

Infrastruktura za in-app chat — Music Reactions

Pripremne napomene za DevOps

1. Osnovni problem: WebSocket konekcije i deploy strategija

Glavni problem nije Kubernetes sam po sebi — K8s sasvim dobro radi sa stateful workload-ima. Pravi problem je **deploy strategija**. HTTP zahtevi su kratkotrajni, pa rolling update bez problema prekida i ponovo pokreće pod-ove. WebSocket konekcije su dugotrajne i stateful — ako pod bude ubijen tokom rolling update-a, svi korisnici konektovani na njega gube vezu. Ovo je problem već sa postojećim chat-om, a postaje ozbiljno sa stotinama korisnika u event chat-ovima.

2. Preporuka: izolovati Reverb od API pod-ova

Reverb treba pokretati izolovano od API saobraćaja — bilo na dedikovanoj EC2 instanci, bilo na **zasebnoj EKS node grupi** (sa taints/tolerations, odgovarajućim terminationGracePeriodSeconds, PodDisruptionBudget-ima i connection draining-om). Ako je DevOps ekipa već EKS-fluentna, zasebna node grupa je verovatno manje friction nego uvođenje pet EC2 instanci u operativni alat. Oba pristupa su validna — bitno je da Reverb ne deli node-ove sa API pod-ovima.

Razlozi za izolaciju:

- **Persistencija konekcija** — dedikovan setup omogućava da se deployment-i rade po maintenance schedule-u, a ne pri svakom API release-u
- **Resource isolation** — hiljade persistent WebSocket konekcija stalno troše RAM. Mešano sa API pod-ovima, jedni drugima gladuju resurse
- **OS-level tuning** — WebSocket serveru su potrebni specifični kernel parametri koje ne želiš da primeniš na API pod-ove

3. Dimenzionisanje

Za scale od „nekoliko stotina po event-u”: jedna `c6i.large` (ili čak `t3.medium` za početak) drži više hiljada idle WebSocket konekcija bez napora. `c6i.xlarge` je overkill za trenutni scale — povećavati tek ako se pojavi stvarni pritisak.

4. ALB i timeout

ALB ispred Reverb-a sa enable-ovanim WebSocket support-om. Idle timeout postaviti na **3600s**.

Napomena: Pusher protokol (koji Reverb implementira) šalje ping na ~30s, pa konekcije obično prežive default 60s timeout — ali povećanje na 3600s je svakako pravi potez radi otpornosti na network hiccup-ove.

5. Health check-ovi

TCP-level health check-ovi, ne HTTP. Reverb ne odgovara na HTTP health probe-ove na WS portu kao API pod.

6. Sticky sessions

Sticky sessions na ALB-u za inicijalni WebSocket upgrade. Pub/sub rešava cross-node fanout za broadcast poruke, ali konekcija pojedinačnog klijenta mora ostati zakačena za jednu Reverb instancu.

7. Redis

Reverb koristi Redis pub/sub za sinhronizaciju između instanci — mora biti na **dedikovanoj Redis instanci** ili dedikovanoj bazi, ne deljeno sa cache-om i queue-ovima. Pub/sub saobraćaj će izgladniti cache operacije ako su mešani.

8. Najvažnija tačka — media nikada ne prolazi kroz Reverb ili API

Ovo je ono što sprečava kolaps tokom event-a gde svi dele slike i video:

- API generiše S3 **presigned upload URL** (jedan lagani endpoint poziv)
- Flutter upload-uje fajl **direktno na S3** — naši serveri nikada ne vide bajtove
- Flutter šalje S3/CloudFront URL kao payload chat poruke preko WebSocket-a
- Reverb prenosi samo URL-ove, nikada binarne podatke

Bez ovoga: jedan event sa 200 ljudi koji dele video-e zasićuje API bandwidth i ruši sve. Sa ovim pristupom, naši serveri su potpuno indiferentni na veličinu medija.

Detalji media pipeline-a koje treba pokriti:

- Politika expiry-ja za presigned URL-ove
- Server-side validacija MIME tipa i veličine **pre** izdavanja URL-a
- Thumbnailing korak (Lambda triggered na S3 PUT) — da telefoni ne preuzimaju originalne 40MB fajlove u chat scroll-u

9. CloudFront

CloudFront ispred S3 bucket-a koji se koristi za chat media. Recipijenti učitavaju media sa edge node-ova, ne sa S3 origin-a.

10. Auth strategija

Auth za presence/private channel-e koristi postojeće Sanctum token-e. Potvrditi da je Reverb auth endpoint dostupan klijentima i da živi **na API strani**, ne na samom Reverb-u.

11. Rate limiting

Rate limit na broadcast event-ima — jedan korisnik koji spam-uje tokom event-a može pokvariti iskustvo za 200 drugih. Throttle implementirati na controller-u koji publish-uje u kanal.

12. Scaling signal

Skaliranje Reverb-a po **connection count-u**, ne po CPU. WebSocket server na 5000 konekcija može sedeti na 2% CPU — CPU-based autoscaling se nikada neće trigger-ovati, ali će RAM i file descriptor-i biti istrošeni.

13. Observability

Bez ovih metrika nema vidljivosti u jedine stvari koje su važne:

- Connection count po instanci
- Message rate
- Redis pub/sub lag
- File descriptor usage

14. Odluka pre razgovora: single-instance ili horizontalno?

Single-instance pojednostavljuje sve (nema pub/sub koordinacije, sticky sessions trivijalan) i pravilno dimenzionisana mašina drži niske hiljade istovremenih konekcija. Za naš scale („stotine po event-u”), praktičan potez je **krenuti single-instance sa jasnim path-om ka horizontal scaling-u**. Vredi naglasiti eksplicitno u handoff-u da se DevOps ne bi overengineer-ovao prvi dan.

15. Sažetak za DevOps handoff

Stavka	Akcija
Reverb	Dedikovana EC2 instanca ili izolovana EKS node grupa — ne deljeno sa API pod-ovima
Dimenzionisanje	c6i.large kao start (ne c6i.xlarge); skalirati po stvarnom pritisku
ALB timeout	Idle timeout 3600s, WebSocket support enabled
Health checks	TCP-level, ne HTTP
Sticky sessions	Enabled na ALB-u za WebSocket upgrade
Redis	Dedikovana instanca za Reverb pub/sub — odvojeno od cache-a i queue-ova
OS limits	ulimit -n 100000, net.core.somaxconn=65535, TCP keepalive tuning
Media upload	Presigned S3 URL-ovi — direktno klijent→S3, nikada kroz API ili Reverb
Media validacija	MIME tip i veličina validirani server-side pre izdavanja URL-a
Thumbnailing	Lambda triggered na S3 PUT — generiše thumbnails za chat scroll
Media delivery	CloudFront ispred S3 bucket-a za chat media
Auth	Postojeći Sanctum tokeni; auth endpoint na API strani, ne na Reverb-u
Rate limiting	Throttle na controller-u koji publish-uje u broadcast kanal
Scaling metric	Connection count po instanci, ne CPU
Observability	Connection count, message rate, pub/sub lag, FD usage
Početni setup	Single-instance sa path-om ka horizontalnom skaliranju