# QUICをRubyの世界に持ってくる活動をしています

unasuke

鹿児島Ruby会議02

2023-03-04


鹿児島 Kagoshima Ruby Community Ruby会議 02

# 自己紹介

- Name: うなすけ
- Work: フリーランス
- Ruby歴: 8年くらい
- Kaigi on Rails オーガナイザー
- GitHub https://github.com/unasuke
- Mastodon https://mstdn.unasuke.com/@unasuke
- Twitter https://twitter.com/yu_suke1994

# 今日話すこと

- QUICとは何か
- 具体的にやっていること
- 今後の展望

# QUICとは何か


image from https://github.com/quicwg/wg-materials

# QUICとは何か

- 2021年にRFC 9000を含むいつくかのRFCにより標準化
- TCPではなくUDPによって通信をするプロトコル
  - これまでのTCP(HTTP)より高速 ← ？
- HTTP/3はQUICを使用する

# 「これまでのTCP(HTTP)より高速」とは？

ざっくりと……

- HTTP/2はTCPを使う
  - TCP: ”Transmission Control Protocol”
    - 信頼性が高いがオーバーヘッドが大きい
  - 途中のパケットが欠損すると、再送されるまで処理が止まる
- HTTP/3はUDP(QUIC)を使う
  - UDP: ”User Datagram Protocol”
    - オーベーヘッドがないぶん高速だが信頼性は落ちる
  - 途中のパケットが欠損しても、影響がない範囲で処理を継続する

ref: https://eng-blog.iij.ad.jp/archives/11166

# TCPとUDP

- TCPは信頼性が高いがオーバーヘッドが大きい
  - HTMLとかJavaScriptとかCSSとか
    - 欠損したら困る
- UDPはオーベーヘッドがないぶん高速だが信頼性は落ちる
  - VoIP、RTPなどの音声、ビデオ通話とか
    - 完全性よりリアルタイム性

しかしHTTP/3はQUICを使うし、QUICはUDP。これはどういうことか？

# QUICがやっていること

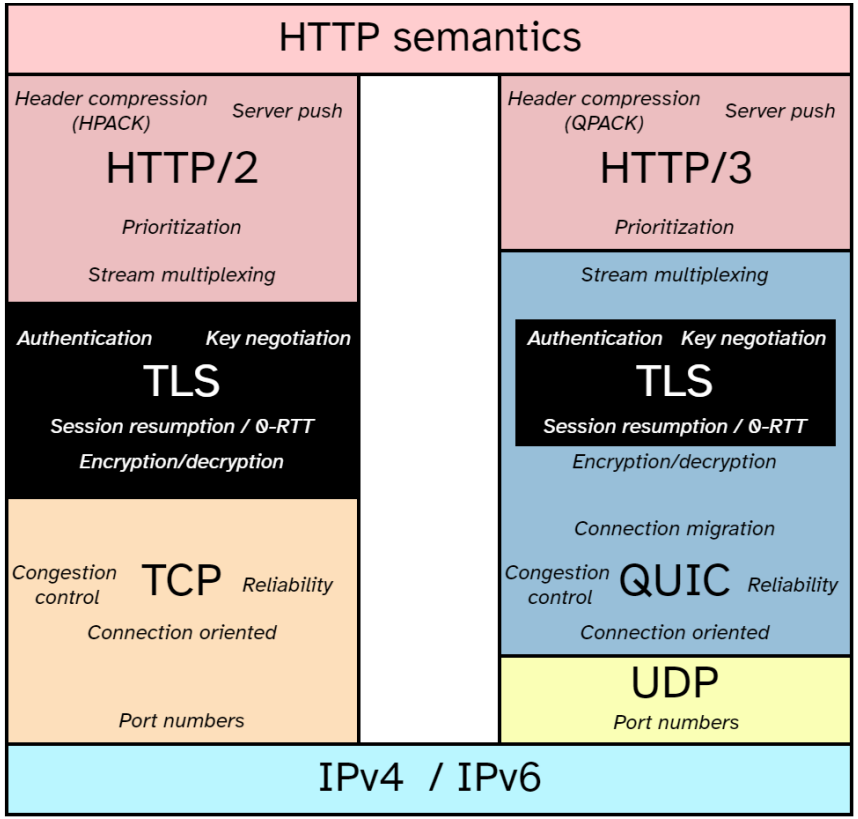



image from https://github.com/rmarx/h3-protocol-stack

# QUICとは何か、詳しくは……

QUICをゆっくり解説 – 新しいインターネット通信規格｜IIJ Engineers Blog

# 今日話すこと

- QUICとは何か
- 具体的にやっていること ←
- 今後の展望

# 具体的にやっていること

2022年までのぼく

「よ〜し、RFC読んでQUICの実装するぞ〜！」

# 具体的にやっていること

2022年までのぼく

「よ～し、RFC読んでQUICの実装するぞ～！」

# 大挫折

# 大挫折2022

- 例えば「よ～しオレオレpuma作るぞ～」がうまくいくのか？
  - 「オレオレRails」でも可
  - → 無理

やったことがないことをやるのは大変

# 大挫折2022

- 例えば「よ～しオレオレpuma作るぞ～」がうまくいくのか？
  - 「オレオレRails」でも可
  - → 無理

やったことがないことをやるのは大変
→ じゃあ「やったことあること」をする、のはどうだろう

# 大挫折2022

- 例えば「よ～しオレオレpuma作るぞ～」がうまくいくのか？
  - 「オレオレRails」でも可
  - → 無理

やったことがないことをやるのは大変
→ じゃあ「やったことあること」をする、のはどうだろう
→ じゃあ「QUICの実装」を「やったことある」状態にするにはどうしたら？

# この世界にあるQUICの実装

- lsquic (C言語)
- quicly (C言語)
- quiche (Rust)
- quic-go (Go)
- aioquic (Python)

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

# この世界にあるQUICの実装

- lsquic (C言語)
- quicly (C言語)
- quiche (Rust)
- quic-go (Go)
- aioquic (Python) ←

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

PythonはRubyに似てるらしい……

# この世界にあるQUICの実装、にRubyを加える

QUICのPython実装であるaioquicをRubyに移植しよう！

# 採択

## RubyによるQUICプロトコルの他言語からの移植ならびに独自実装の作成

### プロジェクト概要

2021年に標準化されたインターネットプロトコルであるQUICの利用は急速に広まっており、様々なプログラミング言語による実装が盛んである。中には複数の実装があるプログラミング言語も存在する。しかし現時点で公になっている、Rubyによる実装は存在していない。
このプロジェクトでは、最終的にRubyによるQUICプロトコルの実装を作成することを目指す。まずはその前段階として、Rubyに似たPythonによるQUIC実装である、aioquicをRubyに移植することで、QUIC実装の指針、知見を構築する。

### 応募者名

unasuke (Yusuke Nakamura)

https://www.ruby.or.jp/ja/news/20220823

# ねらい

- 締切
  - 1/16 中間報告書提出締切
  - 3/20 最終報告書提出締切
- メンター

# やっていること

ここまでが「やっている」までの経緯

# aioquicを移植する

## aiortc/**aioquic**

QUIC and HTTP/3 implementation in Python


20 Contributors　7 Issues　18 Discussions　1k Stars　179 Forks

https://github.com/aiortc/aioquic

# aioquicを移植する、その規模感

```
> tokei
===============================================================================
 Language            Files        Lines         Code     Comments       Blanks
===============================================================================
 Autoconf                1            4            4            0            0
 C                       2         1025          751          128          146
 CSS                     1           10            9            0            1
 HTML                    2           63           63            0            0
 JSON                    1            3            3            0            0
 Makefile                1           20           10            6            4
 Python                 55        22918        18483         1230         3205
 ReStructuredText        8          463          300            0          163
 SVG                     1           72           72            0            0
 Plain Text              2            5            0            5            0
 TOML                    1            2            2            0            0
===============================================================================
 Total                  75        24585        19697         1369         3519
===============================================================================
> 
```

# aioquicの構造

- Buffer
- Crypto
- TLS
- QUIC
  - Packet
  - PacketBuilder
  - Rangeset
  - Configuration
  - Retry
  - Crypto
  - Stream, Connection...

# aioquicの構造

- Buffer ← まあ……
- Crypto ← やばい
- TLS ← やばすぎ
- QUIC
  - Packet ← まあ……
  - PacketBuilder ← まあ……
  - Rangeset ← easy
  - Configuration ← easy
  - Retry ← まあ……
  - Crypto ← やばい
  - Stream, Connection... ← やばすぎ

# ”やばすぎ” mean...

- 行数？
  - tls.py（1445行、テスト1163行）
  - connection.py（2774行、テスト2171行）
- 行数は真のやばさではない

# QUICがやっていること(再)

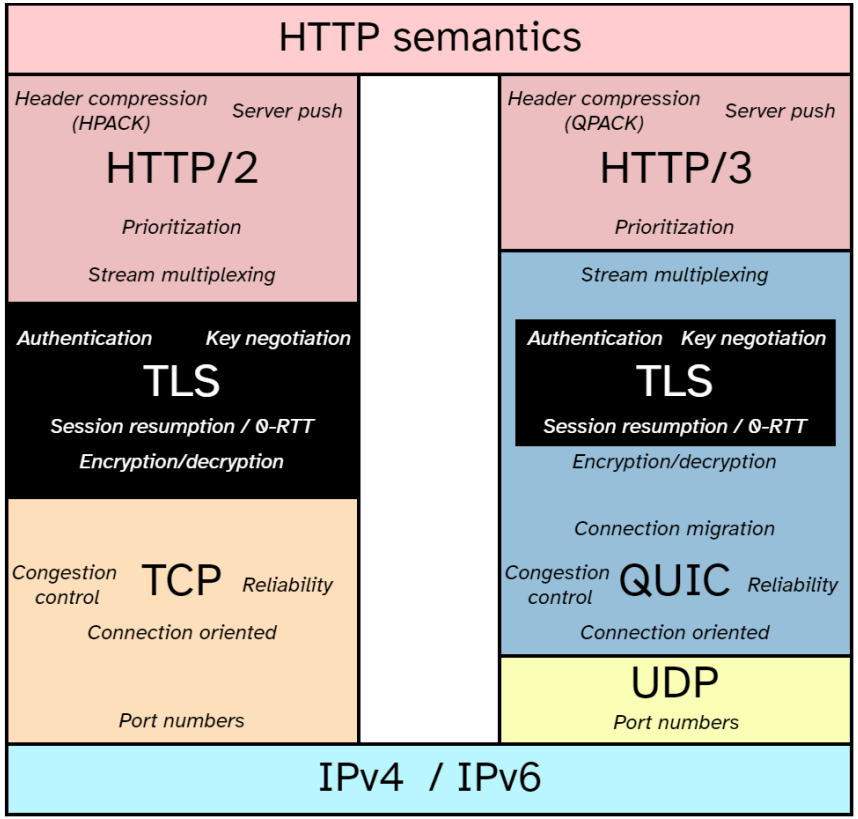



image from https://github.com/rmarx/h3-protocol-stack

# ”やばすぎ” mean...

- QUICは暗号化も受け持つ
  - つまり、TLSを実装する
    - 鍵交換
    - 認証
    - etc
- 扱うデータの中身も暗号化されている
  - OpenSSL「こんにちは～」
  - PythonとRubyでOpenSSLのAPIの抽象度合いが異なる

# で、どこまでできてるの？

- connection.py の移植の真っ最中
- Rubyishにするにはどうすればいいかの設計

# デモ

- curlから送信されたQUICのPacketを解析する

# 今日話すこと

- QUICとは何か
- 具体的にやっていること
- 今後の展望 ←

# 今後の展望

- QUIC自体
- 移植プロジェクト

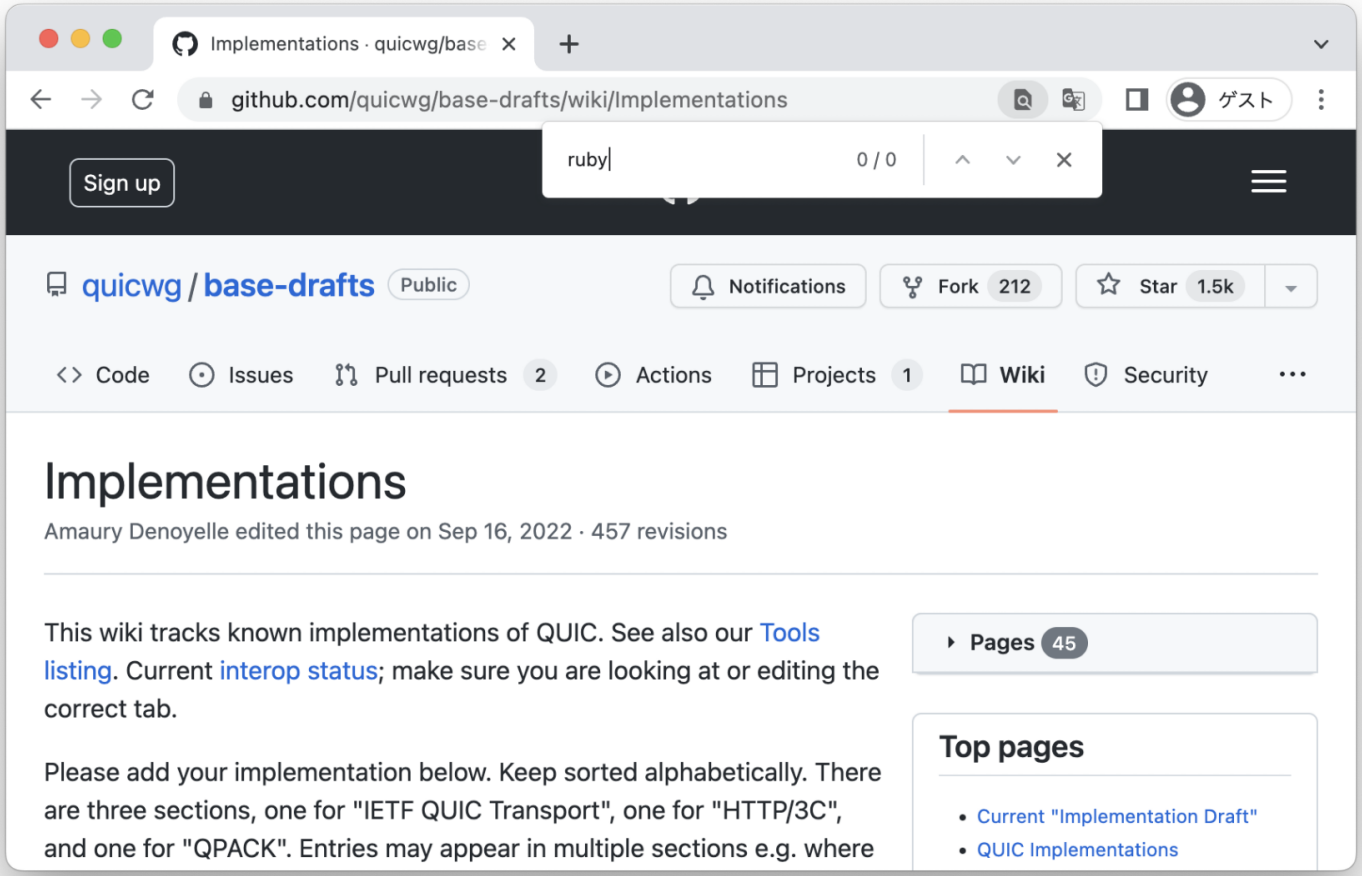
# 今後の展望 - QUIC

- QUICは様々な場所で使われつつある
  - Media over QUIC
    - Warp（Twitch）
  - WebTransport
- QUIC v2が策定中
  - https://datatracker.ietf.org/doc/draft-ietf-quic-v2/
  - 3月末 IETF 116 Yokohama
- https://datatracker.ietf.org/wg/quic/about/

# 今後の展望 - 移植プロジェクト

- aioquicの移植を終わらせる
  - 3/20 最終報告書提出期限
- ”Rubyishとは何か？”
  - 設計、API、etc...

# 今後の展望、そもそもなぜこんなことを？



# 現実問題

- お客様の中にpumaでHTTPSの通信をされている方は
  - TLSを終端するのは誰なのか

# 現実問題

- AWS CloudFront
  - https://aws.amazon.com/about-aws/whats-new/2022/08/amazon-cloudfront-supports-http-3-quic/
- Google Cloud Load Balancing
  - https://cloud.google.com/load-balancing/docs/https#QUIC
- NGINX
  - https://www.nginx.com/blog/binary-packages-for-preview-nginx-quic-http3-implementation/
- H2O
- Apache

# 現実問題への回答、RubyがQUICを喋る理由

- Just for Fun
- そこにRubyがないから
- 副産物
  - OpenSSL gemの修正
    - https://github.com/ruby/openssl/pulls?q=is%3Apr+author%3Aunasuke (現在3つ)
  - OpenSSL gemのドキュメント(るりま)が古い
    - 思案中
  - RDoc内の回遊を楽にしたい
    - https://github.com/ruby/rdoc/pull/973

# まとめ

- QUICをRubyの世界に持ってくる活動をしています

```
> tokei
===============================================================================
 Language            Files        Lines         Code     Comments       Blanks
===============================================================================
 BASH                    1            8            4            2            2
 Python                  1           43           33            1            9
 Rakefile                1           16           10            1            5
 Ruby                   39        13878        10821         1153         1904
-------------------------------------------------------------------------------
 Markdown                5          134            0           77           57
 |- Ruby                 1            1            1            0            0
 (Total)                             135            1           77           57
===============================================================================
 Total                  47        14079        10868         1234         1977
===============================================================================
```

# 参考資料

- https://quicwg.org
- https://www.rfc-editor.org/rfc/rfc9000.html
- QUIC、RFC 9000 として正式に公開 - Fastly blog
- QUICをゆっくり解説 – 新しいインターネット通信規格 | IIJ Engineers Blog
- HTTP/3の解説を書いた（2020/06/01） - ASnoKaze blog
- 2021年 HTTPやQUICの最新動向振り返り - ASnoKaze blog
- 『プロフェッショナルSSL/TLS』 – 技術書出版と販売のラムダノート
- 徹底解剖 TLS 1.3（古城 隆 松尾 卓幸 宮崎 秀樹 須賀 葉子）| 翔泳社の本
- https://quic.xargs.org

# 参考資料2

- https://github.com/aiortc/aioquic
- https://github.com/thekuwayama/tttls1.3
- https://datatracker.ietf.org/wg/quic/about/