Enniu微服务框架简介

Microservice

- The microservice architectural style is an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API.
- These services are built around business capabilities and independently deployable by fully automated deployment machinery.
- There is a bare minimum of centralized management of these services , which may be written in different programming languages and use different data storage technologies.

– by Martin Fowler

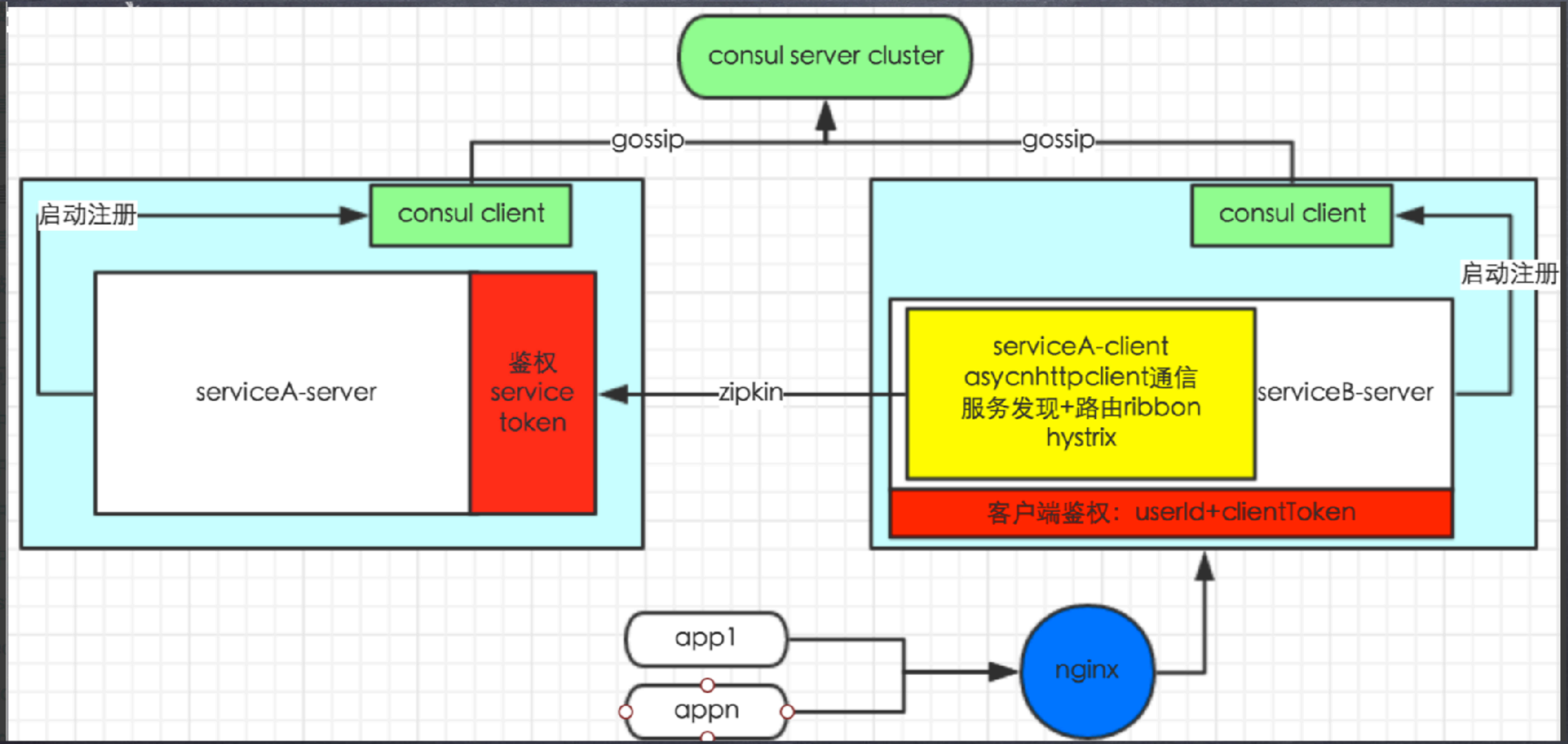
Base简介

- 基础框架：Spring Boot, Spring Cloud
- 服务注册、发现、健康检查：Consul
- 配置管理: Consul KV
- 服务通信：Retrofit + Async Http Client
- 统一文档：Swagger
- 日志管理：Logback + ELK
- 服务容错：Hystrix
- 监控计数：Prometheus
- 链路跟踪: zipkin, spring-cloud-sleuth
- 统一网关: Janus

Consul

- Server Agent
  - 高可用集群
  - 跨数据中心感知
  - Raft分布式一致协议，与Zookeeper Paxos的比较，更简洁易实现
- Client Agent
  - 与server agent通信
  - 负责转发接口请求到server集群

Consul


consul server cluster
gossip
gossip
启动注册
consul client
consul client
启动注册
serviceA-server
鉴权
service
token
zipkin
serviceA-client
asycnhttpclient通信
服务发现+路由ribbon
hystrix
serviceB-server
客户端鉴权：userId+clientToken
app1
appn
nginx

Base Server-服务注册

- jave -jar -Dserver.port=8080 -Dservice.tag=prod enniu-service-foo-1.0.jar
- EnniuServiceRegister监听ContextRefreshedEvent
- bootstrap.properties
  - service.name, service.tag, health check

Base Server-服务注册

- 微服务和consul agent运行在同一台机器上

```
java -jar -Dserver.port=8080 -Dservice.tag=dev enniu-microservice.jar
```

- 微服务和consul agent运行在不同机器上

```
java -jar -Dserver.port=8080 -Dservice.tag=dev -Dhost.ip=10.0.40.3 -
Dconsul.agent.ip=10.0.40.2 enniu-microservice.jar
```

- 微服务运行在docker里面，consul agent运行在docker宿主机

```
java -jar -Dserver.port=8080 -Dservice.tag=dev -Dhost.ip=10.0.40.2 -
Dexternal.port=10080 enniu-microservice.jar
```

Base Server-服务注册

- GC log: -Xloggc
- Dump log: -XX:HeapDumpPath
- 服务自己配置不同环境jvm args: -Xms256m -Xmx1024m ...

Base Server-统一配置

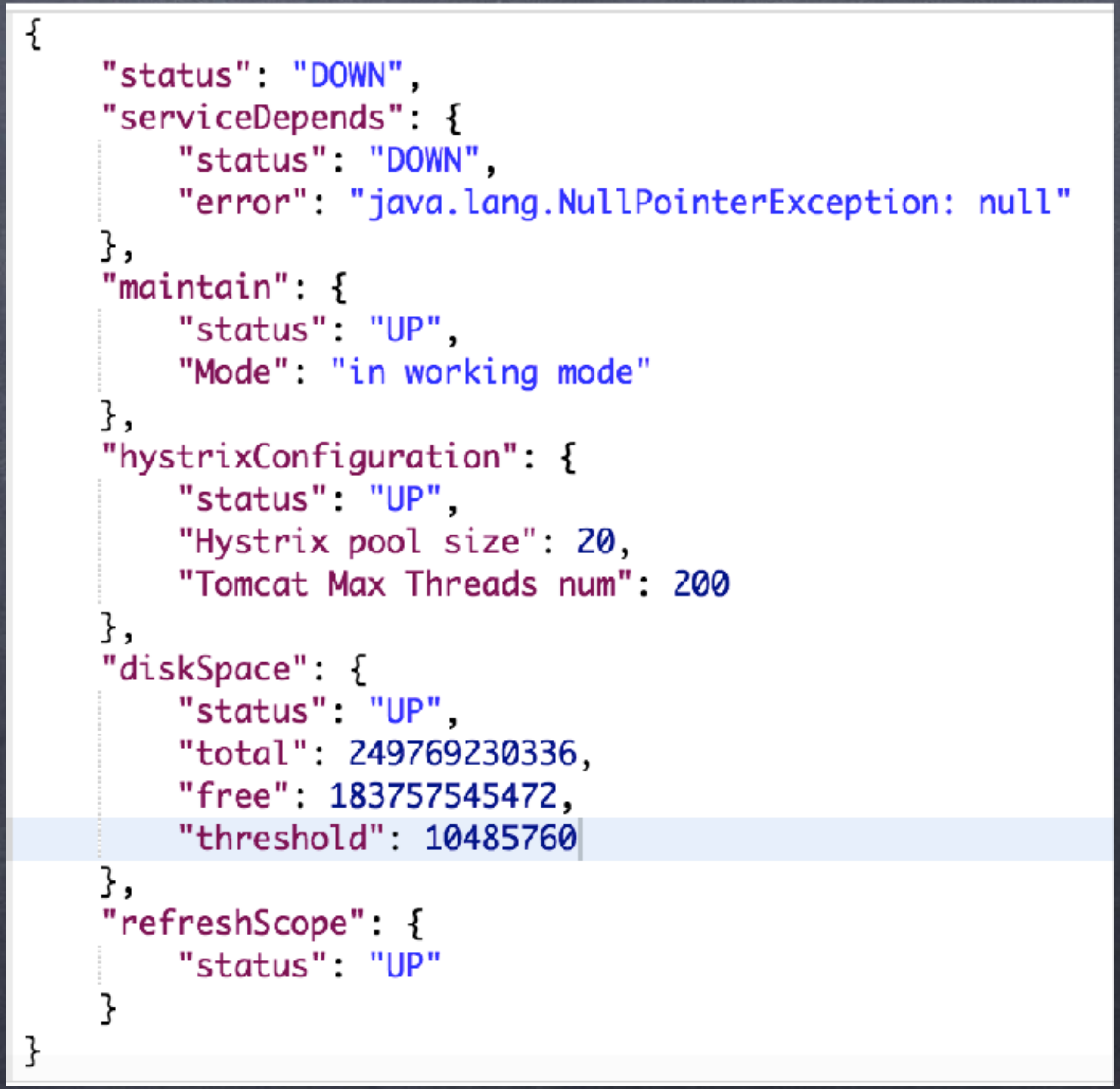
- Consul KV存储
- /service/foo/prod/config
  - -Dservice.tag=local直接从application.properties读取
- ConfigurationManager.getConfigInstance().getString("key", "default value");
- @Value的配置需重启服务

Base Server - 健康检查

- bootstrap.properties
- enniu.cloud.consul.checks.http.url
- enniu.cloud.consul.checks.http.interval
- enniu.cloud.consul.checks.http.timeout

Base Server - 健康检查

- 4种状态DOWN > OUT_OF_SERVICE > UP > UNKNOWN
- HealthAggregator将各Indicator的检测状态聚合成一个状态
- ServiceDependsHealthIndicator
  - enniu.cloud.service.health.dependency.services
- HystrixConfigurationHealthIndicator
- MaintainHealthIndicator
  - 维护模式 /tmp/{service.name}-{external.port}.DOWN
- 实现接口HealthIndicator自定义Indicator



```
{
    "status": "DOWN",
    "serviceDepends": {
        "status": "DOWN",
        "error": "java.lang.NullPointerException: null"
    },
    "maintain": {
        "status": "UP",
        "Mode": "in working mode"
    },
    "hystrixConfiguration": {
        "status": "UP",
        "Hystrix pool size": 20,
        "Tomcat Max Threads num": 200
    },
    "diskSpace": {
        "status": "UP",
        "total": 249769230336,
        "free": 183757545472,
        "threshold": 10485760
    },
    "refreshScope": {
        "status": "UP"
    }
}
```

Base Server-鉴权

- HTTP Header Authorization
  - service F8113C16-4BDD-4048-8C2C-14AE7A688835
  - encrypt 4ZHZDMGjMD5RFKyAeDiXZpPtIXS8y7TX...
- Token type: SERVER
  - @Internal, 正则匹配
- Token type: CLIENT
  - 调用usercenter service, 5min redis cache
- 直接对外服务（gateway）使用CLIENT模式，不直接对外服务使用SERVER模式

Swagger

- http://{host}:{port}/swagger-ui.html#/

gnh-user Api

gnh-user相关Api

gnh-user-controller : Gnh User Controller

Show/Hide
List Operations
Expand Operations

POST /gnh-user/api/v1/gnh-users 创建gnh-user.

GET /gnh-user/api/v1/gnh-users/{id} 获取gnhUser.

PUT /gnh-user/api/v1/gnh-users/{id} 更新gnhUser.

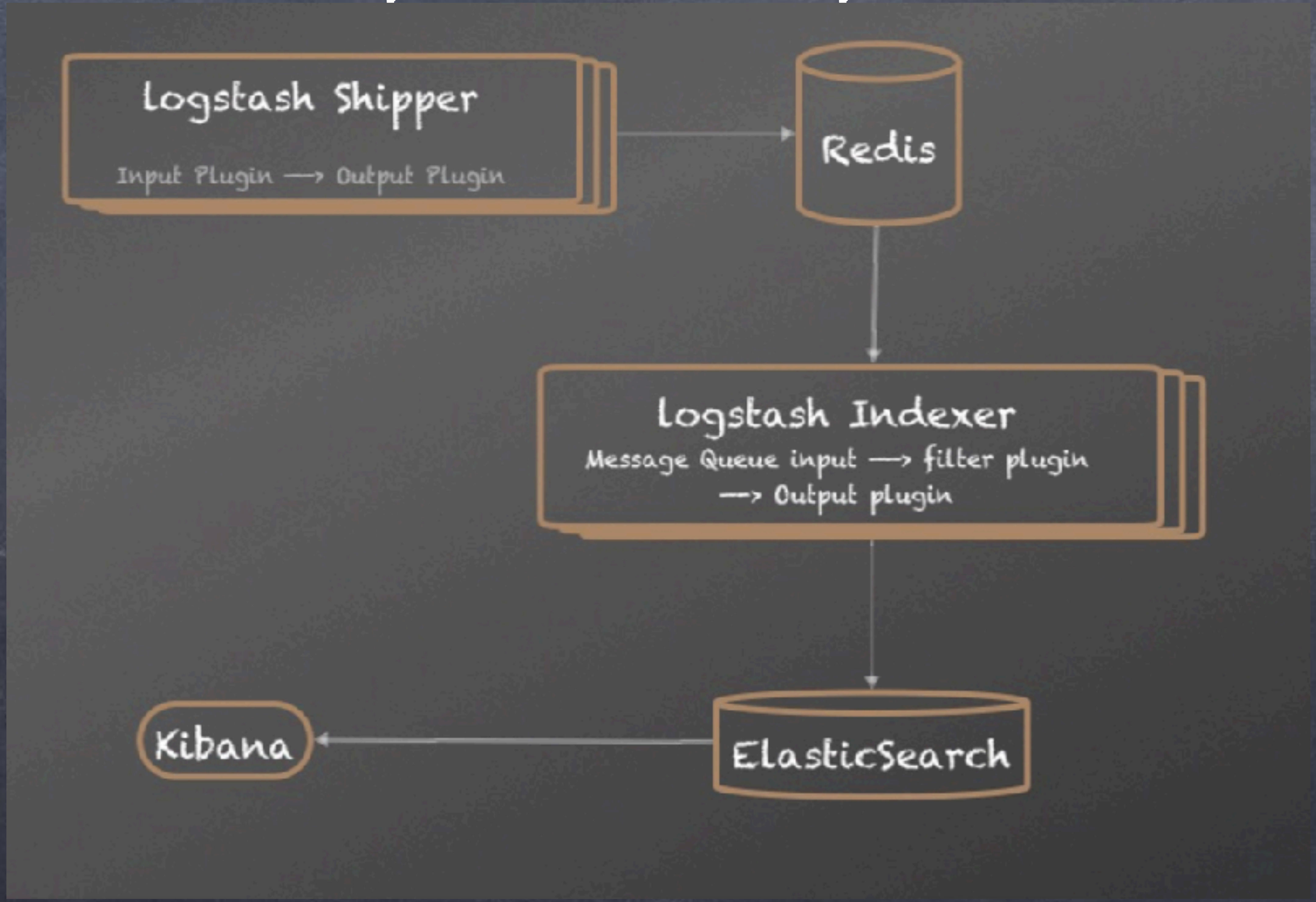
[BASE URL: / , API VERSION: V1]

Swagger

- @Api
- @ApiOperation
- @ApiParam
- @ApiImplicitParam
- @ApiResponse
- @ApiModel

Log

- logback + ELK
- http://wiki.51.nb/pages/viewpage.action?pageId=15107111



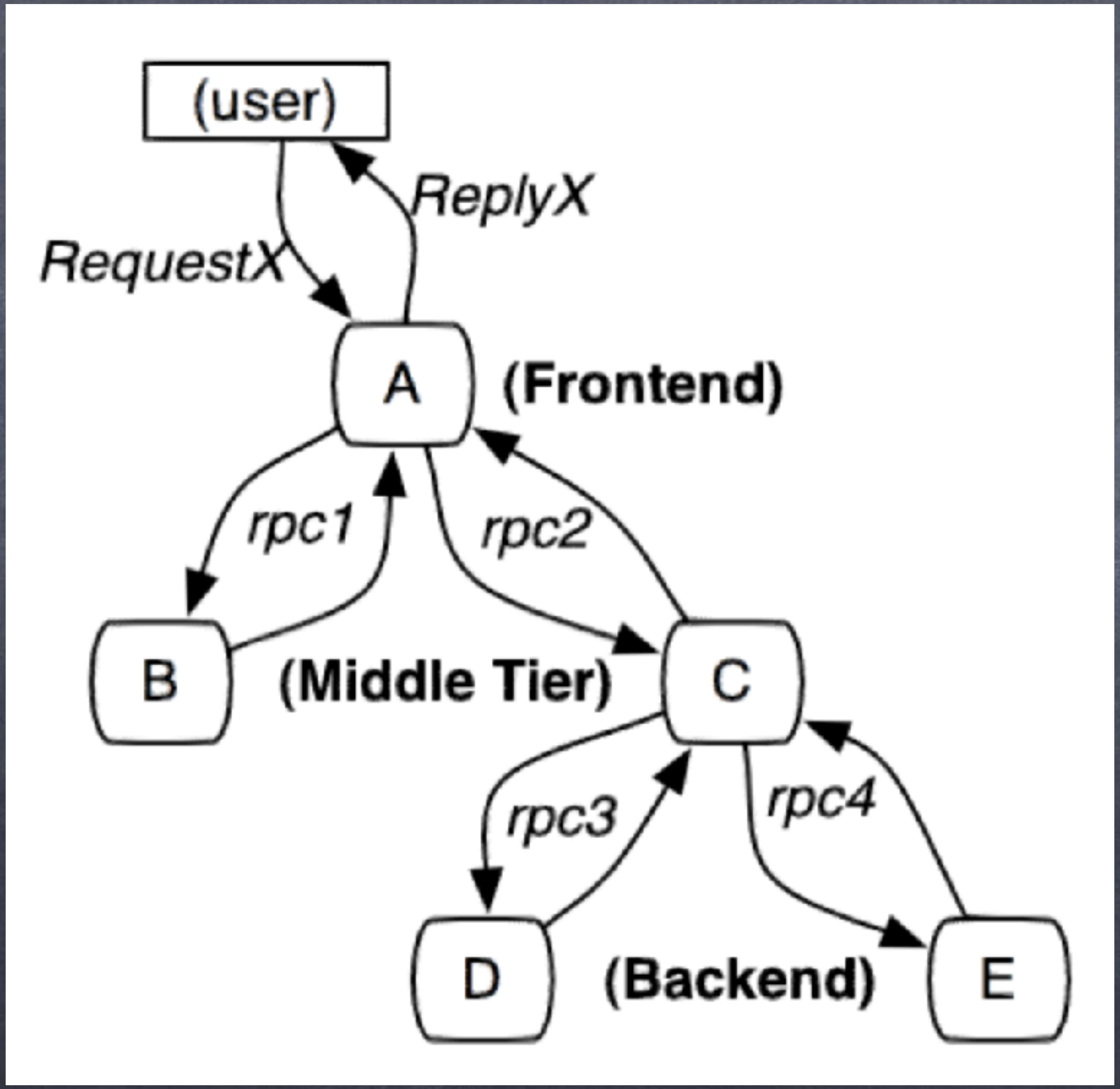
Log

- 接入配置简单
  - enniu.cloud.service.logstash.mode=tcp
  - enniu.cloud.service.logstash.remote.host=10.0.40.157
  - enniu.cloud.service.logstash.remote.port=<端口>
  - enniu.cloud.service.logs.level=INFO
- Kibana日志查看
  - 全文检索
  - 按字段查询，支持Lucene语法 service:"userservice" AND level:/ERROR|WARN/
  - 过滤选定字段
- http://wx-kibana.u51.com/app/kibana

Base Server - 监控计数

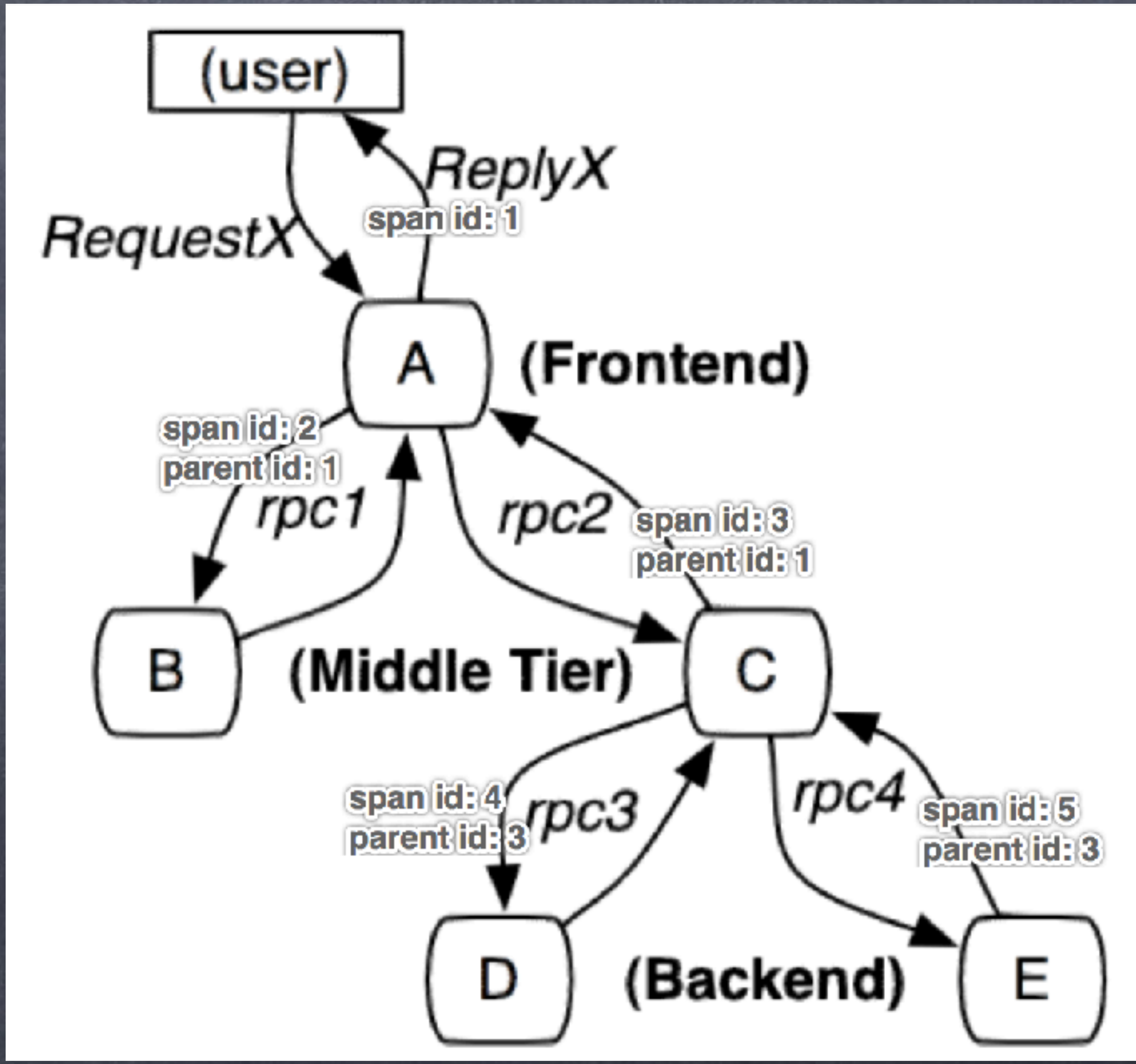
- Metrics是微服务监控和告警的数据来源，系统指标、业务指标
- 常见方案，分析nginx日志得出QPS,平均响应时间通过Dashboard展现
- 集成Prometheus，时序数据库，pull模式拉取数据
- 内置/metrics endpoint查看目前业务暴露出的各项指标项和值
- How to code
  - 定义、注册metrics
  - 定义切面拦截被监控方法
  - prometheusMetricsService.getOrRegisterCounter(...).inc()

Base Server - 链路跟踪



Base Server - 链路跟踪

- 选型
  - Zikpin, twitter开源, spring-cloud-sleuth封装友好
- 目的
  - 快速定位问题，寻找性能瓶颈，复杂服务依赖关系整理...
- 关键概念
  - traceId, span (spanId, parentId), 采样率
- X-B3-TraceId的传递: TraceFilter根据Header中的span信息创建新的span
- DAO aspect span: com.enniu.cloud.services..*DAO.*(..)
- 线上环境：http://fdzipkin.51.nb/   测试环境：http://zipkin.51.nb:9411/

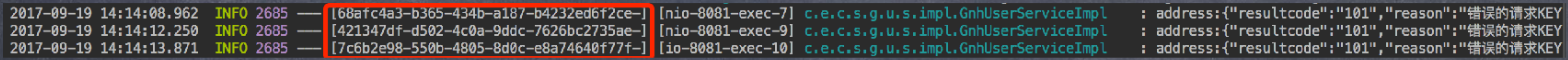


Base Server - 链路跟踪

- Pinpoint
- 阿里巴巴分布式调用跟踪和监控实践
- LTrace-链家全链路跟踪平台设计实践

Base Server - others

- Filter
  - TrackingIdFilter Ordered.HIGHEST_PRECEDENCE 可据此在日志中查询某个具体请求的日志



```
2017-09-19 14:14:08.962  INFO 2685 --- [68afc4a3-b365-434b-a187-b4232ed6f2ce-] [nio-8081-exec-7] c.e.c.s.g.u.s.impl.GnhUserServiceImpl    : address:{"resultcode":"101","reason":"错误的请求KEY
2017-09-19 14:14:12.250  INFO 2685 --- [421347df-d502-4c0a-9ddc-7626bc2735ae-] [nio-8081-exec-9] c.e.c.s.g.u.s.impl.GnhUserServiceImpl    : address:{"resultcode":"101","reason":"错误的请求KEY
2017-09-19 14:14:13.871  INFO 2685 --- [7c6b2e98-550b-4805-8d0c-e8a74640f77f-] [io-8081-exec-10] c.e.c.s.g.u.s.impl.GnhUserServiceImpl    : address:{"resultcode":"101","reason":"错误的请求KEY
```

  - TraceFilter Ordered.HIGHEST_PRECEDENCE + 5
  - BaseServiceFilter Ordered.HIGHEST_PRECEDENCE + 10 可重载以跳过token验证
- 内置endpoint: /whoami,...
- 基于consul实现的DistributeLockService
- enniu-jedis, enniu-rabbit-client

Base client - 服务发现

- consul根据service.name和service.tag发现服务
- consul.healthClient().getHealthyServiceInstances(serviceName, options)

Base client - 调用方式

```
EnniuClient enniuClient = EnniuClient.forService(serviceName, serviceTag);
this.gnhDemoApi = enniuClient.create(GnhDemoApi.class);
```

- Retrofit API -> HTTP request -> HystrixCommand
- warmup OPTIONS请求
- 请确保整个容器只有一个Service A的client，在使用时通过@AutoWired获取

Base client - 调用方式

- 调用非微服务 local-https://api.github.com
- 跨DC调用 prod@dc1, prod@dc1,dc2

```
EnniuClient enniuClient = EnniuClient.forService(serviceName, "local-https://
api.github.com");
GithubApi githubApi = enniuClient.create(GithubApi.class);
```

Base client - config

- 3种级别的配置
- default.enniuclient.http.conn.timeout.ms=3000
- payment.enniuclient.http.conn.timeout.ms=10000
- RepayfundService.getRepayfunds.enniuclient.http.conn.timeout.ms=3000

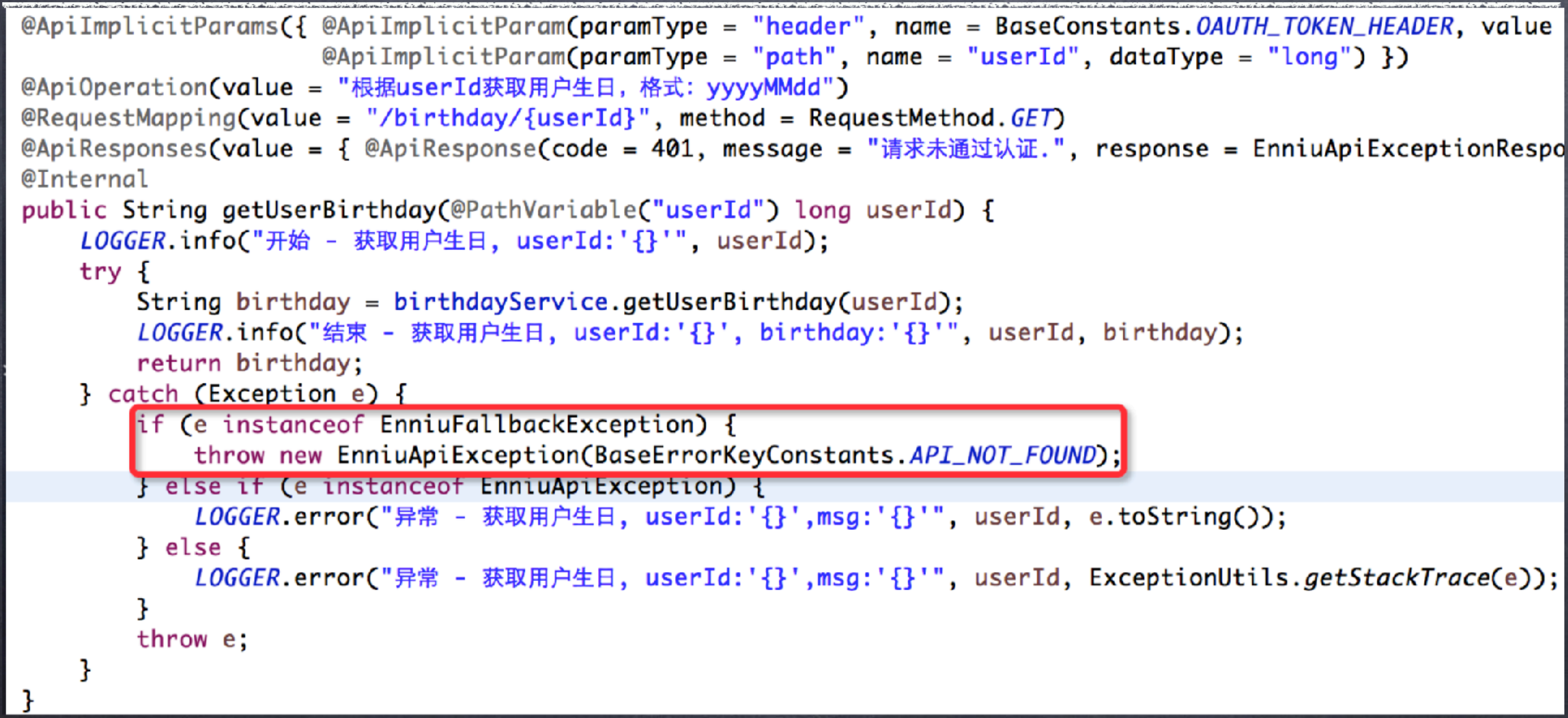
Base client - 负载均衡

- Load Balancer
    - WEIGHT 基于响应时间的权重
    - RR 轮询调度
- EnniuClient enniuClient = EnniuClient.forService(serviceName, serviceTag, loadBalancerPolicy);

Base client - 熔断保护

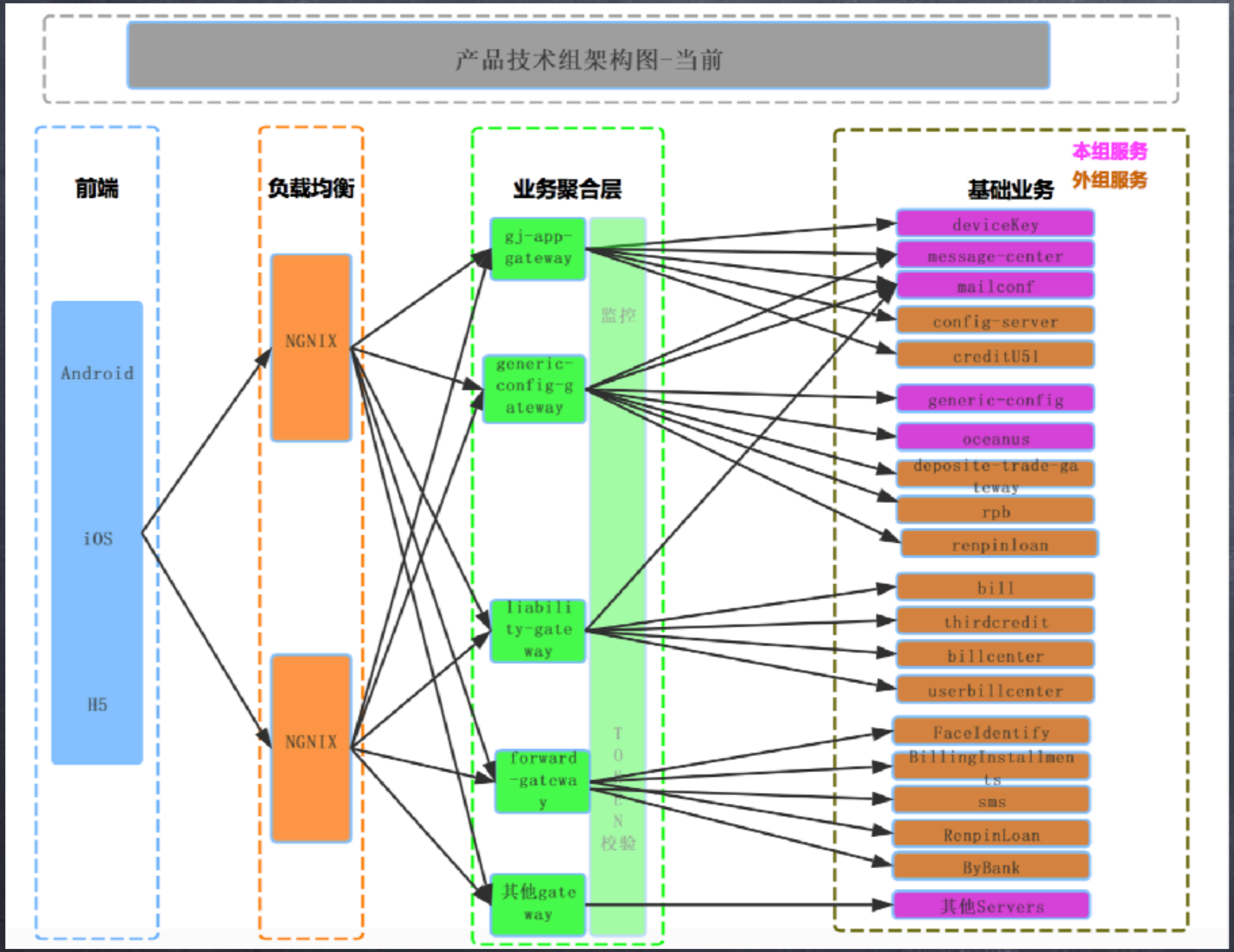
- A -> B -> C -> D, 如果服务D超时未响应，如何不连累上游服务
- Retrofit API -> HTTP request -> HystrixCommand
- EnniuFallbackException
- 熔断策略
    - hystrix.command.default.circuitBreaker.errorThresholdPercentage=50

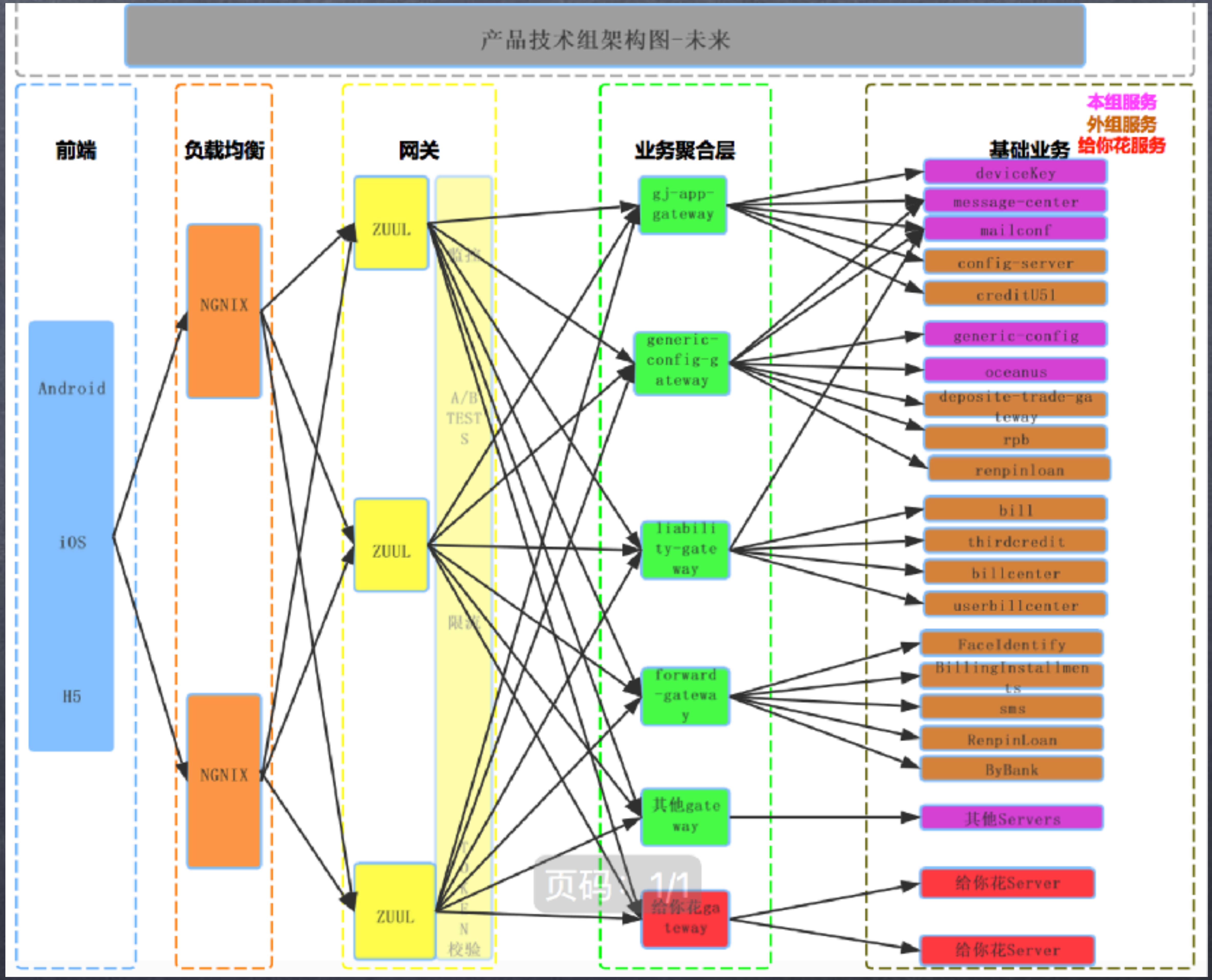
Base client - 熔断保护



```
@ApiImplicitParams({ @ApiImplicitParam(paramType = "header", name = BaseConstants.OAUTH_TOKEN_HEADER, value
                     @ApiImplicitParam(paramType = "path", name = "userId", dataType = "long") })
@ApiOperation(value = "根据userId获取用户生日，格式：yyyyMMdd")
@RequestMapping(value = "/birthday/{userId}", method = RequestMethod.GET)
@ApiResponses(value = { @ApiResponse(code = 401, message = "请求未通过认证.", response = EnniuApiExceptionRespo
@Internal
public String getUserBirthday(@PathVariable("userId") long userId) {
    LOGGER.info("开始 - 获取用户生日, userId:'{}'", userId);
    try {
        String birthday = birthdayService.getUserBirthday(userId);
        LOGGER.info("结束 - 获取用户生日, userId:'{}', birthday:'{}'", userId, birthday);
        return birthday;
    } catch (Exception e) {
        if (e instanceof EnniuFallbackException) {
            throw new EnniuApiException(BaseErrorKeyConstants.API_NOT_FOUND);
        } else if (e instanceof EnniuApiException) {
            LOGGER.error("异常 - 获取用户生日, userId:'{}',msg:'{}'", userId, e.toString());
        } else {
            LOGGER.error("异常 - 获取用户生日, userId:'{}',msg:'{}'", userId, ExceptionUtils.getStackTrace(e));
        }
        throw e;
    }
}
```

Janus


产品技术组架构图-当前
前端
Android
iOS
H5
负载均衡
NGNIX
NGNIX
业务聚合层
gj-app-gateway
generic-config-gateway
liability-gateway
forward-gateway
其他gateway
监控
TOKEN校验
基础业务
本组服务
外组服务
deviceKey
message-center
mailconf
config-server
creditU51
generic-config
oceanus
deposite-trade-gateway
rpb
renpinloan
bill
thirdcredit
billcenter
userbillcenter
FaceIdentify
BillingInstallments
sms
RenpinLoan
ByBank
其他Servers


产品技术组架构图-未来
前端
Android
iOS
H5
负载均衡
NGNIX
NGNIX
网关
ZUUL
ZUUL
ZUUL
A/B TESTS
TOKEN校验
业务聚合层
gj-app-gateway
generic-config-gateway
liability-gateway
forward-gateway
其他gateway
给你花gateway
基础业务
本组服务
外组服务
给你花服务
deviceKey
message-center
mailconf
config-server
creditU51
generic-config
oceanus
deposite-trade-gateway
rpb
renpinloan
bill
thirdcredit
billcenter
userbillcenter
FaceIdentify
BillingInstallments
sms
RenpinLoan
ByBank
其他Servers
给你花Server
给你花Server

Janus

- 统一网关充当US1应用的服务端所有请求的前门，提供如动态路由、监控、弹性、限流、安全等
- 主要功能
  - 平滑上下线：保证不丢失请求
  - 动态扩缩容：无需reload nginx
  - 精细灵活的流量分发与限流（HTTP method, URL, 参数等）
  - …
- http://wiki.51.nb/pages/viewpage.action?pageId=37333076

参考资料

- http://git.51.nb/service/documentation/wikis/home
- http://wiki.51.nb/pages/viewpage.action?pageId=40442307&preview=%2F40442307%2F40442590%2Fbase%E6%A1%86%E6%9E%B6%E6%95%B4%E4%BD%93%E4%BB%8B%E7%BB%8D.pdf
- http://wiki.51.nb/pages/viewpage.action?pageId=39788748
- http://wiki.51.nb/pages/viewpage.action?pageId=34814209
- http://wiki.51.nb/pages/viewpage.action?pageId=34814734
- http://blog.spring-cloud.io/blog/sc-sleuth.html