0. Packet arrives

Receive Buffer ← 1. RXI notification

OS

9.

GSocket

8. Call recvmsg() with 4 buffers & return them with, ideally, one protocol header in each buffer. Slow path: if the protocols are not fixed-size headers, or the layering is not as expected, memcpy()s happen.

2. GSource ready

10. Snapshot buffer S

7. Allocate GInputVector of size S, reallocate GInputMessage array to prepend buffer S; call receive_messages() with these 4 buffers

libnice

Base GSource

NiceSource — DemuxSource2 / DemuxSource1

DemuxSource!

3. GSource ready

11. Snapshot buffer M and check multiplexing ID. If it matches (a), return that; otherwise ELXAUDBLOCK and (b) the data to an internal buffer or drop it

Demux

6. Allocate GInputVector of size M, reallocate GInputMessage array to prepend buffer M; call receive_messages() with these 3 buffers

3a. GSource ready

PseudoTcpSocket
Receive buffer

PseudoTcpSource

4a. GSource ready   4b. GSource ready   4c. GSource ready

12. Snapshot buffer and read header, return #Source?

5. Allocate GInputVector of size X, call receive_messages() returns X bytes

GDtlsConn Receive buffer
GDtlsConn Receive buffer
GDtlsConn Receive buffer

TlsSource   TlsSource2   TlsSource3

Major code   User code

code that the user probably didn't want to exist

13. GSource ready

14. Call receive_messages(), pop off the internal receive buffer

15. Even MORE user code

Packet format (for example):

DTLS header	Muxer header	STUN header (libnice)	UDP header	User data
T bytes	M bytes	S bytes	U bytes	X bytes

UDP header	STUN header (libnice)	Muxer header	DTLS header	User data
U bytes	S bytes	M bytes	T bytes	X bytes
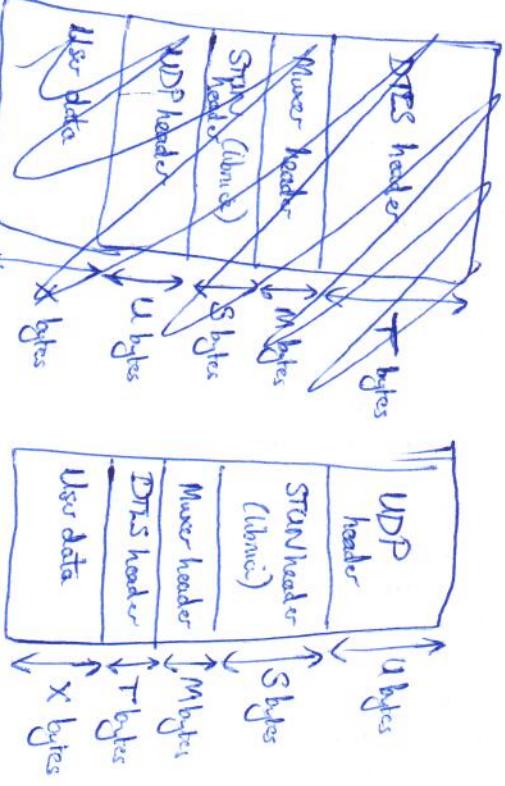
Caveats:
- Every stream element which doesn't do output GSource ready at least as often as its input — this results in over-notification for demux outputs.
- Buffer handling in the multiplexer gets hairy.

Key points:
- Every stream element which does buffering can break partition the chain of (synchronous) pulls
- Variable allocations above could be taken from pre-allocated receive buffers, or be static