Opis vsebine predlaganega projekta

Sistem za decentralizirano sporočanje kot plast nad blockchainom

(Ekaterina Bochvaroska - 89252018)

13. december 2025

1 Opis teme projekta

Ta projekt predlaga in prototipno implementira decentraliziran protokol za sporočanje, ki združuje lahek blockchain zapisnik z razširljivim shranjevanjem sporočil izven verige. Namesto shranjevanja celotnih šifriranih sporočil na verigi (on-chain) blockchain hrani le kompaktne reference (zaveze/kazalce) na *razširitvene bloke* (extension blocks), ki vsebujejo dejanske šifrirane vsebine. Tako ostanejo podatki na verigi majhni in učinkoviti, hkrati pa je ohranjena **preverljivost**: odjemalci lahko preverijo, da pridobljeni podatki izven verige ustrezajo temu, kar je bilo referencirano na verigi. Ključen prispevek projekta je **dispatcher poizvedb**, ki omogoča nizko-zakasnitveno pridobivanje razširitvenih blokov. Namesto zanašanja na eno samo vozlišče ali na popolno replikacijo odjemalec razdeli zahtevan interval blokov na manjše pod-intervale in jih pridobiva **vzporedno** iz več kandidatnih vozlišč. Sistem uporablja *retention metapodatke*, ki jih posredujejo vozlišča, da podpoizvedbe usmeri na vozlišča, ki najverjetneje hranijo zahtevane podatke; sprejme prvi veljaven odgovor in prekliče odvečne zahtevke. Takšna zasnova zmanjšuje *tail latency* ter izboljša odpornost na napake ob delnih izpadih.

2 Cilji

- Načrtovati dvoslojno (L2) arhitekturo, kjer blockchain hrani samo reference, medtem ko so polna šifrirana sporočila shranjena v off-chain razširitvenih blokih.
- Implementirati **retention-aware** mehanizem **razbijanja poizvedb in razpošiljanja** z vzporednim pridobivanjem.
- Implementirati omrežne komponente: odjemalsko plast, regular/archive vozlišča, storitev registra vozlišč ter medvozliščne komunikacije.
- Celoten sistem kontejnerizirati z **Dockerjem** in večvozliščne eksperimente orkestrirati prek **Docker Compose**.
- Ovrednotiti zakasnitve, prepustnost in obnašanje ob napakah (izpadi vozlišč, počasna vozlišča, manjkajoča retention pokritost).

3 Pregled sistema in komponente

Sistem je organiziran okoli štirih glavnih komponent:

3.1 Blockchain plast (L1)

Kot osnovni zapisnik se uporabi obstoječi blockchain. Protokol ne spreminja konsenza. Transakcije (ali zapisi na aplikacijski ravni) vsebujejo reference na off-chain razširitvene bloke. Te reference služijo kot vir resnice pri verifikaciji.

3.2 Razširitveni bloki (off-chain shranjevanje)

Razširitveni bloki hranijo celotna šifrirana sporočila in pripadajoče metapodatke, potrebne za pridobivanje. Vozlišča razširitvene bloke shranjujejo in servirajo glede na svojo kapaciteto/vlogo.

3.3 Vozliščna plast (regular in archive vozlišča)

Vozlišča sodelujejo v protokolu tako, da servirajo razširitvene bloke in izpostavljajo retention metapodatke:

- **Regular vozlišče:** shranjuje le nedavno okno razširitvenih blokov velikosti R (retention okno).
- **Archive vozlišče:** shranjuje celotno zgodovino.

Vozlišče lahko postreže odjemalčevo zahtevo za interval $[s, e]$ (kjer $s \leq e$), če je interval pokrit z njegovim retention oknom pri trenutni višini h. Če ni, se zahteva preusmeri na archive vozlišča.

3.4 Odjemalska plast (lightweight client)

Odjemalci ne sodelujejo v konsenzu in ne shranjujejo blockchaina. Namesto tega:

- z vprašanjem **registra vozlišč** zgradijo lokalni pogled na razpoložljiva vozlišča,
- oddajo zahteve za razpone razširitvenih blokov,
- preverijo pridobljene podatke glede na on-chain reference,
- rekonstruirajo zahtevani razpon z združevanjem odgovorov več vozlišč.

4 Retention model (visok nivo)

Vsako vozlišče oglašuje retention metapodatke, ki določajo, katere višine blokov lahko postreže. Regular vozlišča ohranjajo le nedavne podatke, archive vozlišča pa zagotavljajo popolno pokritost. Takšna delitev izboljša skalabilnost (večina vozlišč shranjuje manj) in hkrati ohranja razpoložljivost (archive vozlišča zagotavljajo dostop do celotne zgodovine).

5 Razbijanje poizvedb in dispatcher

Dispatcher poizvedb je osrednji fokus projekta.

5.1 Razbijanje (dekompozicija)

Za zahtevan interval $[s, e]$ in največjo velikost kosa Δ dispatcher razdeli interval na pod-intervale

$$I_t = [\max(s, end_t - \Delta + 1),\ end_t]$$

in iterira od e nazaj, dokler ni pokrit celoten razpon. Za vsak I_t dispatcher izbere nabor kandidatnih vozlišč, katerih retention metapodatki pokrivajo I_t.

5.2 Vzporedno pridobivanje in preklic

Dispatcher izvaja dve ravni vzporednosti:

- **vzporednost med kosi:** vsi pod-intervali I_t se zahtevajo sočasno.

To zmanjšuje zakasnitve in povečuje robustnost, ko so vozlišča počasna, začasno nedosegljiva ali vračajo napake.

6 Omrežje in implementacijski poudarki

Implementacija bo poudarila praktično omrežno izvedbo:

- protokol zahtevkov/odgovorov prek TCP/HTTP in WebSocketov za:
  - poizvedbe na registru (odkrivanje vozlišč in retention metapodatkov),
  - pridobivanje razširitvenih blokov,
  - nadzorna sporočila dispatcherja (časovne omejitve, preklic).
- jasna ločitev odgovornosti: register, storitev shranjevanja/serviranja na vozlišču, dispatcher in odjemalska verifikacija.
- opazljivost: strukturirani logi in metrike (zakasnitev na kos, število ponovitev, stopnja preklicev).

7 Eksperimentalna postavitev z Dockerjem

Celoten sistem bo zapakiran v Docker slike:

- več vozlišč (regular/archive) zagnanih z različnimi retention parametri,
- kontejner registra,
- en ali več odjemalskih kontejnerjev, ki generirajo poizvedbe,
- Docker Compose scenariji za ponovljive eksperimente (npr. počasno vozlišče, izpad vozlišča, mešana retention pokritost).

8 Načrt evalvacije

Za določitev.

9 Rezultati (deliverables)

1. prototipna implementacija (odjemalec, dispatcher, register, regular/archive storitve na vozliščih),
2. Docker/Docker Compose postavitev za večvozliščni zagon in testiranje,
3. tehnična dokumentacija: arhitektura, protokolarna sporočila in zasnova dispatcherja,
4. končno poročilo in predstavitev z eksperimentalnimi rezultati ter demonstracijo.