の議論は先述の通りIETF QUIC WGで行われています。その議論はメーリングリスト及びオフラインのミーティングで行われています。これらの内容はすべて閲覧できるようになっています。

GitHubリポジトリ

- メインの仕様 https://github.com/quicwg/base-drafts
- 議事録・発表スライド https://github.com/quicwg/wg-materials

メーリングリスト

- https://mailarchive.ietf.org/arch/browse/quic/

最新動向に興味のある方は、まず仕様をみつつ、GitHubのコミットログとメーリングリストの議論を追いかけるのが良いと思います。また、GitHubのQUIC WG Organizationにはその他の仕様も管理されています。興味があれば御覧ください。

また、実装状況は刻々と変わるものではありますが、下記をご参考ください。

Implementations

- https://github.com/quicwg/base-drafts/wiki/Implementations