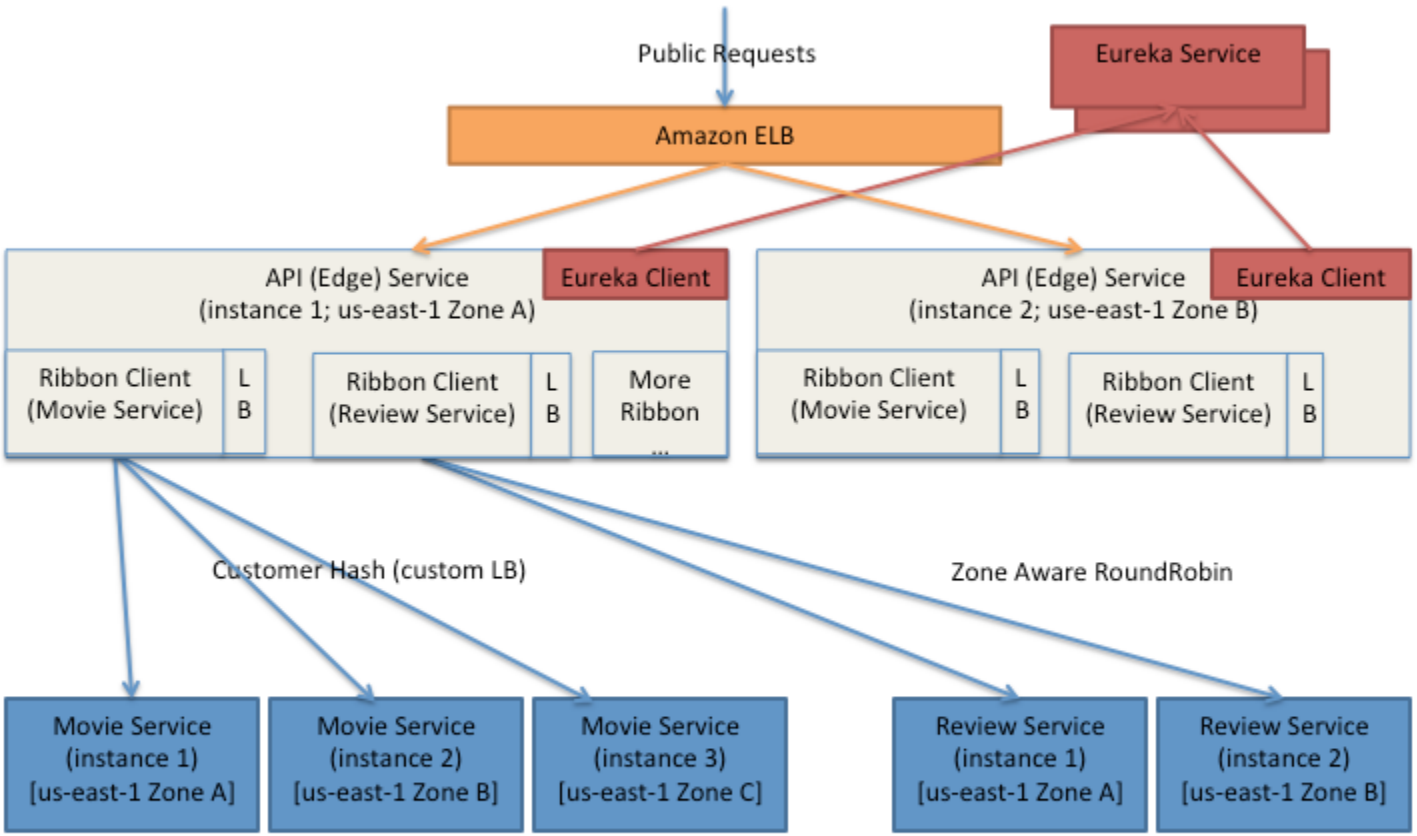
Spring Cloud Netflix: Client Side Load Balancing with Ribbon

Spring Cloud Netflix: Client Side Load Balancing

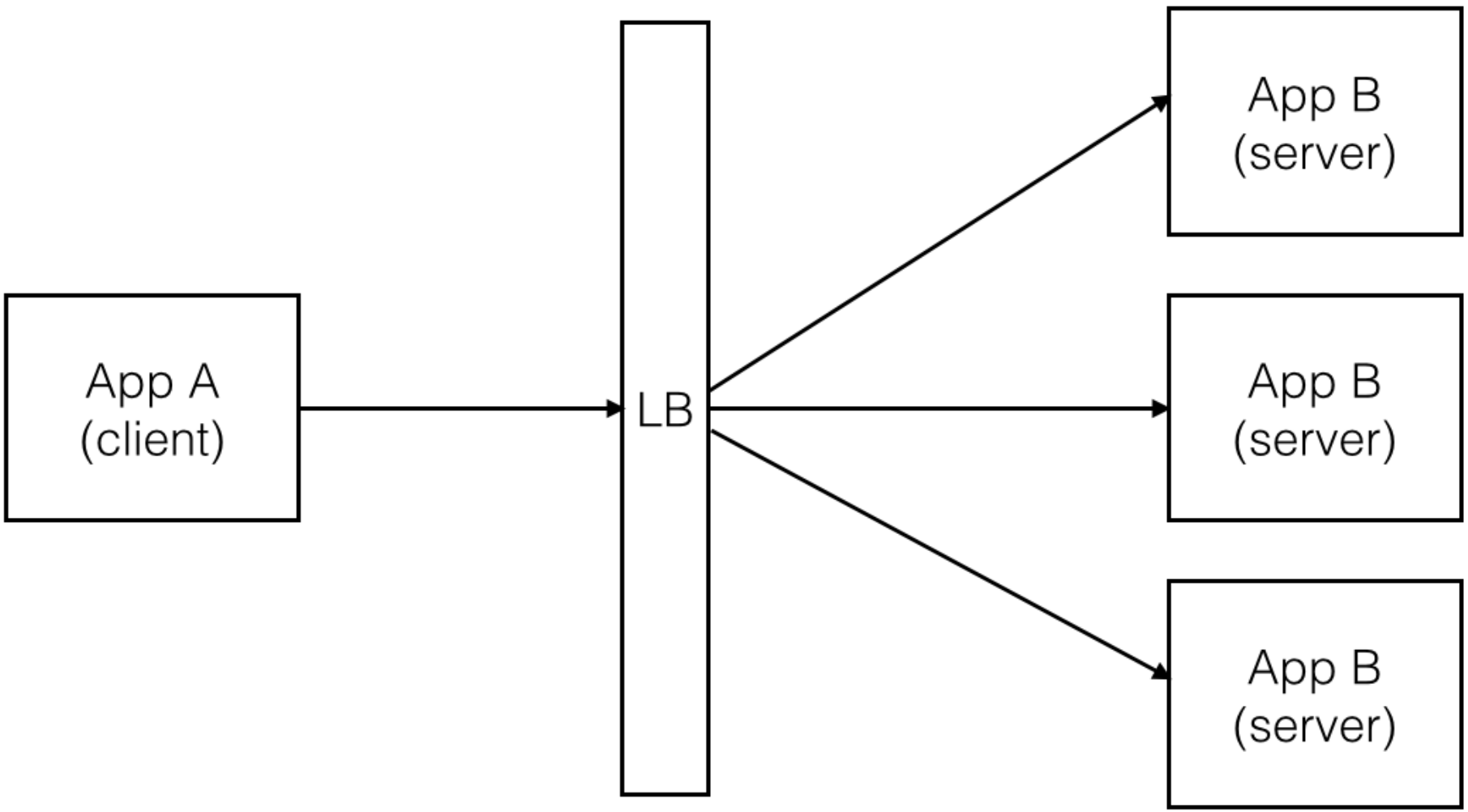
- Review Load Balancing
- Ribbon
- Ribbon with Spring

Some Context for This Discussion

- This is load balancing with regard to middle-tier applications
- This is NOT about load balancing for public facing applications



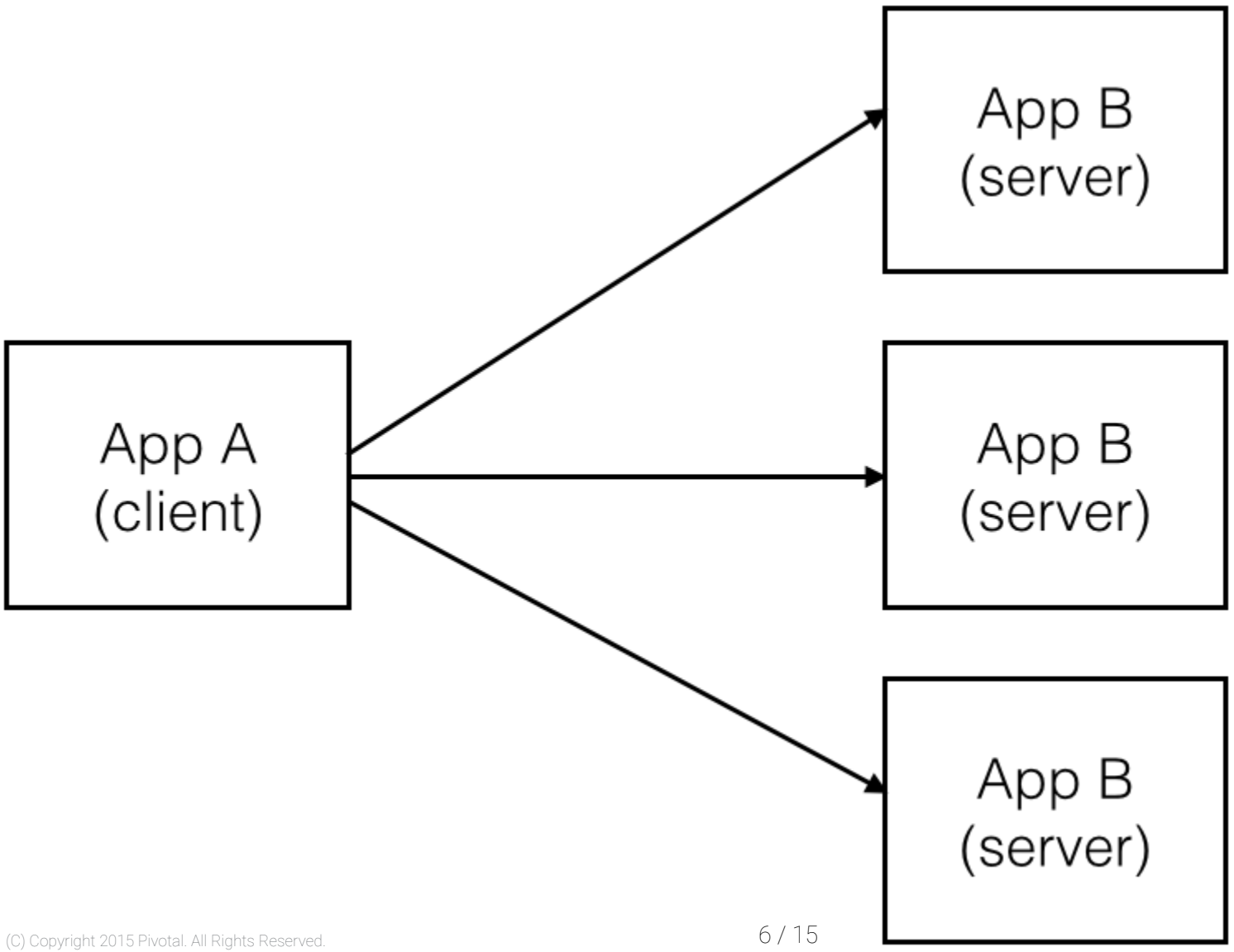
Server Side Load Balancing



Server Side Load Balancing Challenges

- One size does not fit all
- Multiple setup/configurations
- Extra hops
- Not cloud/app aware (zones, circuit-breakers, etc.)
- Limited algorithms

Client Side Load Balancing



Client Side Load Balancing Benefits

- Pick the right algorithm for your app
- No additional setup, just deploy apps
- 1 less hop
- Cloud/app aware (zones, circuit-breakers, etc.)
- Extensible algorithms

Spring Cloud Netflix: Client Side Load Balancing

- Review Load Balancing
- Ribbon
- Ribbon with Spring

Ribbon

Ribbon is a client side IPC (Inter Process Communication) library that is battle-tested in cloud. The primary usage model involves REST calls with various serialization scheme support.

Features:

- Multiple and pluggable load balancing rules
- Integration with service registry (Eureka)
- Built-in failure resiliency
- Cloud enabled
- Clients integrated with load balancers
- Archaius configuration driven client factory

Ribbon Load Balancing Algorithms

- Simple Round Robin LB
- Weighted Response Time LB
- Zone Aware Round Robin LB
- Random LB

Zone Aware Load Balancer

1. The LB calculates/examines zones stats in real time. If average active requests have reached a configured threshold the zone will be dropped.

2. Once the the worst zone is dropped, a zone will be chosen among the rest with the probability proportional to its number of instances.

3. A server will be returned from the chosen zone with a given Rule.

Source: http://techblog.netflix.com/2013/01/announcing-ribbon-tying-netflix-mid.html

Spring Cloud Netflix: Client Side Load Balancing

- Review Load Balancing
- Ribbon
- Ribbon with Spring

Using the API Directly

Use the **`LoadBalancerClient`**.

Example:

```
public class MyClass {

    @Autowired
    private LoadBalancerClient loadBalancer;

    public void doStuff() {

        ServiceInstance instance =
          loadBalancer.choose("stores");

        URI storesUri = URI.create(String.format("http://%s:%s",
          instance.getHost(), instance.getPort()));

        // ... do something with the URI
    }
}
```

Spring RestTemplate as a Load Balancer Client

`@LoadBalanced` is a qualifier to make certain the load balanced restTemplate is injected

```
public class MyClass {
    @Autowired
    @LoadBalanced
    private RestTemplate restTemplate;

    public String doOtherStuff() {
        String results =
          restTemplate.getForObject("http://stores/stores",
          String.class);
        return results;
    }
}
```

Customizing the Ribbon Client

Spring Cloud Netflix provides the following beans by default for ribbon (BeanType beanName: ClassName):

- **`IRule`** ribbonRule: **`ZoneAvoidanceRule`**
- **`IPing`** ribbonPing: **`NoOpPing`**
- **`ILoadBalancer`** ribbonLoadBalancer: **`ZoneAwareLoadBalancer`**
- ...

Creating a bean of one of those types and placing it in a @RibbonClient configuration allows you to override each one of the beans described. Example:

```
@Configuration
@RibbonClient(name = "foo", configuration = FooConfiguration.class)
public class FooConfiguration {
    @Bean
    public IPing ribbonPing(IClientConfig config) {
        return new PingUrl();
    }
}
```

This replaces the NoOpPing with PingUrl.