YUKLAMALARNI MUVOZANATLASH, NGINX MISOLIDA

Abduaziz Ziyodov
Toshkent shahridagi Inha Universiteti talabasi
abduaziz.ziyodov@mail.ru
https://doi.org/10.5281/zenodo.8231993

ARTICLE INFO

Qabul qilindi: 31-July 2023 yil
Ma'qullandi: 05-August 2023 yil
Nashr qilindi: 10-August 2023 yil

KEY WORDS

Ko'plab foydalanuvchilarga ega kompaniya, ko'plab mijozlarining so'rovlarini birgina server bilan qayta ishlashi ancha mushkul.

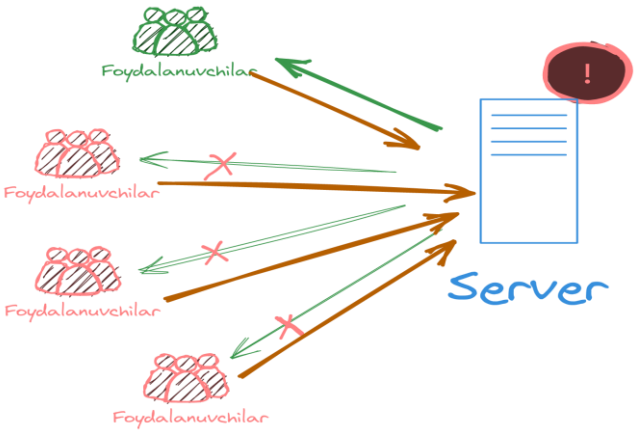
ABSTRACT

Axborot texnologiyalari, veb ilovalar va server texnologiyalar shiddat bilan rivojlangani sari, ularga tushadigan yuklanishlar va so'rovlar soni ham keskin ortib bormoqda. Ko'plab foydalanuvchilarga ega kompaniya, ko'plab mijozlarining so'rovlarini birgina server bilan qayta ishlashi ancha mushkul. Maqola katta hajmdagi yuklamalar qay tarzda muvozanatlanishi va amaliyotda NGINXning ushbu vazifani qanday hal qilishini batafsil muhokama qiladi.

Yuklamalarni bir nechta serverlar o'rtasida taqsimlash, muvozanatlash ilovadagi kechikishlar vaqtini kamaytirish va resurslar sarfini optimallashtirish uchun keng qo'llaniladigan metod hisoblanadi. Server qanchalik kuchli bo'lmasin, bir vaqtning o'zida ochiq/ushlab turilishi mumkin bo'lgan ulanishlar soni cheklangan. Ushbu cheklovning oshib ketishi, serverning ishdan chiqishiga sababchi bo'ladi.

Yuklamalarni muvozanatlash. Tarixi va turlari

Ushbu metod 1990 yillardan "hardware"lar uchun katta trafik(oqim)ni tarmoq bo'ylab



taqsimlash bilan boshlangan. Tashkilotlar serverlarining samaradorligini oshirishni istashgan. Avvallari ushbu metod maxsus qurilma(hardware) asosida amaliyotga tadbiq qilingan. Asosiy muammo bir vaqtning o'zida keladigan haddan tashqari ko'p so'rovlarga javob qaytarish bo'lgan. Ushbu turdagi dasturiy ta'minotlar OSI modeli bo'yicha 2ta qatlamda ish olib boradi.

- **“Layer-4”**
  4-qatlamda ish olib boradigan dasturiy ta’minotlar, TCP va UDPga asoslangan trafik paketlari qayerga va qay tarzda yo’naltirilishi haqida qaror qabul qilishadi. Ular kelayotgan paketning kontentini tekshirishmaydi, shunchaki ma’lum port va IP manzil bo’yicha yo’naltirishadi xolos.
- **“Layer-7”**
  7-qatlam dasturiy ta’minotlari esa HTTP so’rovdan kelgan “header”ning kontentini “titib” ko’rish va umuman olganda ularga kelgan so’rovlarning barcha ma’lumotlarini izlab ko’rish huquqiga ega. Ular L-4da qo’llaniladigan dasturiy ta’minotlarga nisbatan ko’proq protsessor kuchidan foydalanishadi, ammo amaliyotda juda samarador. Ushbu turdagi muvozanatlash “Global Server Load Balancing” deb ham yuritiladi.

“Round-robin” uslubi

Muvozanatlashning engi sodda texnikasi hisoblanadi. Kelayotgan barcha so’rovlarni ketma-ketlikda serverlarga yo’naltiradi. Umuman olganda uni “statik” deb atash ham mumkin. Algoritmik tushuntirilishi:

- I - kelayotgan so’rovning tartib raqami
- T - mavjud serverlarning soni

Ushbu uslub, kelayotgan so’rovni i mod T tartib raqamli serverga yo’naltiradi hamda i ning qiymatini birga oshiradi, va ushbu amallar so’rovlar tugamagunigacha davom etadi. Ushbu algoritmning adaptiv emasligi uning asosiy yomon tomoni hisoblanadi.

“Least connections” uslubi

Ushbu uslub har bir serverga ulangan aktiv foydalanuvchilar sonini hisobga oladi. Kamroq ulanishlarga ega bo’lgan serverga kelayotgan so’rovlarni yo’naltiradi. Algoritmik tushuntirilishi

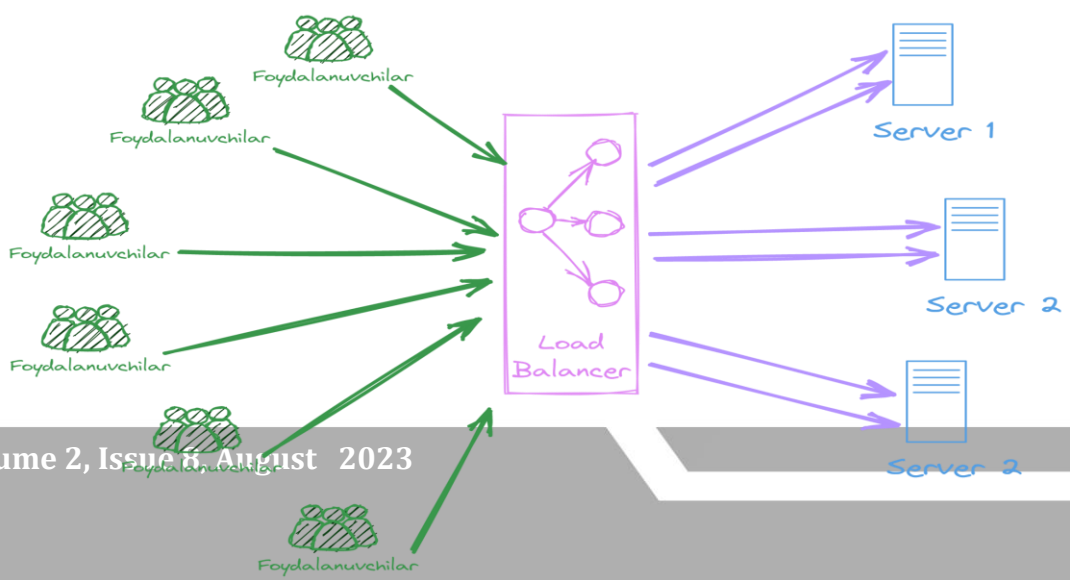
- Serverlar ro’yxatidan eng kam ulanishlarga ega serverni topish.
- So’rovni unga yo’naltirish va so’rovlar qolmagunga qadar buni takrorlash.

“IP-hash” uslubi

Kelayotgan so’rovdan IP manzilni heshlash uchun kalit sifatida oladi, va uning maxsus yo’nalishini topadi (hesh jadval kabi ishlaydi). Hesh jadvallar bir kalit uchun har doim bir xil kalit qaytargani kabi, har bir mijozning IP manzilini aynan bir serverga yo’naltiriladi.

Nginx haqida

Nginx bu Igor Sysoev tomonidan ishlab chiqilgan HTTP, teskari proksi server, pochta serveri va TCP/UDP server hisoblanadi. Ko’p muddat davomida rossiyaning juda katta kompaniyalari shu jumladan Yandex, Mail.Ru, VK va Rambler tomonidan ishlatilib



kelinmoqda. Asinxronligi, va bitta “thread” bilan so’rovlarni amalga oshirishi uning juda kam xotira ishlatishiga sababchi hisoblanadi. So’rovlar bir birini yopib qo’ymaydi, va ular maxsus “ishchi”lar tomonidan qayta ishlanadi. Maqolada esa nginx amaliyotda yuklamalarni muvozanatlashda qanday qo’llanishini ko’rib o’tamiz.

Trafikni serverlar guruhiga yo’naltirish

Serverlar guruhi upstream direktivi ichiga belgilanadi. Upstream direktivining o’zi esa http’ning ichida sozlanadi hamda o’z ichida serverlarning IP manzillarini qamrab oladi:

```
http {
    upstream serverlar {
        server server1.example.com;
        server server2.example.com;
        server server3.example.com;
    }
}
```

HTTP so’rovlarni qayta ishlash uchun bizga server zarur buning uchun server direktividan foydalanamiz:

```
server {
  ....
}
```

Serverimiz unga kelgan so’rovlarni yuqorida belgilangan upstream ya’ni serverlar guruhiga yo’naltirishi lozim. Buning uchun biz proksi sozlamalaridan foydalanamiz:

```
server {
    location / {
        proxy_pass http://serverlar;
    }
}
```

Va bizning to’liq server sozlamalarimiz quyidagi ko’rinishga keladi:

```
http {
    upstream serverlar {
        server server1.example.com;
        server server2.example.com;
        server server3.example.com;
    }
    server {
        location / {
            proxy_pass http://serverlar;
        }
    }
}
```

Ushbu sozlama juda sodda hisoblanadi va hamda muvozanatlash metodi ko'rsatilmagani uchun "round-robin" uslubidan foydalaniladi. Bunda serverlarga vaznlar ham berishimiz mumkin. Bu xuddi serverlarning ahamiyatini baholashga o'xshaydi.

Xulosa

Ushbu texnika yuqori trafikga ega va katta yuklamalar ustida ishlaydigan veb servislar uchun o'ta muhim sanaladi va hozirgi zamonaviy veb olamining ustuni deyish ham mumkin. Server resurslarini samarali ishlatib, bir nechta kichik klasterlarga bo'lib ham muvozanatlash texnikasini qo'llashimiz mumkin. Bunda nginx bizga har doim yordamga tayyor.

Manbalar, Adabiyotlar:

1. https://www.cdnetworks.com/knowledge-center/what_is_load_balancing
2. https://kinsta.com/knowledgebase/what-is-nginx/
3. https://docs.nginx.com/nginx/admin-guide/load-balancer/http-load-balancer/
4. http://nginx.org/en/docs/http/load_balancing.html
5. https://aws.amazon.com/what-is/load-balancing/