TCP vs UDP в стриминге

Какой транспорт у каждого стримингового протокола и почему.

1. Сравнение по шести измерениям

Измерение	TCP — RFC 9293	UDP — RFC 768
Установка соединения	3-way handshake + TLS (1–3 RTT)	Нет — 0 RTT
Надёжность	Гарантия через ACK + retx	Best-effort — задача приложения
Упорядочивание	Строго по порядку (HoL)	Нет — любой порядок
Congestion control	Обязателен (CUBIC, BBR, Reno)	Приложение (RFC 8085)
Flow control	Receive window	Нет
Заголовок	20+ байт	8 байт
Пол задержки	1 RTT на пакет, блокирует поток	0 — следующий пакет сразу

2. Какой протокол сидит на каком транспорте

TCP — файло-подобные, с буфером

- HLS — RFC 8216 (2017)
- MPEG-DASH — ISO/IEC 23009-1:2022
- CMAF · LL-HLS · LL-DASH
- RTMP delivery (легаси)
- HTTP · HTTPS · загрузка файлов
- Сигнализация WHIP

UDP — real-time, терпят потери

- WebRTC медиа — RFC 8825
- SRT — draft-sharabayko-srt-01
- RIST — SMPTE TR-06-1/2/3
- RTP — RFC 3550
- Media over QUIC — draft-ietf-moq
- HTTP/3 поверх QUIC — RFC 9114

3. Head-of-line blocking — факт, который решает всё

TCP: F100 потерян → буфер держит F101–F104 ~160 мс → 4–6 кадров стопора

Строгий порядок — правило; HoL blocking — следствие; выключить нельзя.

OTT терпит: буфер плеера 3–24 с скрывает стопор. Конференцию убивает: бюджет < 200 мс.

UDP: F100 потерян → F101 играется сразу → декодер закрывает пропуск → нет заморозки

NACK / RTX / FEC на уровне приложения восстанавливают что успеют в бюджет; остальное скрывается.

Ближайший I-frame пересинхронизирует за ~2 с. Glass-to-glass держится 100–500 мс.

4. Правило выбора

Надёжность — это бюджет задержки, а не свойство. Берите транспорт под бюджет:

Бюджет > 3 с → TCP через HLS / DASH / LL-HLS. Буфер скрывает retx; CDN кэширует сегменты.

Бюджет < 1 с → UDP через WebRTC / SRT / RIST / RTP. Восстановление в приложении бьёт TCP-retx.

Новый проект, 2026 → Обратитесь к QUIC. Per-stream-мультиплексирование убивает HoL на уровне соединения.