Engineer

Latens

⏱ ventetiden mellom spørsmålet og første del av svaret

Hva det er

Latens er hvor lang tid et system bruker på å begynne å svare: tiden fra en forespørsel forlater klienten til den første byten av svaret kommer tilbake. Ingeniører kaller det også forsinkelse eller etterslep, og for en nettverksvei målt dit og tilbake rundturtid, round-trip time (RTT); innen ytelse på nett heter den samme ideen tid til første byte, time to first byte (TTFB). Svartid er den større søsteren, som løper til den siste byten av svaret.

Latens er lett å forveksle med gjennomstrømning og båndbredde, og de tre svarer på forskjellige spørsmål. Båndbredde er hvor mange biter en forbindelse kan bære per sekund. Gjennomstrømning er hvor mye arbeid et system gjør ferdig per sekund. Latens er hvor lenge ett enkelt stykke arbeid venter. En nattlig batchjobb kan ha utmerket gjennomstrømning og elendig latens, og en chatserver det motsatte.

Hvert millisekund av den kommer fra ett av noen få steder:

- **utbredelse**, signalet som krysser avstanden: lys i optisk fiber tilbakelegger omtrent 200 km per millisekund, så en vei på 1 000 km koster minst 10 ms tur-retur før noen programvare har kjørt;
- **overføring**, å legge bytene ut på ledningen, noe som betyr noe for store nyttelaster på trege forbindelser;
- **kø**, venting bak annet arbeid i en buffer i en ruter, en trådpool, en tilkoblingspool eller en lås, den ene komponenten som vokser med belastningen;
- **behandling**, selve arbeidet: spørringen, malen, krypteringen;
- **rundturer i protokollen**, samtalen før den første nyttige byten. En ny HTTPS-tilkobling over TCP trenger først et TCP-håndtrykk og et TLS-håndtrykk, og TLS 1.3 kuttet det fulle håndtrykket sitt ned til én enkelt rundtur.

Arbeidsgruppen IP Performance Metrics i IETF slo fast nettverksvarianten i RFC 2681: forsinkelsen tur-retur løper fra den første biten i en pakke som sendes, til den siste biten i svaret som tas imot. To av valgene der gjelder for enhver tjeneste. Et svar som aldri kommer, regnes som udefinert, uformelt uendelig, i stedet for å bli utelatt fra utvalget i det stille. Og statistikken RFC-en beskriver, er persentiler, medianen og minimumsverdien. Den påpeker også at minimum bare gjenspeiler utbredelse og overføring, mens alt over minimum tyder på overbelastning, et skille som gjelder like godt for et API som for en vei mellom rutere.

For én enkelt webforespørsel deler curl ventetiden opp i stegene sine:

```
curl -s -o /dev/null "$URL" -w '
dns        %{time_namelookup}
connect    %{time_connect}
tls        %{time_appconnect}
first byte %{time_starttransfer}
total      %{time_total}
'
```

Hvert tall er sekunder siden starten, så det er mellomrommene mellom dem som er kostnadene: oppslaget av navnet i DNS, åpningen av tilkoblingen, TLS-håndtrykket og så serverens egen tenketid frem til første byte.

Bruk begrepet i en beslutning

Bruk «Hva er latens?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Overvåking og varsler som sier fra mens en feil fortsatt er liten, og tallene som viser hvor ofte den holder seg liten.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Bygde selskapets første observerbarhetsdashbord — selskapets første innsyn i egne systemer

Bygde selskapets første observerbarhetsdashbord Ta kontakt →

Hvorfor det er viktig

Latens er det en bruker kjenner; gjennomstrømning er det serverregningen gjenspeiler. Når en side er treg, er spørsmålet hvor pengene skal brukes. De fem komponentene ovenfor trenger forskjellige løsninger; fire av dem viser hvorfor kjøp av maskinvare ikke er et generelt botemiddel. Avstand reduseres ved å flytte nærmere: en cache, et CDN, en region nær brukerne. Tiden brukt på rundturer

reduseres ved å ha færre av dem: gjenbruke tilkoblinger, samle spørringer i batcher, hente parallelt. Behandlingstiden reduseres ved å gjøre mindre arbeid: en indeks, en bedre spørreplan. Mer kapasitet kan redusere køventingen; raskere prosessorer, mer minne eller bedre I/O kan også redusere behandlingstiden når disse ressursene er flaskehalsen. En større server er ikke automatisk raskere: mål hvor tiden går før du velger maskinvare.

Gjennomsnitt får problemet til å se mindre ut enn det er, fordi brukerne møter halen. En side som kaller ti tjenester i backend parallelt, venter på den tregeste. Hvis hvert kall uavhengig av de andre har 1 % sannsynlighet for å være tregt, treffer $1 - 0{,}99^{10}$, ca. 9,6 % av sidevisningene, minst ett tregt kall. En sjelden forsinkelse i delene blir en vanlig forsinkelse i helheten.

Derfor skrives seriøse mål i persentiler, for eksempel at 99 % av forespørslene besvares innen 300 ms over 30 dager, og derfor er den 99. persentilen som regel tallet det er verdt å følge med på. Medianen sier hvordan tjenesten føles en god dag; de høye persentilene sier hvordan den føles for kundene som er i ferd med å gå.

Slik ser det ut i praksis

I en geografisk kartapplikasjon gikk latensen ved gjengivelse av kartlag og spørringer mot stedsbaserte data ut over både brukerne og påliteligheten til backend. Arbeidet var rettet mot behandling av romlige spørringer i en backend som brukte en eldre Microsoft SQL Server-database. Overgangen til PostgreSQL med PostGIS kuttet gjennomsnittlig svartid fra 2,5 sekunder til under 250 millisekunder, en forbedring av ytelsen til romlige spørringer med en faktor 10. Bak dette lå GiST- og SP-GiST-indekser på de romlige kolonnene, mer enn 50 millioner migrerte poster og en konfigurasjon justert for samtidig tilgang; CPU-belastningen i backend falt med 65 %.

Latens setter også et tak for gjennomstrømningen der det er kamp om én rad. Et notat om lynsalg der alle kjøperne skriver til den samme raden setter taket for én enkelt vare til omtrent hvor mange salg én lås bærer, delt på hvor lenge låsen holdes. Alt som er tregt inne i låsen, senker det taket: et kall til betaling som gjøres mens låsen holdes, tar det ned til rundt to salg i sekundet. Selv et sikkerhetstiltak har en pris her. En synkron standby i PostgreSQL legger latens til hver commit, og det forlenger nettopp den holdetiden for låsen som taket deles på.

Og for å finne ut hvor tiden går, registrerer den egenutviklede overvåkingen på én plattform latensen i API-et i sin egen database og sporer forespørsler med OpenTelemetry, slik at en treg forespørsel kan følges på tvers av frontenden, API-et og databasen i stedet for å gjettes på.

Hvor det går galt

Gjennomsnittet skjuler halen. Et snitt på 100 ms er forenlig med at én forespørsel av femti tar tre sekunder. Registrer latensen som et histogram, og les persentilene fra det. Pass så på hvordan de kombineres: gjennomsnittet av den 99. persentilen fra fem servere er ikke den 99. persentilen for hele maskinparken. Prometheus poengterer det selv i dokumentasjonen sin: kvantiler som er beregnet av en summary, kan ikke aggregeres, mens bøttene i et histogram kan summeres først og kvantilen beregnes etterpå.

Tidtakeren starter på feil sted. En server som måler fra øyeblikket håndtereren starter, går glipp av tiden forespørselen tilbrakte i lastbalansereren, i køen av ventende tilkoblinger eller i ventingen på en ledig tilkobling fra poolen. Lastgeneratorer som venter på hvert svar før de sender neste forespørsel, sender færre forespørsler akkurat når systemet står fast, så stoppet synes knapt i resultatene. Mål fra klienten også, og la serveren melde sin egen andel i en standard header:

```
Server-Timing: db;dur=53, app;dur=47.2
```

Samtalen er for pratsom. En side som kjører én spørring per rad i en liste, betaler én rundtur per rad. Med databasen i en annen region, 30 ms unna per rundtur, koster hundre rader tre sekunder før noen spørring har gjort ekte arbeid. Én spørring med en join, eller ett samlet kall, betaler avstanden én gang.

Nye forsøk ganger opp belastningen. En klient som gir opp etter ett sekund og prøver på nytt tre ganger, gjør én treg server om til fire ganger så mye trafikk, og det gjør den enda tregere. Gi hver forespørsel en samlet frist, vent eksponentielt lenger mellom forsøkene med en tilfeldig variasjon, og sett et tak på antallet nye forsøk.

Les mer

- RFC 2681: A Round-trip Delay Metric for IPPM — forsinkelse tur-retur definert ned til biten, med tapte svar og persentiler uttrykkelig behandlet.
- RFC 7679: A One-Way Delay Metric for IP Performance Metrics — motstykket for forsinkelse én vei, utgitt som internettstandard.
- Server Timing — W3C-headeren som lar en server fortelle nettleseren hvor tiden ble av.
- Histograms and summaries — hvorfor kvantiler fra en summary ikke kan kombineres, mens bøtter i et histogram kan.
- RFC 8446: TLS 1.3 — utformingen av håndtrykket, også rundturen det fjerner.

Ordliste

Velg ut fra deres faktiske rammer

Ta med utrullingsløpet, driftsrisikoene og dem som har vakt. Vi kan avgrense en endring med kontroller, en vei tilbake og tydelig ansvar.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er latens?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

welcome@engineer.company +45 50 10 61 84 København, Danmark

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/latency/