

QUIC with GStreamer & Rust

Sanchayan Maity

Who

- ▶ Who am I?

Who

- ▶ Who am I?
 - ▶ Embedded Systems background

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore
- ▶ Work?

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore
- ▶ Work?
 - ▶ Software Engineer @ [asymptotic](#)

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore
- ▶ Work?
 - ▶ Software Engineer @ [asymptotic](#)
 - ▶ Open source consulting firm based out of Bangalore and Toronto

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore
- ▶ Work?
 - ▶ Software Engineer @ **asymptotic**
 - ▶ Open source consulting firm based out of Bangalore and Toronto
 - ▶ Work on low level systems software centred around multimedia

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore
- ▶ Work?
 - ▶ Software Engineer @ **asymptotic**
 - ▶ Open source consulting firm based out of Bangalore and Toronto
 - ▶ Work on low level systems software centred around multimedia
 - ▶ GStreamer, PipeWire, PulseAudio

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore
- ▶ Work?
 - ▶ Software Engineer @ **asymptotic**
 - ▶ Open source consulting firm based out of Bangalore and Toronto
 - ▶ Work on low level systems software centred around multimedia
 - ▶ GStreamer, PipeWire, PulseAudio
 - ▶ Language Polyglots

Open source contributions

- ▶ GStreamer

Open source contributions

- ▶ GStreamer
- ▶ gst-plugins-rs

Open source contributions

- ▶ GStreamer
- ▶ gst-plugins-rs
- ▶ PipeWire

Open source contributions

- ▶ GStreamer
- ▶ gst-plugins-rs
- ▶ PipeWire
- ▶ PulseAudio

Open source contributions

- ▶ GStreamer
- ▶ gst-plugins-rs
- ▶ PipeWire
- ▶ PulseAudio
- ▶ Linux

Open source contributions

- ▶ GStreamer
- ▶ gst-plugins-rs
- ▶ PipeWire
- ▶ PulseAudio
- ▶ Linux
- ▶ u-boot

Agenda

- ▶ QUIC which is a UDP-Based Multiplexed and Secure Transport and standardized in RFC 9000

Agenda

- ▶ QUIC which is a UDP-Based Multiplexed and Secure Transport and standardized in RFC 9000
- ▶ Whirlwind tour of GStreamer

Agenda

- ▶ QUIC which is a UDP-Based Multiplexed and Secure Transport and standardized in RFC 9000
- ▶ Whirlwind tour of GStreamer
- ▶ QUIC implementations in Rust

Agenda

- ▶ QUIC which is a UDP-Based Multiplexed and Secure Transport and standardized in RFC 9000
- ▶ Whirlwind tour of GStreamer
- ▶ QUIC implementations in Rust
- ▶ QUIC support in GStreamer

Agenda

- ▶ QUIC which is a UDP-Based Multiplexed and Secure Transport and standardized in RFC 9000
- ▶ Whirlwind tour of GStreamer
- ▶ QUIC implementations in Rust
- ▶ QUIC support in GStreamer
- ▶ Demo

Agenda

- ▶ QUIC which is a UDP-Based Multiplexed and Secure Transport and standardized in RFC 9000
- ▶ Whirlwind tour of GStreamer
- ▶ QUIC implementations in Rust
- ▶ QUIC support in GStreamer
- ▶ Demo
- ▶ Future work

QUIC

- ▶ QUIC is pronounced exactly like the English word “quick”

QUIC

- ▶ QUIC is pronounced exactly like the English word “quick”
- ▶ Not an acronym

QUIC

- ▶ QUIC is pronounced exactly like the English word “quick”
- ▶ Not an acronym
- ▶ Reliable and secure transport protocol

QUIC

- ▶ QUIC is pronounced exactly like the English word “quick”
- ▶ Not an acronym
- ▶ Reliable and secure transport protocol
- ▶ Addresses some of the known shortcomings of doing HTTP/2 over TCP and TLS

QUIC

- ▶ QUIC is pronounced exactly like the English word “quick”
- ▶ Not an acronym
- ▶ Reliable and secure transport protocol
- ▶ Addresses some of the known shortcomings of doing HTTP/2 over TCP and TLS
- ▶ Standardized QUIC in RFC 9000

QUIC

- ▶ QUIC is pronounced exactly like the English word “quick”
- ▶ Not an acronym
- ▶ Reliable and secure transport protocol
- ▶ Addresses some of the known shortcomings of doing HTTP/2 over TCP and TLS
- ▶ Standardized QUIC in RFC 9000
- ▶ Supported by RFC 8999, RFC 9001 and RFC 9002

Building on shoulders of giants

- ▶ HTTP/2 **RFC7540** published in May 2015

Building on shoulders of giants

- ▶ HTTP/2 **RFC7540** published in May 2015
- ▶ Makes use of multiplexing

Building on shoulders of giants

- ▶ HTTP/2 **RFC7540** published in May 2015
- ▶ Makes use of multiplexing
 - ▶ Multiple logical streams over same logical connection

Building on shoulders of giants

- ▶ HTTP/2 **RFC7540** published in May 2015
- ▶ Makes use of multiplexing
 - ▶ Multiple logical streams over same logical connection
 - ▶ Better congestion control

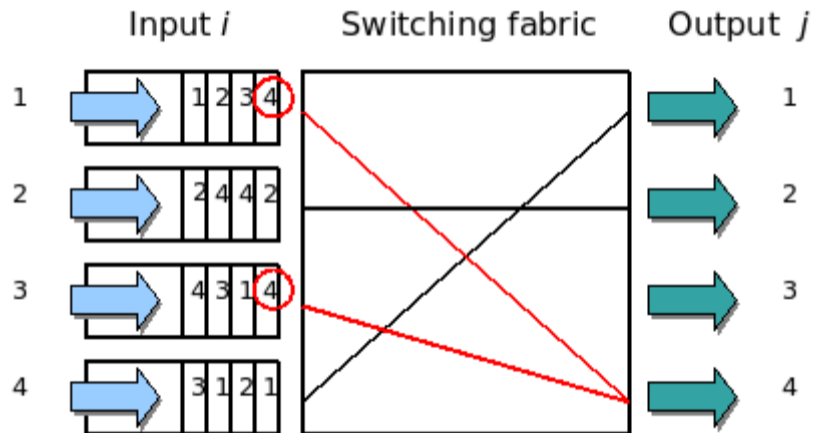
Building on shoulders of giants

- ▶ HTTP/2 [RFC7540](#) published in May 2015
- ▶ Makes use of multiplexing
 - ▶ Multiple logical streams over same logical connection
 - ▶ Better congestion control
 - ▶ Makes better use of TCP with bandwidth saturation

Building on shoulders of giants

- ▶ HTTP/2 **RFC7540** published in May 2015
- ▶ Makes use of multiplexing
 - ▶ Multiple logical streams over same logical connection
 - ▶ Better congestion control
 - ▶ Makes better use of TCP with bandwidth saturation
 - ▶ Less bandwidth consumption due to header compression

Head of line blocking¹



¹Head of line blocking

Protocol

- ▶ Something new?

Protocol

- ▶ Something new?
- ▶ TCP?

Protocol

- ▶ Something new?
- ▶ TCP?
- ▶ UDP?

Security/encryption

- ▶ No clear text version of the protocol

Security/encryption

- ▶ No clear text version of the protocol
- ▶ Negotiation employs cryptography and security with TLS 1.3

QUIC

- ▶ Implemented on top of UDP

QUIC

- ▶ Implemented on top of UDP
- ▶ Uses UDP port numbers

QUIC

- ▶ Implemented on top of UDP
- ▶ Uses UDP port numbers
- ▶ Implements re-transmission, congestion control among others

QUIC

- ▶ Implemented on top of UDP
- ▶ Uses UDP port numbers
- ▶ Implements re-transmission, congestion control among others
- ▶ Logical streams similar to HTTP/2

QUIC

- ▶ Implemented on top of UDP
- ▶ Uses UDP port numbers
- ▶ Implements re-transmission, congestion control among others
- ▶ Logical streams similar to HTTP/2
 - ▶ In-order

QUIC

- ▶ Implemented on top of UDP
- ▶ Uses UDP port numbers
- ▶ Implements re-transmission, congestion control among others
- ▶ Logical streams similar to HTTP/2
 - ▶ In-order
 - ▶ Reliable

QUIC

- ▶ Implemented on top of UDP
- ▶ Uses UDP port numbers
- ▶ Implements re-transmission, congestion control among others
- ▶ Logical streams similar to HTTP/2
 - ▶ In-order
 - ▶ Reliable
 - ▶ Different streams can be out-of-order

QUIC

- ▶ Implemented on top of UDP
- ▶ Uses UDP port numbers
- ▶ Implements re-transmission, congestion control among others
- ▶ Logical streams similar to HTTP/2
 - ▶ In-order
 - ▶ Reliable
 - ▶ Different streams can be out-of-order
- ▶ Flow control

QUIC

- ▶ Implemented on top of UDP
- ▶ Uses UDP port numbers
- ▶ Implements re-transmission, congestion control among others
- ▶ Logical streams similar to HTTP/2
 - ▶ In-order
 - ▶ Reliable
 - ▶ Different streams can be out-of-order
- ▶ Flow control
- ▶ Fast handshakes (0-RTT and 1-RTT)

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications
 - ▶ PiTiVi (Video Editor)

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications
 - ▶ PiTiVi (Video Editor)
 - ▶ amaroK, Banshee, Clementine (audio players)

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications
 - ▶ PiTiVi (Video Editor)
 - ▶ amaroK, Banshee, Clementine (audio players)
 - ▶ Empathy (VOIP and video conferencing)

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications
 - ▶ PiTiVi (Video Editor)
 - ▶ amaroK, Banshee, Clementine (audio players)
 - ▶ Empathy (VOIP and video conferencing)
 - ▶ GstLAL (gravitational wave data analysis)

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications
 - ▶ PiTiVi (Video Editor)
 - ▶ amaroK, Banshee, Clementine (audio players)
 - ▶ Empathy (VOIP and video conferencing)
 - ▶ GstLAL (gravitational wave data analysis)
 - ▶ Rygel (DLNA streaming server and renderer)

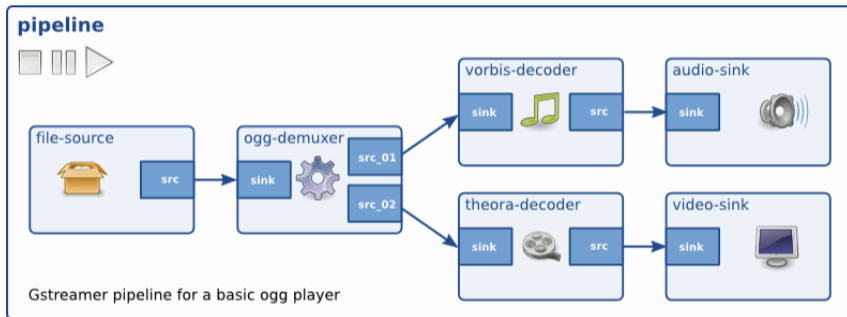
GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications
 - ▶ PiTiVi (Video Editor)
 - ▶ amaroK, Banshee, Clementine (audio players)
 - ▶ Empathy (VOIP and video conferencing)
 - ▶ GstLAL (gravitational wave data analysis)
 - ▶ Rygel (DLNA streaming server and renderer)
 - ▶ Totem (movie player for the GNOME desktop)

Simple pipeline

```
gst-launch-1.0 videotestsrc ! autovideosink  
gst-launch-1.0 audiotestsrc ! autoaudiosink
```

Media pipeline²



Rust implementations

- ▶ quinn-rs

Rust implementations

- ▶ quinn-rs
- ▶ quiche

Rust implementations

- ▶ quinn-rs
- ▶ quiche
- ▶ s2n-quic

Rust implementations

- ▶ quinn-rs
- ▶ quiche
- ▶ s2n-quic
- ▶ neqo

Rust implementations

- ▶ quinn-rs
- ▶ quiche
- ▶ s2n-quic
- ▶ neqo
- ▶ msquic

QUIC in GStreamer

- ▶ Prior work

QUIC in GStreamer

- ▶ Prior work
 - ▶ `gst-quick-transport`

QUIC in GStreamer

- ▶ Prior work
 - ▶ `gst-quit-transport`
- ▶ `quinnquicsink` and `quinnquicsrc` (Merged just a month ago)

QUIC in GStreamer

- ▶ Prior work
 - ▶ `gst-quic-transport`
- ▶ `quinnquicsink` and `quinnquicsrc` (Merged just a month ago)
- ▶ Written in Rust

QUIC in GStreamer

- ▶ Prior work
 - ▶ `gst-quic-transport`
- ▶ `quinnquicsink` and `quinnquicsrc` (Merged just a month ago)
- ▶ Written in Rust
- ▶ Uses `quinn-rs`

QUIC in GStreamer

- ▶ Prior work
 - ▶ `gst-quic-transport`
- ▶ `quinnquicsink` and `quinnquicsrc` (Merged just a month ago)
- ▶ Written in Rust
- ▶ Uses `quinn-rs`
- ▶ New elements `quinnquicmux` and `quinnquicdemux` to support stream multiplexing

Audio demo

```
gst-launch-1.0 -v -e audiotestsrc blocksize=4096 ! \  
audio/x-raw,format=S16LE,rate=48000,channels=2,layout=interleaved ! \  
opusenc ! quinnquicsink use-datagram=false secure-connection=false
```

```
gst-launch-1.0 -v -e audiotestsrc blocksize=4096 ! \  
audio/x-raw,format=S16LE,rate=48000,channels=2,layout=interleaved ! \  
opusenc ! quinnquicsink use-datagram=false secure-connection=false
```

Video demo

quin-quick-mux

- ▶ Shows stream and datagram multiplexed on same connection

Video demo

quin-quick-mux

- ▶ Shows stream and datagram multiplexed on same connection
- ▶ Merge request: !1634

Future work

- ▶ Handling flow control

Future work

- ▶ Handling flow control
- ▶ Congestion control

Future work

- ▶ Handling flow control
- ▶ Congestion control
- ▶ RTP over QUIC

Future work

- ▶ Handling flow control
- ▶ Congestion control
- ▶ RTP over QUIC
- ▶ Media over QUIC

References

- ▶ RFC 9000

References

- ▶ RFC 9000
- ▶ Road to QUIC

References

- ▶ RFC 9000
- ▶ Road to QUIC
- ▶ 0-RTT

Questions

- ▶ Reach out on

Questions

- ▶ Reach out on
 - ▶ Email: sanchayan@sanchayanmaity.net

Questions

- ▶ Reach out on
 - ▶ Email: sanchayan@sanchayanmaity.net
 - ▶ Mastodon: sanchayanmaity.com

Questions

- ▶ Reach out on
 - ▶ Email: sanchayan@sanchayanmaity.net
 - ▶ Mastodon: sanchayanmaity.com
 - ▶ Telegram: <https://t.me/SanchayanMaity>

Questions

- ▶ Reach out on
 - ▶ Email: sanchayan@sanchayanmaity.net
 - ▶ Mastodon: [sanchayanmaity.com](https://mastodon.social/@sanchayanmaity)
 - ▶ Telegram: <https://t.me/SanchayanMaity>
 - ▶ Blog: sanchayanmaity.net