|  |
| --- |
| Nama : Aulia Rahmi |
| Nim : 2301081001 |
| Kelas : Teknik Komputer 2-A |
| Matakuliah : Microservice - Teori |

**1. Jelaskan Tentang Server side Discovery**

* Server-Side Discovery

Server-side discovery adalah pola di mana load balancer atau komponen serversentral bertanggung jawab untuk menemukan dan mengarahkan permintaan klien ke instance layanan yang tersedia. Prosesnya melibatkan:

* Klien mengirimkan permintaan ke load balancer.
* Load balancer mengkueri *service registry* untuk mendapatkan daftar instance layanan yang tersedia.
* Load balancer memilih instance yang sesuai (berdasarkan algoritma seperti round-robin) dan meneruskan permintaan tersebut.

Contoh Implementasi:

* Kubernetes menggunakan *kube-proxy* untuk menangani server-side discovery.
* AWS Elastic Load Balancer (ELB) juga merupakan contoh implementasi server-side discovery.

Kelebihan:

* Abstraksi untuk klien: Klien tidak perlu mengetahui detail lokasi layanan.
* Sederhana untuk klien: Tidak memerlukan logika discovery di sisi klien, sehingga lebih mudah untuk dikembangkan.
* Integrasi bawaan di beberapa platform: Beberapa lingkungan cloud (seperti AWS) menyediakan mekanisme ini secara otomatis.

Kekurangan:

* Ketergantungan pada load balancer: Jika load balancer gagal, seluruh sistem dapat terpengaruh.
* Biaya tambahan: Memerlukan infrastruktur tambahan untuk load balancer dan pengelolaannya.
* Kurang fleksibel: Klien tidak dapat memilih instance layanan secara langsung.

**2. Jelaskan Tentang Cleint side Discovery**

* Client-Side Discovery

Pada client-side discovery, klien bertanggung jawab untuk menemukan lokasi layanan secara langsung melalui *service registry*. Prosesnya meliputi:

* Klien mengakses *service registry* untuk mendapatkan daftar instance layanan.
* Klien memilih instance menggunakan algoritma load balancing tertentu.
* Klien mengirimkan permintaan langsung ke instance yang dipilih.

Contoh Implementasi:

* Netflix Ribbon adalah pustaka yang mendukung client-side discovery.
* Eureka (dari Netflix OSS) sering digunakan dalam pola ini.

Kelebihan:

* Fleksibilitas tinggi: Klien dapat menerapkan logika load balancing khusus sesuai kebutuhan aplikasi.
* Efisiensi langsung: Tidak ada perantara tambahan seperti load balancer, sehingga latensi lebih rendah.
* Penghematan biaya: Tidak memerlukan infrastruktur tambahan seperti load balancer.

Kekurangan:

* Kompleksitas di sisi klien: Klien harus memiliki logika discovery bawaan, yang dapat menambah beban pengembangan.
* Ketergantungan pada registry: Jika registry gagal atau tidak tersedia, klien tidak dapat menemukan layanan.
* Kesulitan integrasi lintas bahasa: Setiap klien harus memiliki pustaka discovery yang kompatibel dengan bahasa pemrograman yang digunakan.

**3. Jelaskan kekurangan dan kelebihan dari Server Side Discovery dan Client Side Discovery**

* Perbandingan Kelebihan dan Kekurangan

|  |  |  |
| --- | --- | --- |
| **Aspek** | **Server-Side Discovery** | **Client-Side Discovery** |
| Kompleksitas | Sederhana di klien, kompleks di infrastruktur | Kompleks di klien, sederhana di infrastruktur |
| Biaya Infrastruktur | Lebih mahal karena membutuhkan load balancer | Lebih hemat karena tidak membutuhkan load balancer |
| Fleksibilitas Load Balancing | Terbatas pada kemampuan load balancer | Fleksibel dengan algoritma custom |
| Ketahanan Sistem | Rentan terhadap kegagalan single point (load balancer) | Lebih resilien dengan caching data registry |
| Integrasi Lintas Bahasa | Mudah karena abstraksi dilakukan di server | Kompleks karena setiap klien memerlukan pustaka khusus |

**4. Jelaskan tentang Service Registry**

* Service Registry

Service registry adalah komponen inti dalam mekanisme service discovery. Registry bertindak sebagai basis data terpusat yang menyimpan informasi tentang semua instance layanan dalam sistem, termasuk alamat IP, port, status kesehatan, dan metadata lainnya.

Fungsi Utama:

* Registrasi Layanan:
  + Layanan mendaftar secara otomatis (*self-registration*) atau melalui pihak ketiga (*third-party registration*).
* Pencarian Layanan:
  + Klien atau load balancer mengkueri registry untuk menemukan lokasi layanan aktif.
* Monitoring Kesehatan:
  + Registry sering kali memeriksa status kesehatan layanan menggunakan *heartbeat* atau mekanisme lain.

Contoh Tools Service Registry:

* Consul**:** Mendukung multi-datacenter dan ACL untuk keamanan.
* Eureka**:** Dirancang oleh Netflix untuk arsitektur cloud-native.
* ZooKeeper**:** Digunakan secara luas dalam sistem terdistribusi.

Kelebihan Service Registry:

* Memungkinkan pencarian layanan secara dinamis meskipun lokasi layanan berubah secara terus-menerus.
* Mendukung skalabilitas dengan menambahkan atau menghapus instance layanan secara otomatis.

Kekurangan Service Registry:

* Membutuhkan pemeliharaan agar tetap tersedia (*high availability*), terutama dalam sistem besar.
* Jika registry gagal, seluruh sistem dapat terganggu.