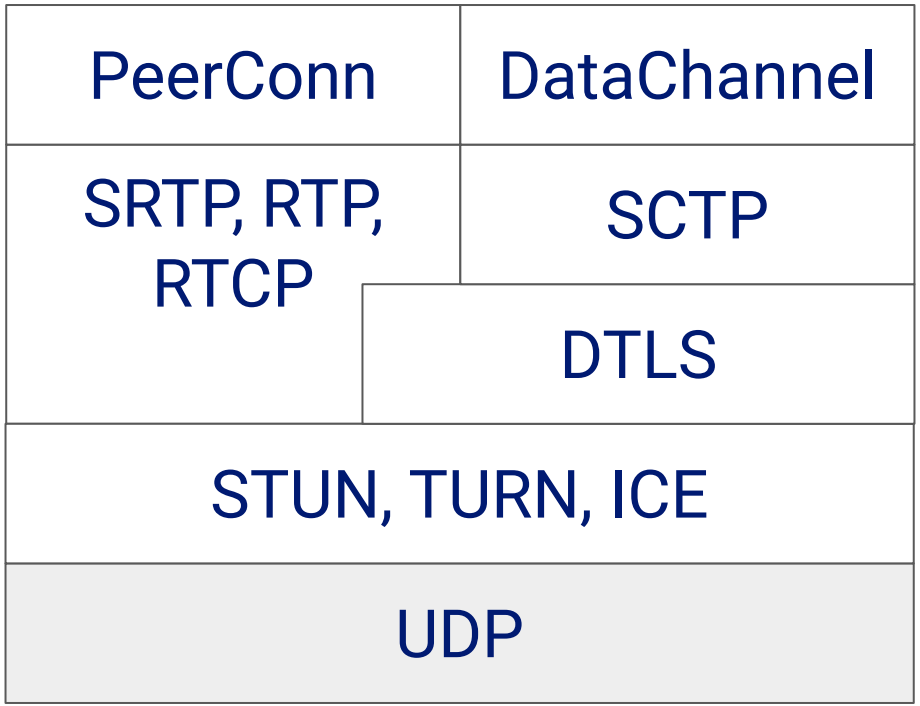
Membrane

Rewrite Pion in Elixir


PeerConn
DataChannel
SRTP, RTP, RTCP
SCTP
DTLS
STUN, TURN, ICE
UDP

Pion

- WebRTC implementation written in Go
- implements everything from scratch
- fast and popular (used by LiveKit)
- large community
- WebRTC for the curious

Elixir WebRTC

Why Elixir?

- not so popular
- pretty slow (comparing to Go, Rust or C++)
- functional and dynamically typed
- there are WebRTC implementations available on the market

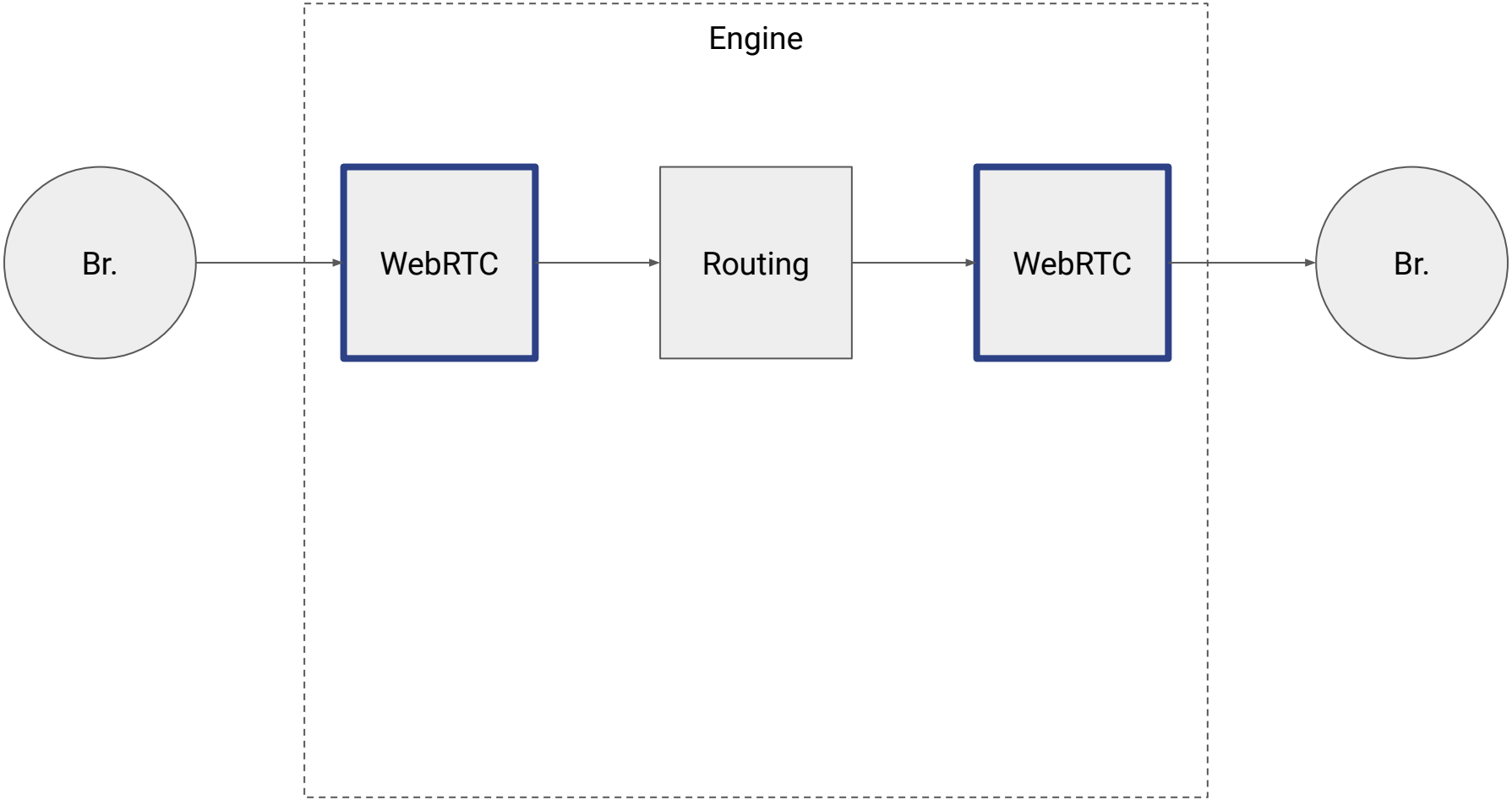
Bigger picture

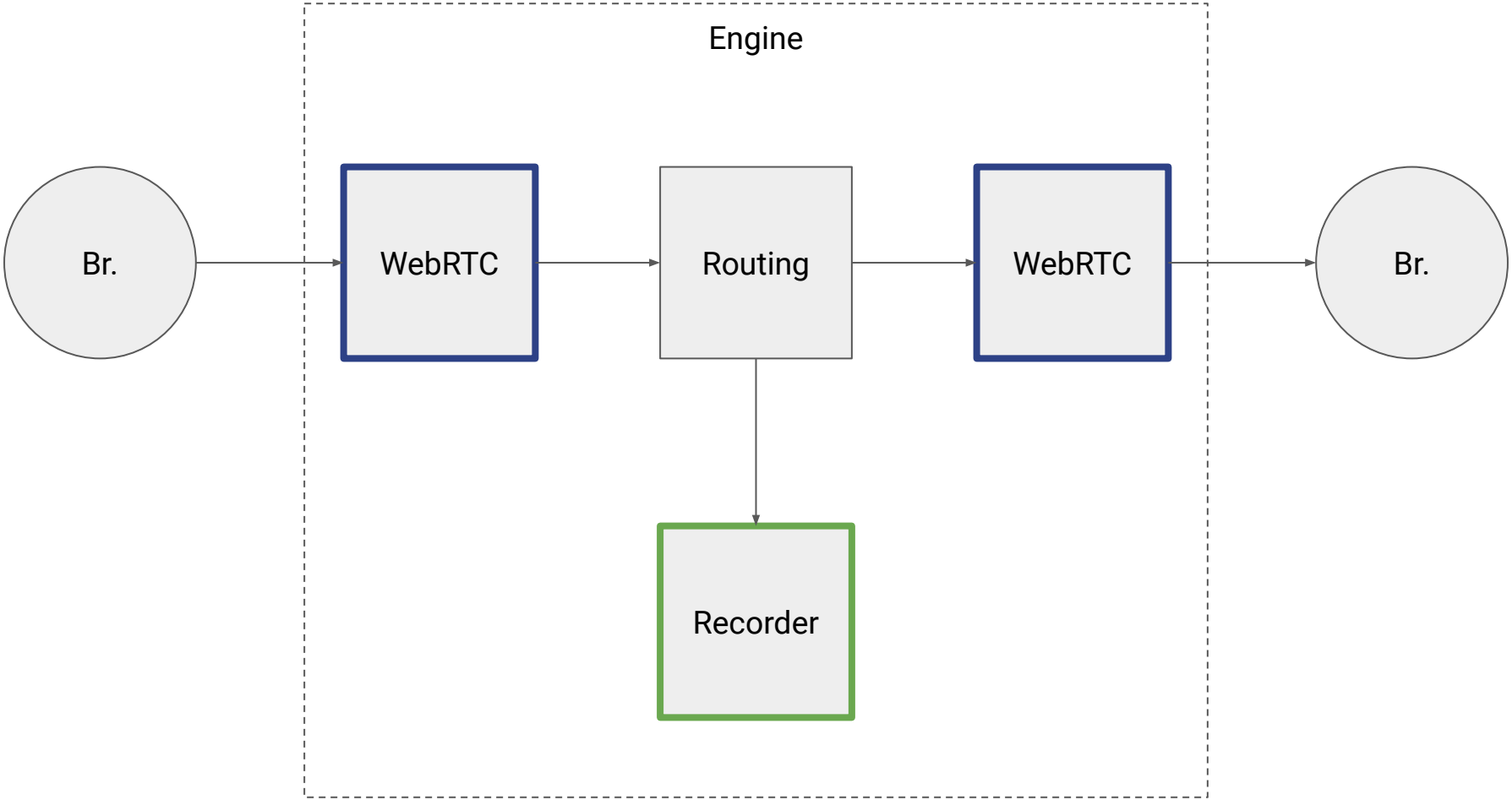
Jumping into multimedia world

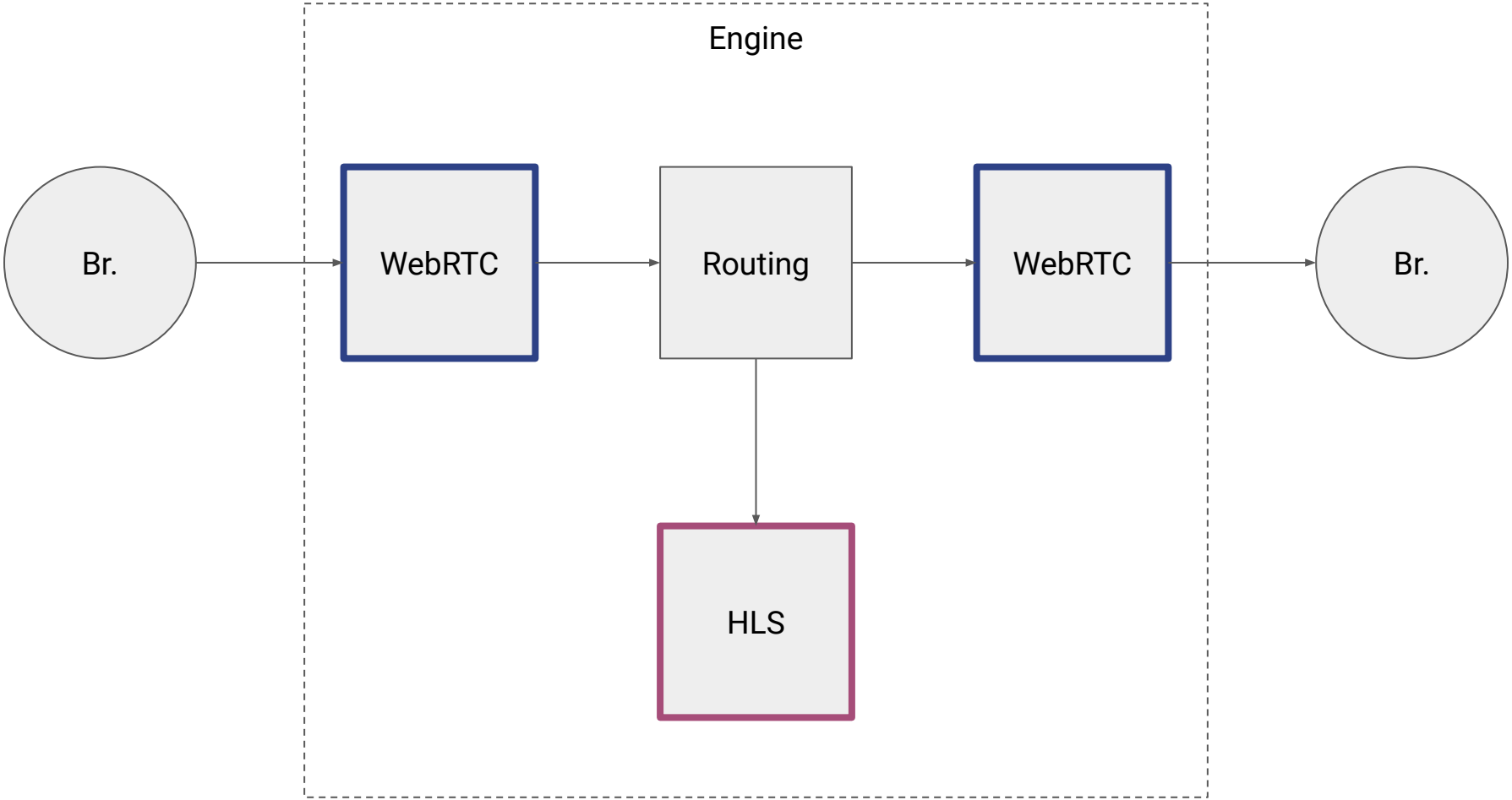
- so many weird terms, codecs, network protocols and RFCs
- few learning resources
- legacy network stuff
- different kinds of multimedia systems
- poor UX

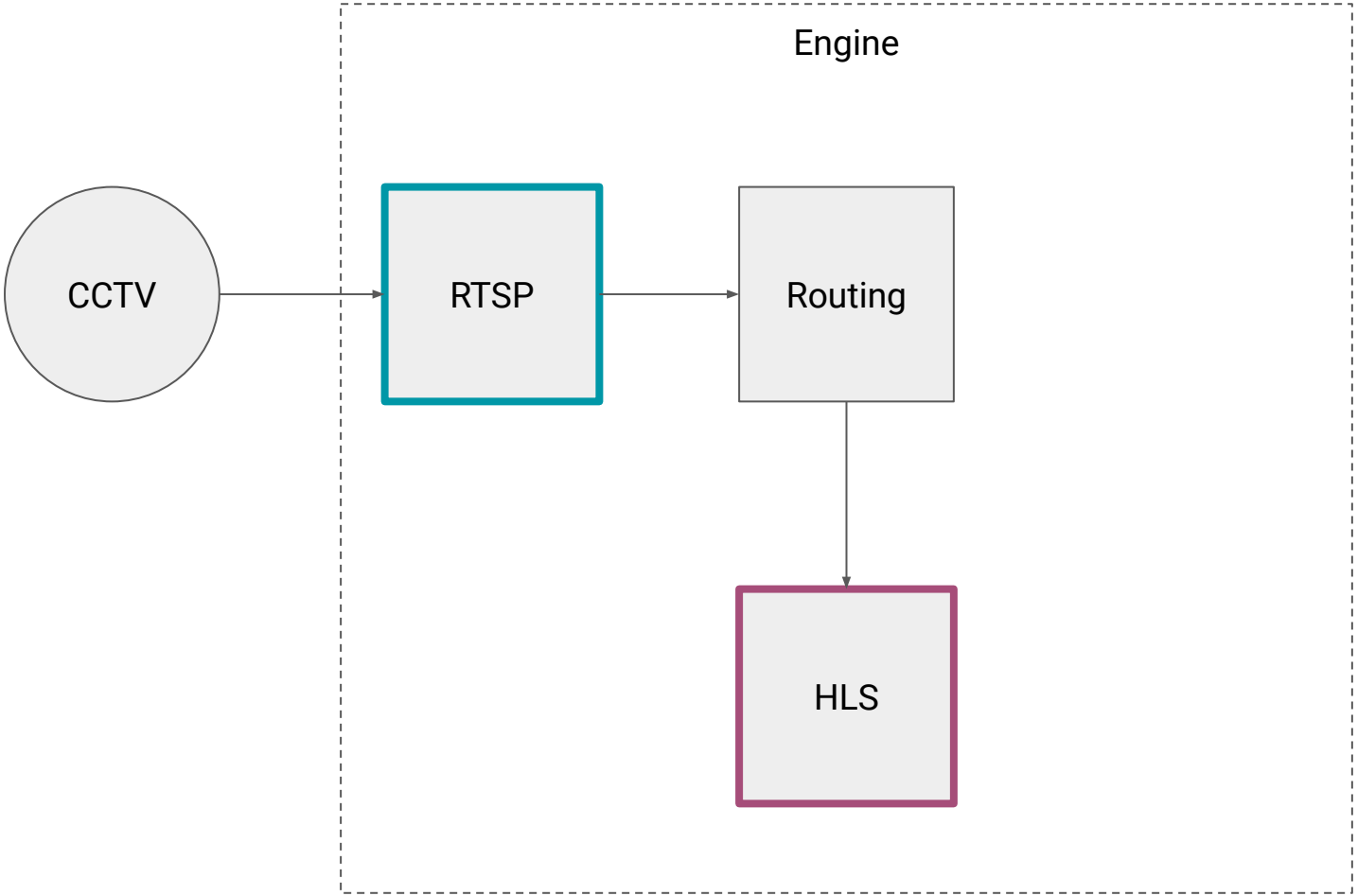
Make multimedia more accessible

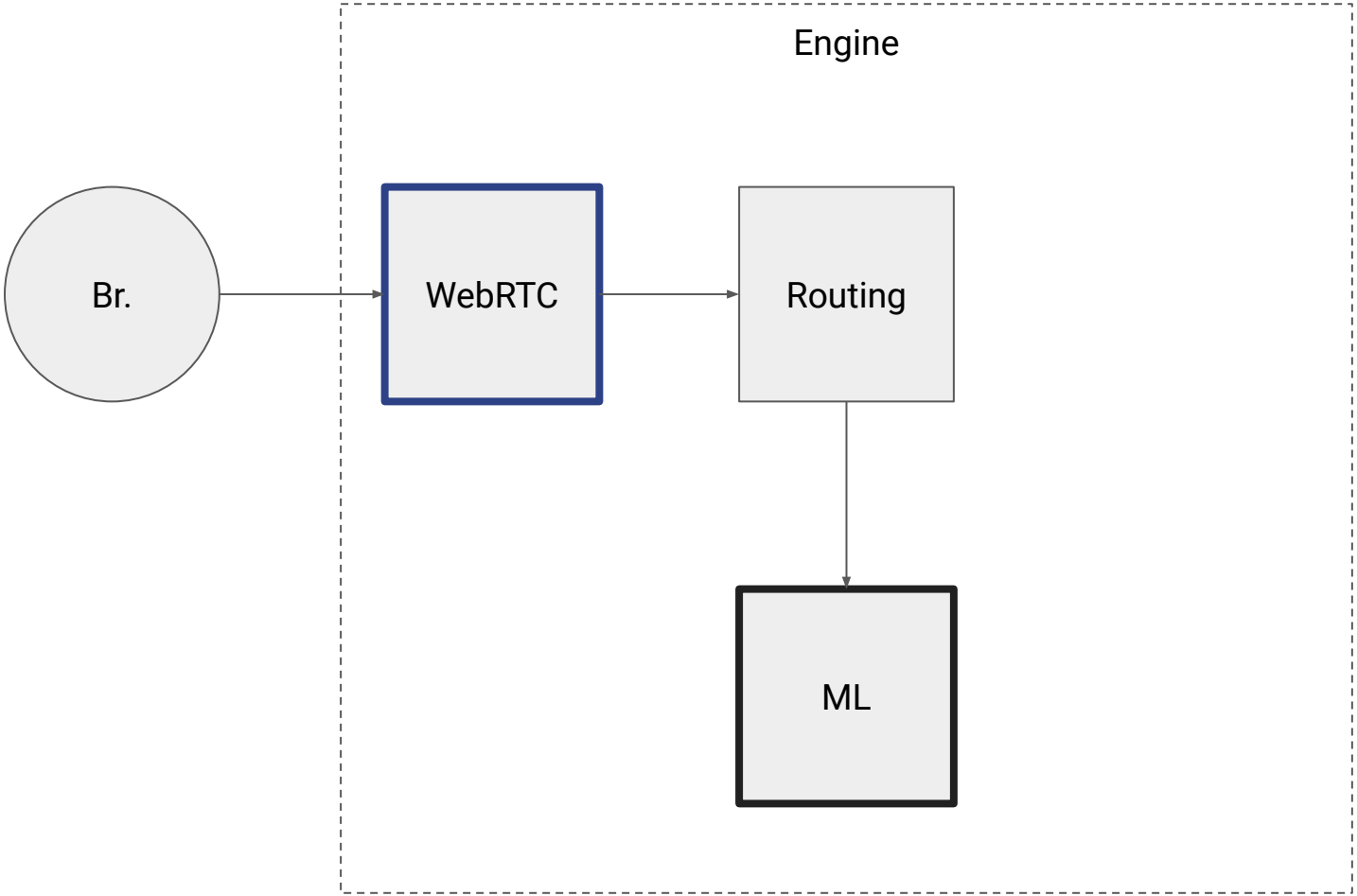
Media Engine

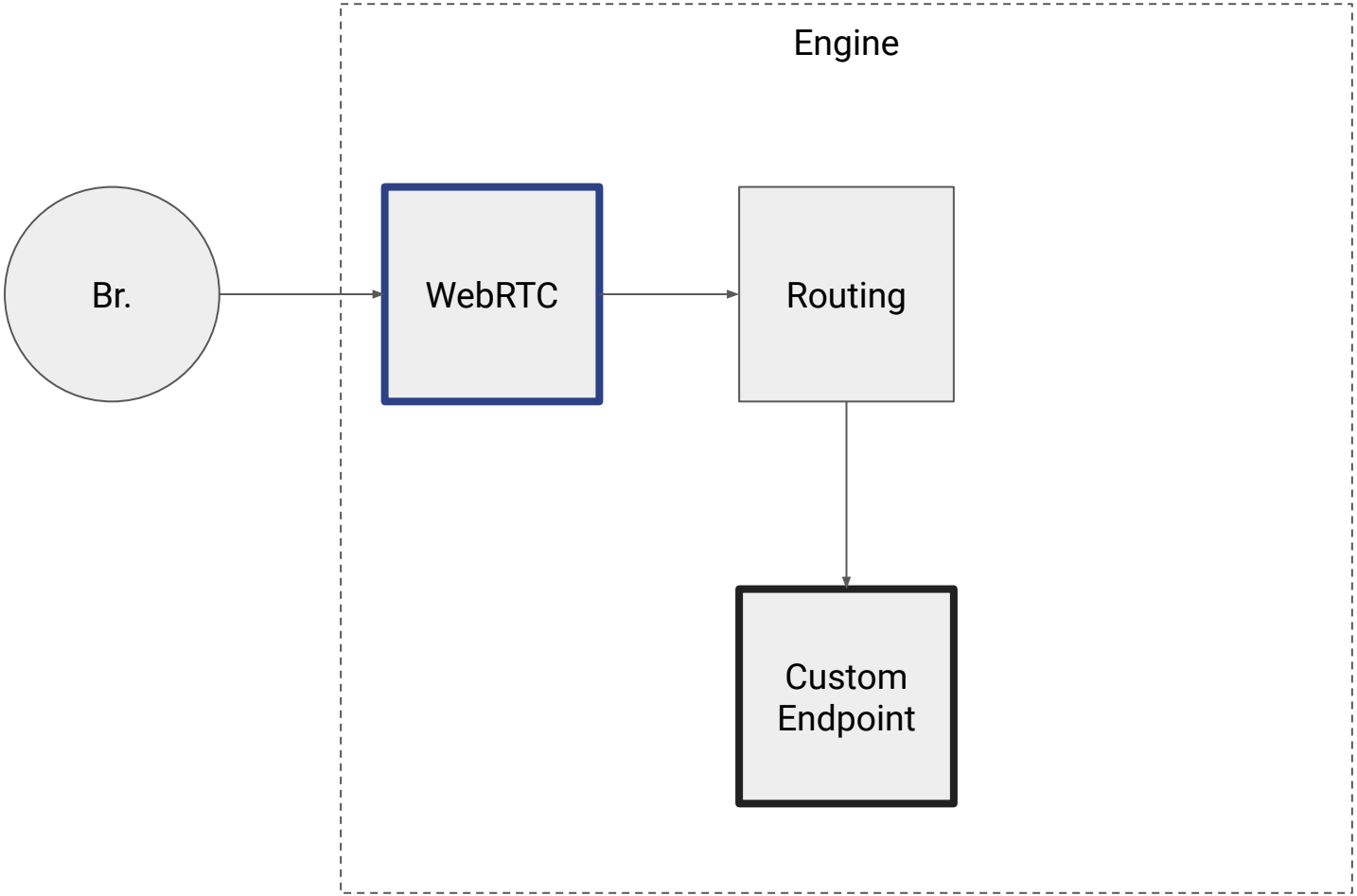

Engine
Br.
WebRTC
Routing
WebRTC
Br.


Engine
Br.
WebRTC
Routing
WebRTC
Br.
Recorder


Engine
Br.
WebRTC
Routing
WebRTC
Br.
HLS


Engine
CCTV
RTSP
Routing
HLS


Engine
Br.
WebRTC
Routing
ML


Engine
Br.
WebRTC
Routing
Custom
Endpoint

Requirements

- customizable/pluggable
- high level
- built-in fundamental protocols support
- intuitive API
- good package manager and documentation tool
- wide ecosystem

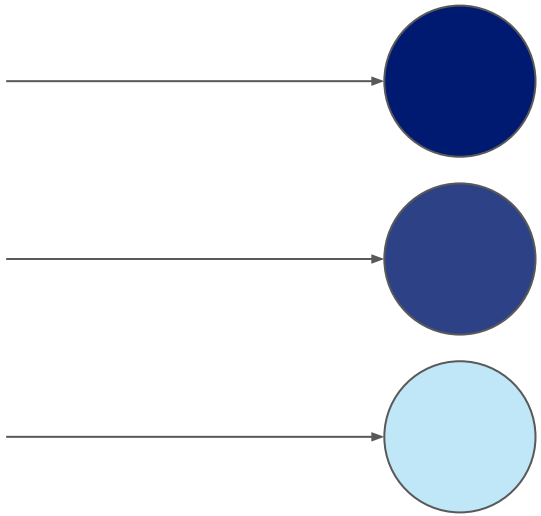
Elixir is a perfect choice!

- high level
- easy to understand concurrency model
- Phoenix
- Livebook
- Nx
- mix and ex_doc

I want SaaS!

I want to deploy!

I want to prototype!

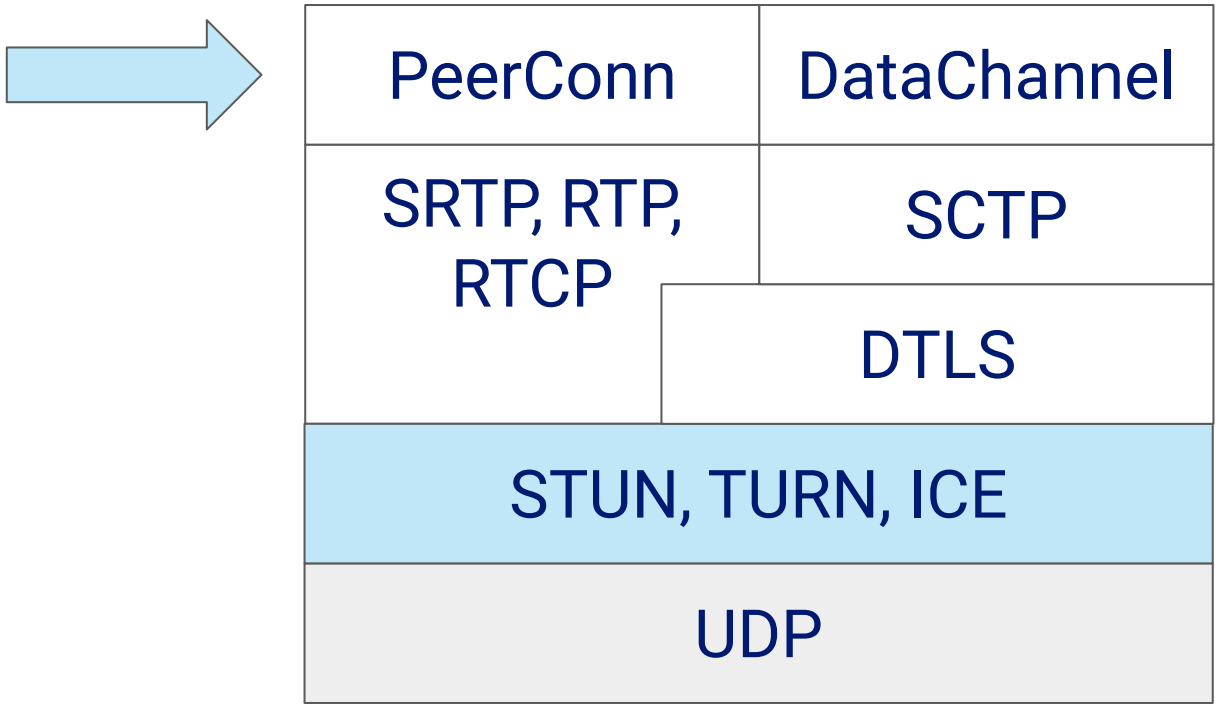


Cloud

Jellyfish

Engine

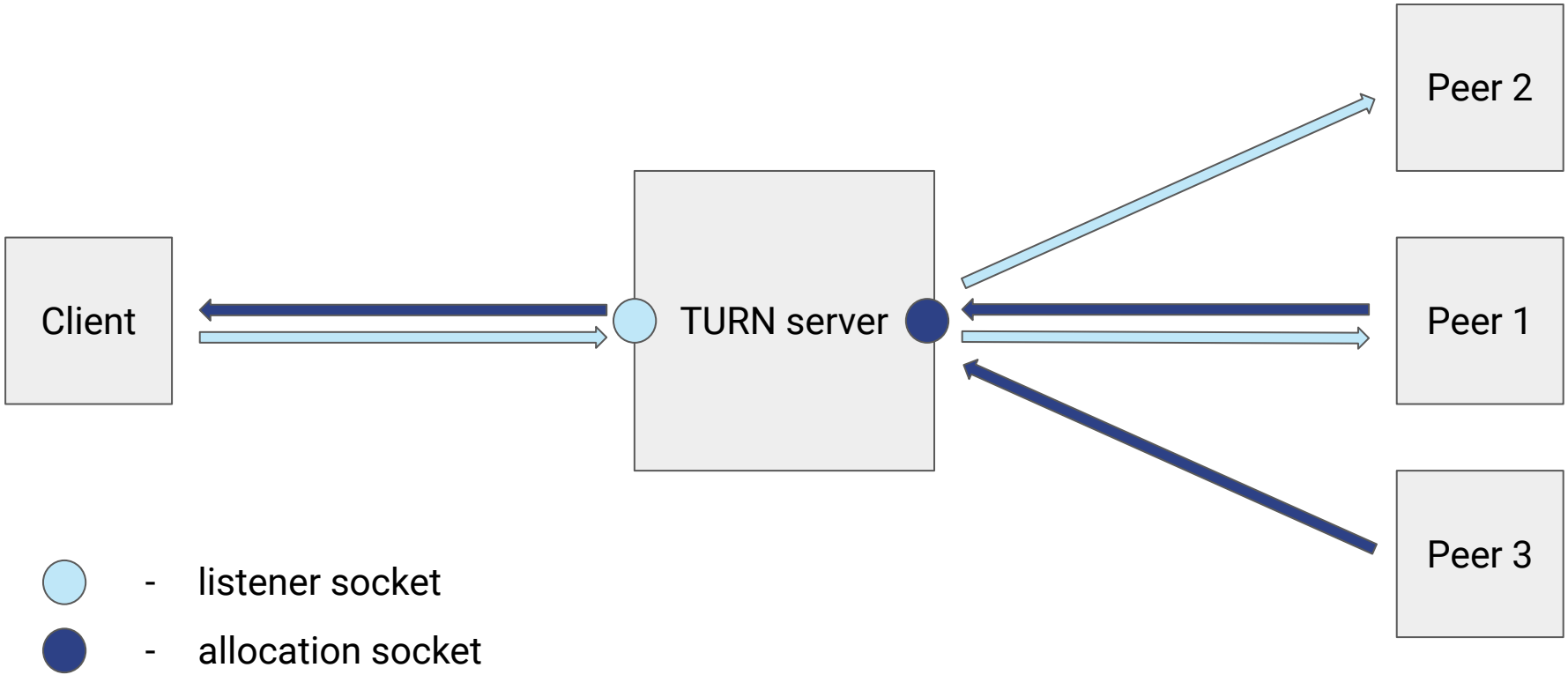
Elixir WebRTC


PeerConn
DataChannel
SRTP, RTP, RTCP
SCTP
DTLS
STUN, TURN, ICE
UDP

Libraries

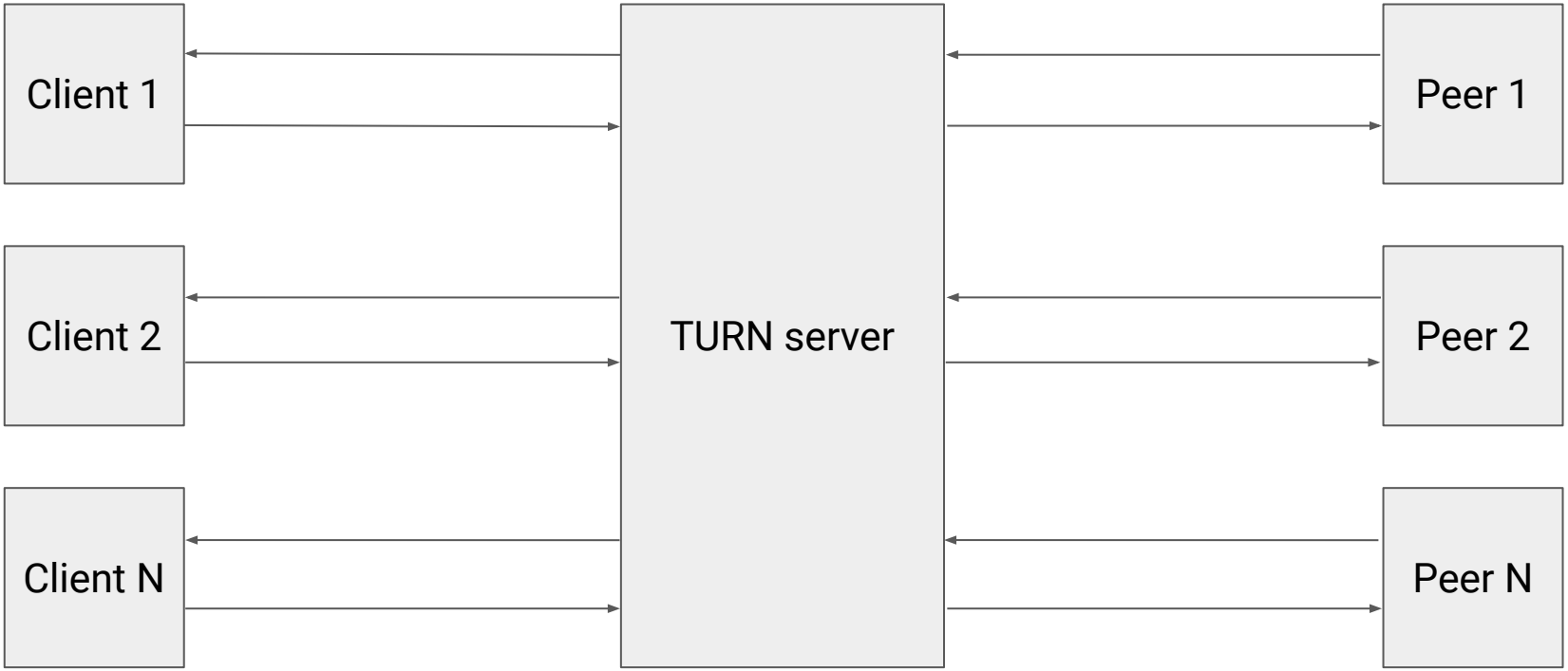
Rel

TURN server

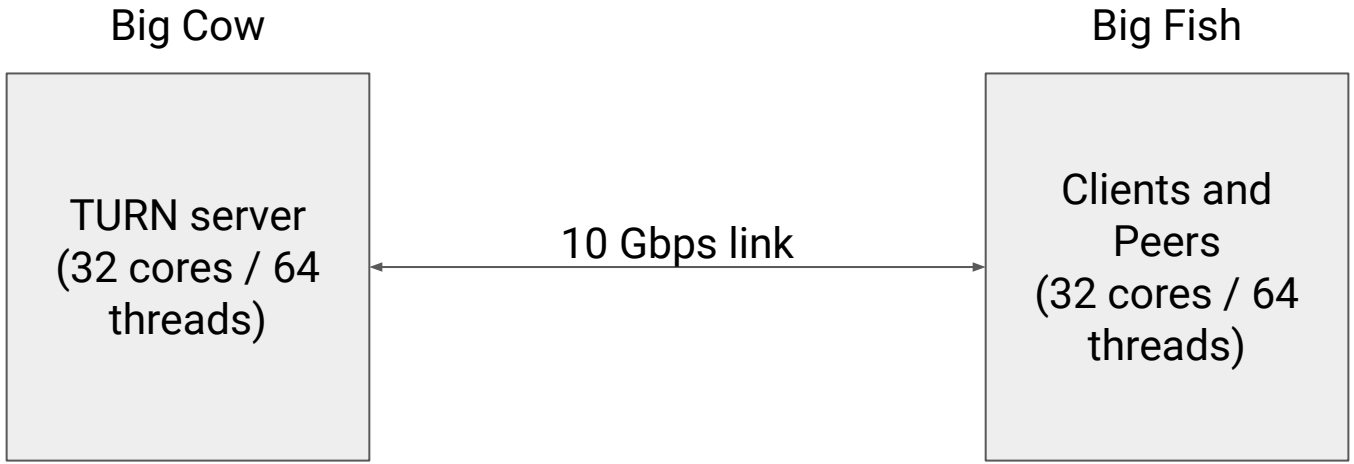

Client
TURN server
Peer 2
Peer 1
Peer 3
- listener socket
- allocation socket

Benchmarks

Scenario

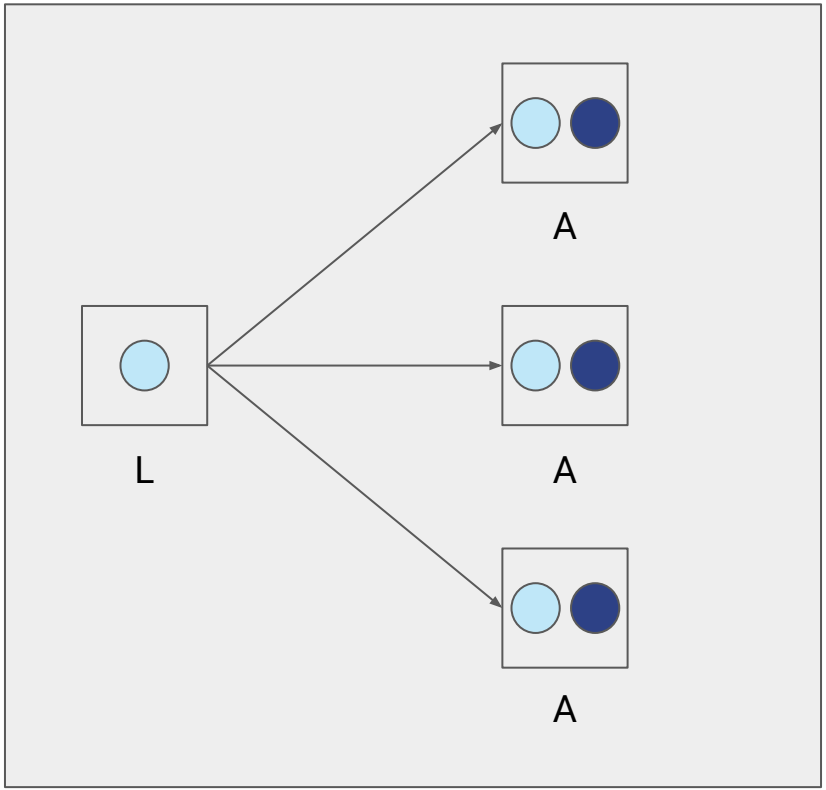


Testbed


Big Cow
Big Fish
TURN server
(32 cores / 64
threads)
10 Gbps link
Clients and
Peers
(32 cores / 64
threads)

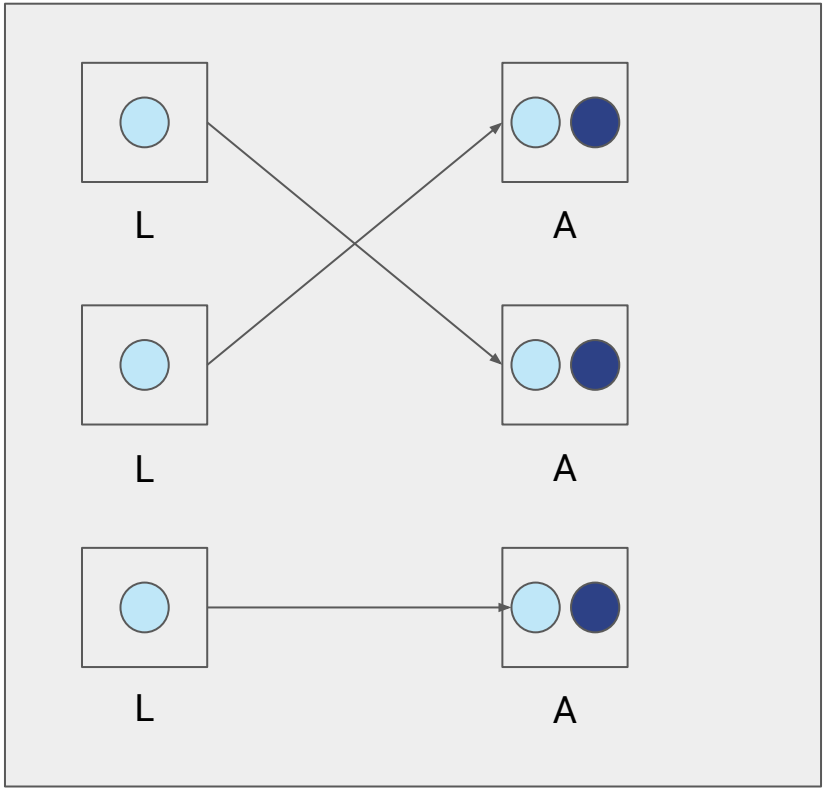
Results

Conns	Bitrate in one direction (kbps)	Payload (bytes)	Overall Bitrate (Mbps)	Rel (Elixir)	coTURN (C)	eturnal (Erlang)
2000	50	150	400	25%	10%	?
2000	50	1200	400	2-4%	1.3%	?
1000	1500	1200	5200	45%	15%	crashes



- listener socket

- allocation socket



- listener socket

- allocation socket

Rel is publicly available!

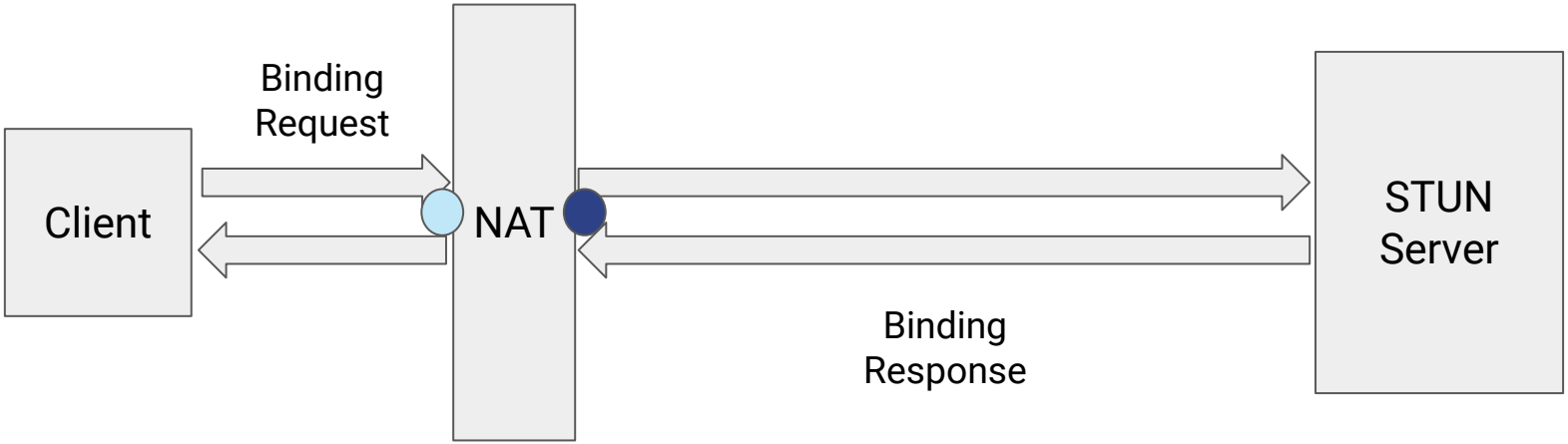
```
$ curl -X POST
"https://turn.bigcow.ovh/?service=turn&username=john"


{
  "password":"l6hs9SzUgudFeb5XjrfCfOWKeOQ=",
  "ttl":1728,
  "uris":["turn:167.235.241.140:3478?transport=udp"],
  "username":"1691574817:johnsmith"
}
```

```
pc = new RTCPeerConnection({
  iceServers: [
    {
      credential: "l6hs9SzUgudFeb5XjrfCfOWKeOQ=",
      urls: "turn:167.235.241.140:3478?transport=udp",
      username: "1691574817:john"
    }
  ]
});
```

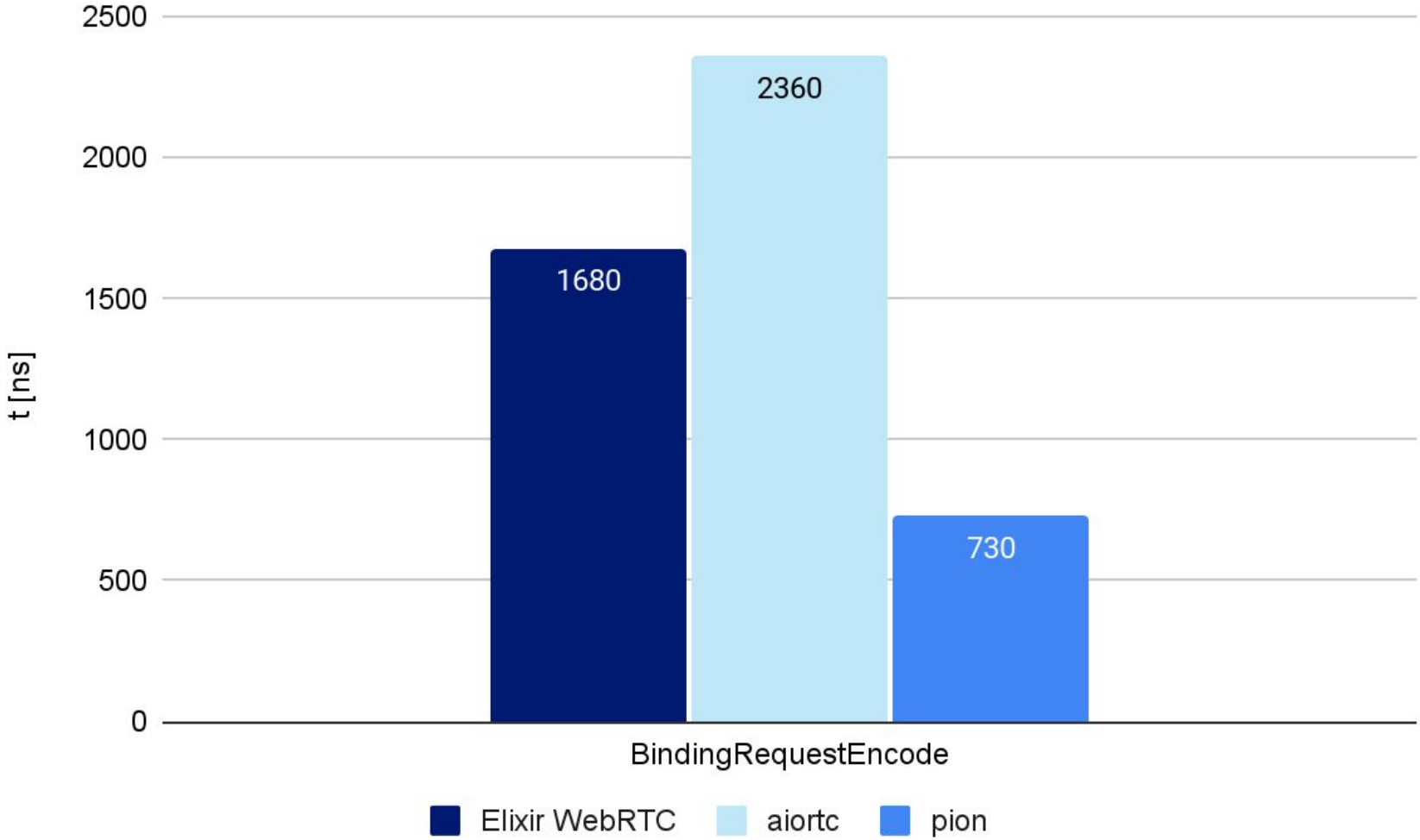
ExICE

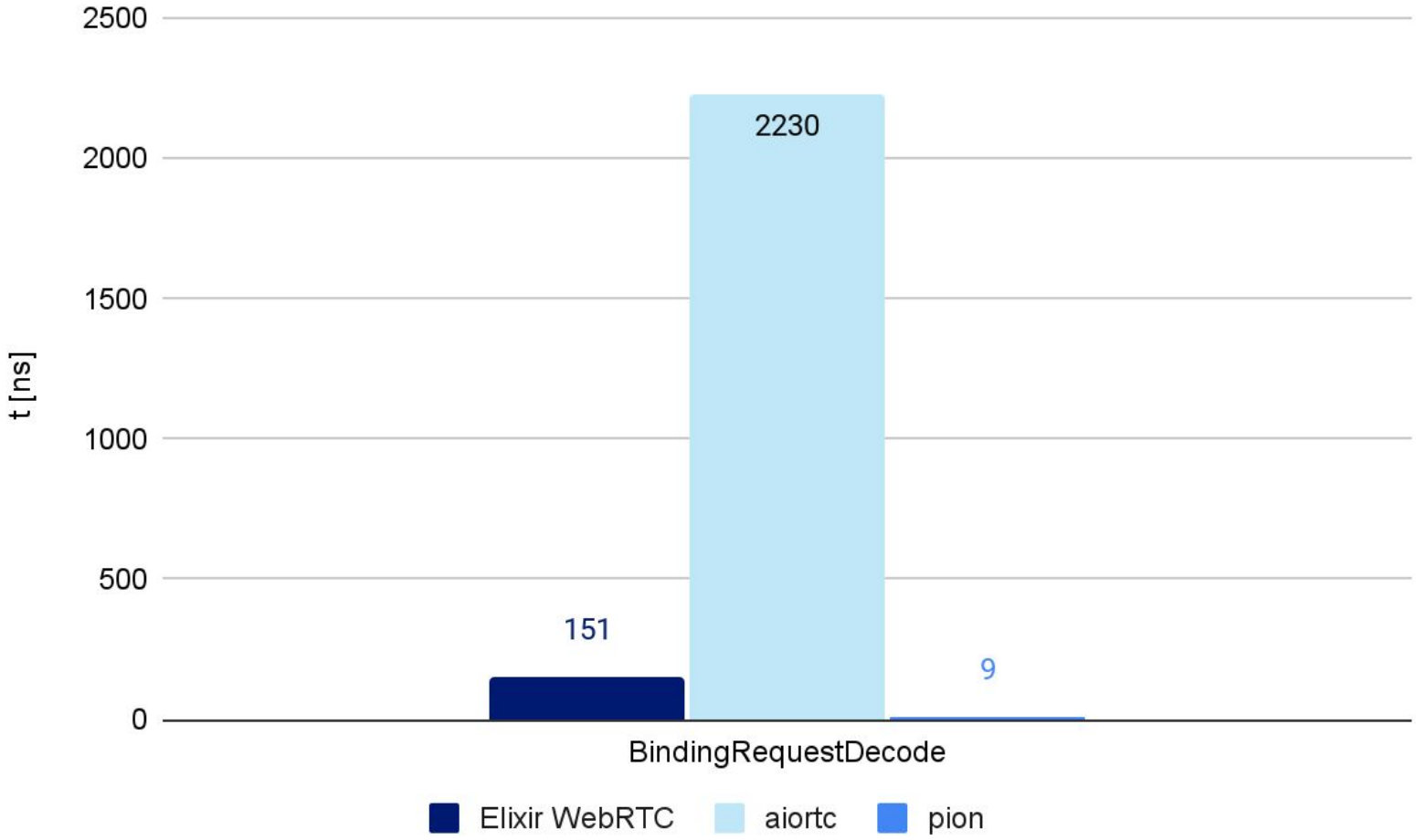
- compatible both with aggressive and regular nomination
- role conflict resolution
- supports host, prflx, srflx and remote relay candidates
- transaction pacing
- keepalives on valid and selected pairs

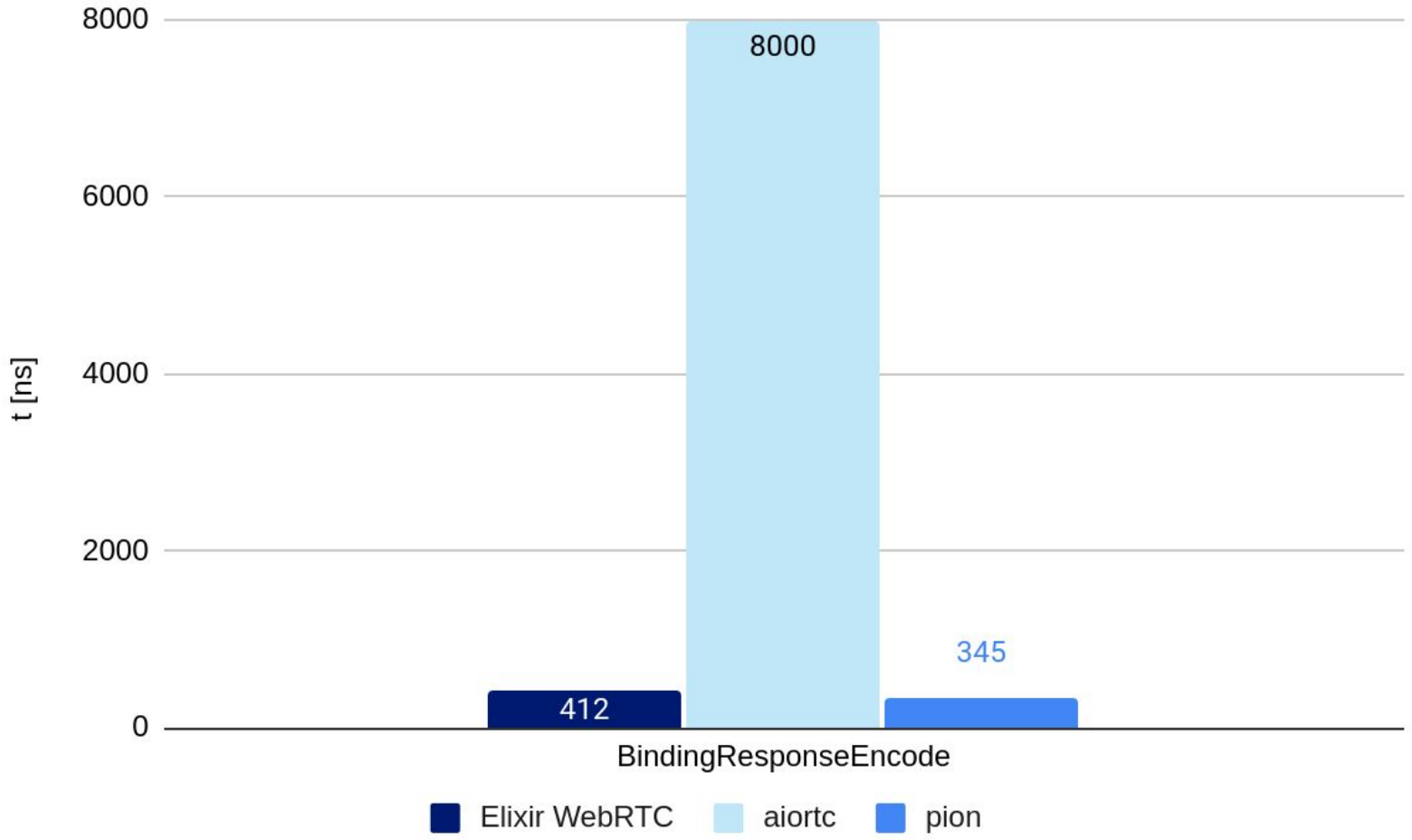
STUN

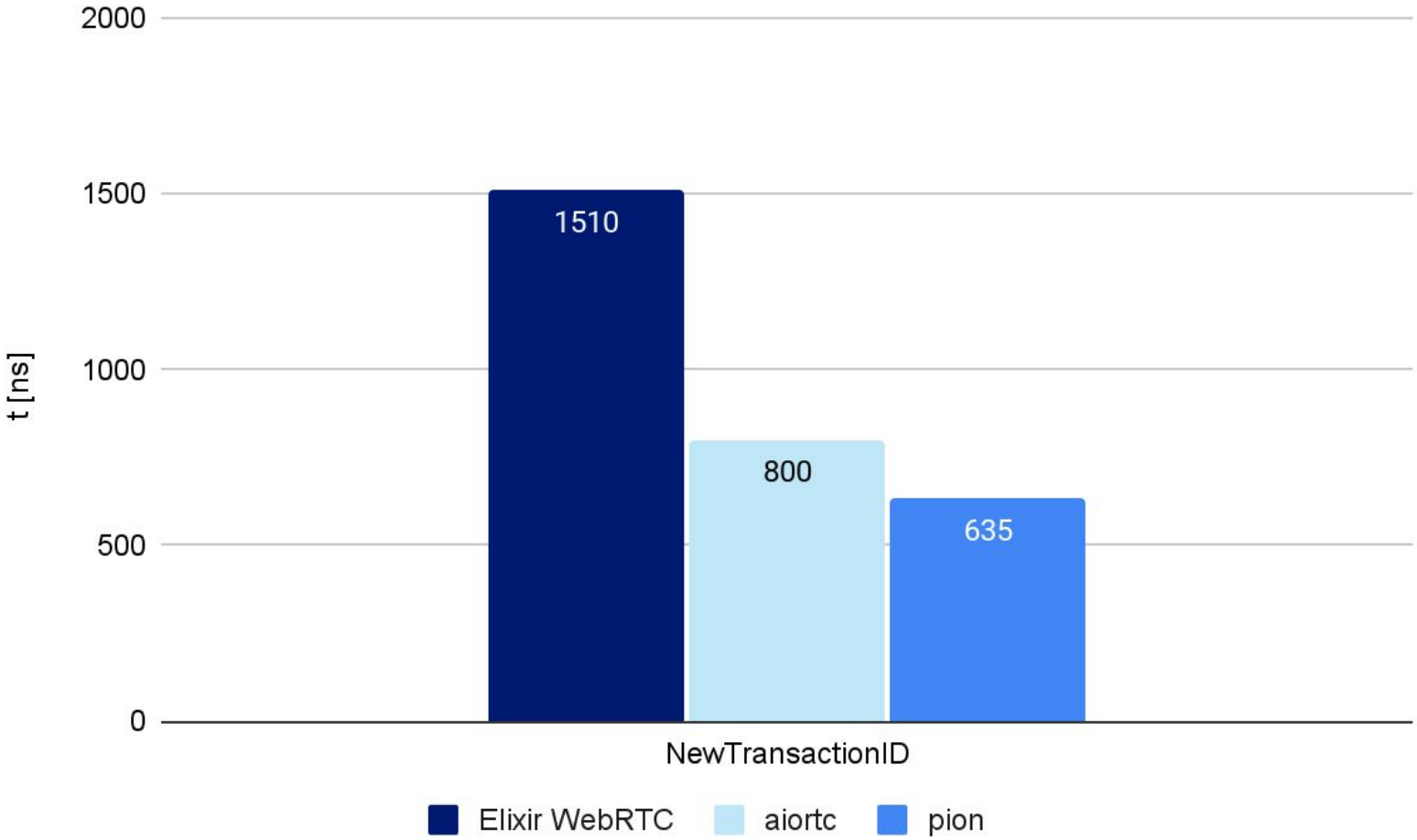


Benchmarks


2500
2000
1500
1000
500
0
t [ns]
1680
2360
730
BindingRequestEncode
Elixir WebRTC
aiortc
pion


2500
2000
1500
1000
500
0
t [ns]
2230
151
9
BindingRequestDecode
Elixir WebRTC
aiortc
pion


8000
6000
4000
2000
0
t [ns]
8000
412
345
BindingResponseEncode
Elixir WebRTC
aiortc
pion


2000
1500
1000
500
0
t [ns]
1510
800
635
NewTransactionID
Elixir WebRTC
aiortc
pion

Elixir WebRTC
I 2024

Media Engine
VI 2024

Thank you!

- https://github.com/elixir-webrtc
- https://elixir-webrtc.github.io
- https://github.com/jellyfish-dev