RubyにおけるQUICの現在地

unasuke

Fastly Tech Meetup vol.1

2026-09-10

自己紹介

- Name: うなすけ
- Work: フリーランス
- Kaigi on Rails 2026 organizer (10/16-17)
- GitHub https://github.com/unasuke
- Fediverse https://mstdn.unasuke.com/@unasuke
- X https://x.com/yu_suke1994
- Website https://unasuke.com

【宣伝】Kaigi on Rails 2026

このトークの想定聴衆

- QUIC：おおまかに理解している
  - QUICの実装の名前をいくつか知っている
- Ruby：書いたことはある
  - 型のないスクリプト言語で、Webアプリの実装で使われがちなのは知ってる
- Ruby on Rails：聞いたことはある
  - RubyのWebアプリケーションフレームワークなんでしょ

そもそも、プログラミング言語のHTTP実装ってどんなもんよ

- Ruby: `net/http` (HTTP/1.1まで)
- Python: `http.client` (HTTP/1.1まで)
- Perl: `HTTP::Tiny` (HTTP/1.1まで)
- Go: `net/http` (HTTP/2まで)
  - QUICについては `golang.org/x/net/quic` があるにはある
- C#/.NET: .Net 7以降でHTTP/3をサポート
  - `HttpVersion.Version30`
- Rust: 標準ライブラリとしてHTTPを提供していない

※あくまでも言語標準としてどうなのか、の話

Rubyの net/http の現状

- サポートされているのは HTTP/1.1 まで
- 初コミットは1999年12月17日

```
require 'net/http'
require 'uri'

url = URI.parse('http://example.com/index.html')
res = Net::HTTP.start(url.host, url.port) { |http|
  http.get('/index.html')
}
puts res.body
```

Rubyの net/http の現状

- サポートされているのは HTTP/1.1 まで
- 初コミットは1999年12月17日
- HTTPSを使いたかったらこう

```ruby
require 'net/http'
require 'uri'

url = URI.parse('https://example.com/index.html')
res = Net::HTTP.start(url.host, url.port, use_ssl: true) { |http|
  http.get('/index.html')
}
puts res.body
```

そんなRubyにおけるQUICの現在地について

- OpenSSL
- protocol-quic
  - protocol-http3
- quic-ruby
- quicsilver
- raiha

OpenSSL

https://github.com/ruby/openssl/

Rubyの標準添付ライブラリのひとつ。OpenSSLのAPIをRubyから使えるようにしている。

OpenSSL側に存在するQUIC APIのRuby側への露出については https://github.com/ruby/openssl/pull/1008 にて作業が進められている。

protocol-quic, protocol-http3

これらの説明をする前に、Rubyの並行並列処理について軽く説明する必要があり……

Rubyの並行並列処理

- Threadを使う
  - いわゆるスレッド、native thread
- Fiberを使う
  - 軽量スレッド、coroutineと呼ばれたりする
  - Fiber schedulerを使う
    - IO待ちにhookしてFiberを切り替えるschedulerを導入する
    - scheduler自体の実装はRuby本体にはない
- Ractorを使う
  - M:N thread、制約は多め

Rubyの並行並列処理とRails

Railsを運用するときに使うサーバーとして挙げられるのは……

- unicorn
  - マルチプロセス（Prefork）型。pumaの前のデファクトスタンダード
  - 今だとforkされたpitchforkを使うという選択もあるかも
  - GitHubが落ちると出るrage unicornの由来
- puma
  - マルチスレッド、マルチプロセス型。現在のRailsのデフォルト
- falcon
  - Fiber schedulerを使う
  - Shopifyの本番で動いている（はず）

Rubyの並行並列処理、Fiber scheduler

Fiber schedulerのスケジューラー実装のひとつに async gem がある。

https://github.com/socketry/async

このasync gemから使いやすいようにsocketry org以下で様々なprotocolの実装が管理されている。

- https://github.com/socketry/async-http
  - https://github.com/socketry/protocol-http
  - https://github.com/socketry/protocol-http2
- https://github.com/socketry/async-websocket
  - https://github.com/socketry/protocol-websocket

protocol-quic, protocol-http3に戻ってきたぞ

- https://github.com/socketry/protocol-quic
- https://github.com/socketry/protocol-http3

ngtcp2、nghttp3とschedulerのためのC++によるコードから成るgem

たぶんまだあんまり広く使われていない（production環境での運用についての話を聞いたことがない）

quic-ruby

- https://github.com/unasuke/quic-ruby

拙作。ngtcp2とLibreSSLをbindingしたRuby gem。まだ開発途上なので……

quicsilver

- https://github.com/hahmed/quicsilver

MsQuicのbinding gem。先ほどのfalconと組み合わせて使うことも想定されているような……

raiha

- https://github.com/unasuke/raiha

拙作。これは既存のライブラリのbinding gemではなくPure RubyによるQUICプロトコルの実装を目指して開発中のものです。