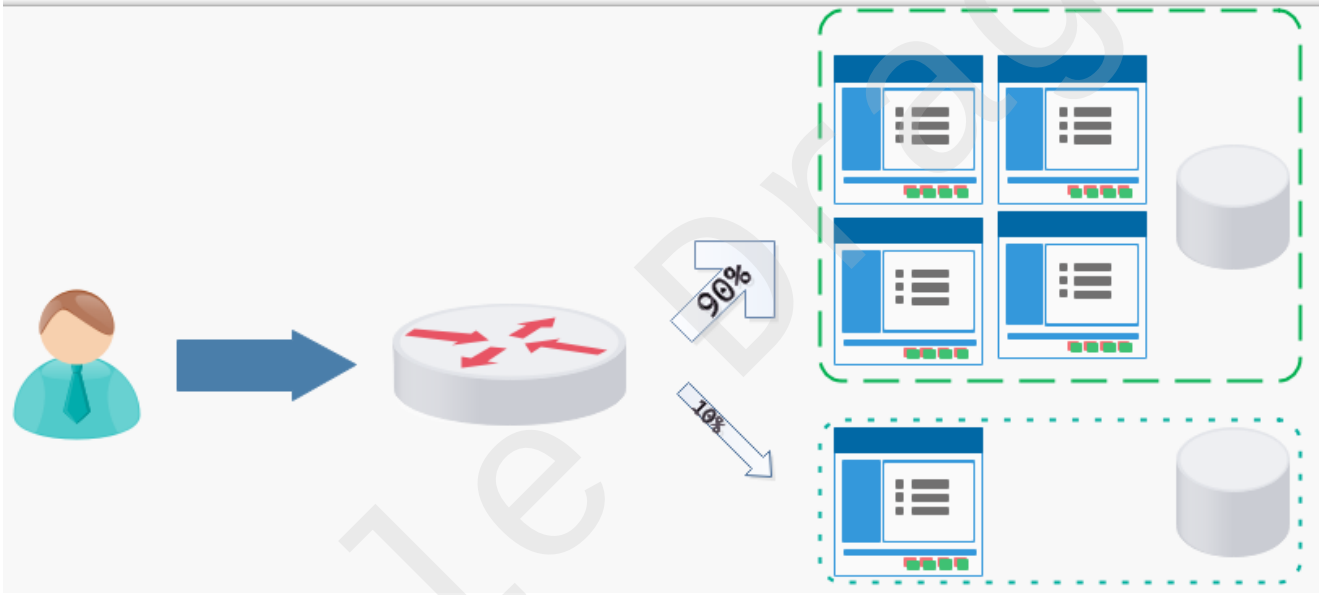
什么是灰度发布

灰度发布又名金丝雀发布，

> 是指在黑与白之间，能够平滑过渡的一种发布方式。 在其上可以进行A/B testing，即让一部分用户继续用产品特性A，一部分用户开始用产品特性B，如果用户对B没有什么反对意见，那么逐步扩大范围，把所有用户都迁移到B上面来。–维基百科

假定线上版本为A, 需要发布的版本为B，通常是新版本。
灰度发布可以让一定百分比的用户(比如10%)优先体验版本B，其余用户仍然使用版本A，并且慢慢扩大百分比，最终将所用用户迁移到版本B。



为什么需要灰度发布

随着微服务架构的普及，服务数量激增，版本更新频繁，如果缺少灰度的能力，容易对现有的生产运行业务造成影响，并且新上线的系统和功能也需要灰度的能力来验证可行性和效果，简而言之，无论是对于系统运行稳定安全还是对于验证业务效果，灰度发布/验证的能力都是现代软件架构所必须的。

灰度发布有一个隐藏的前提条件，线上资源远大于生产团队的规模；
以至于生产团队没有足够的资源来应付使用新版本后带来的问题，包括用户不习惯和数据量所带来的不适应和线上bug等等。

而灰度发布可以通过控制发布面积，让使用新版本的用户控制在生产团队可以应付的范围内；将可能出现问题的用户圈定在“部分”用户，这个部分的范围由生产团队评估得出。

我们通过灰度发布，将一部分经过筛选后的用户纳入测试范围，由生产团队和用户共同完成对新版本的验收；主要的目的是检验了用户对于新版本的接受度，规避了因为产品决策冒失、冒进所带来的风险。

问题

用户标识

用于区分用户，辅助数据统计，并且保证灰度发布过程中用户体验的连贯性(不要在新老版本中跳来跳去)。

目标用户的选取策略

如何选取让哪些用户先行体验新版本，是让用户自己选择还是强制升级等。

数据

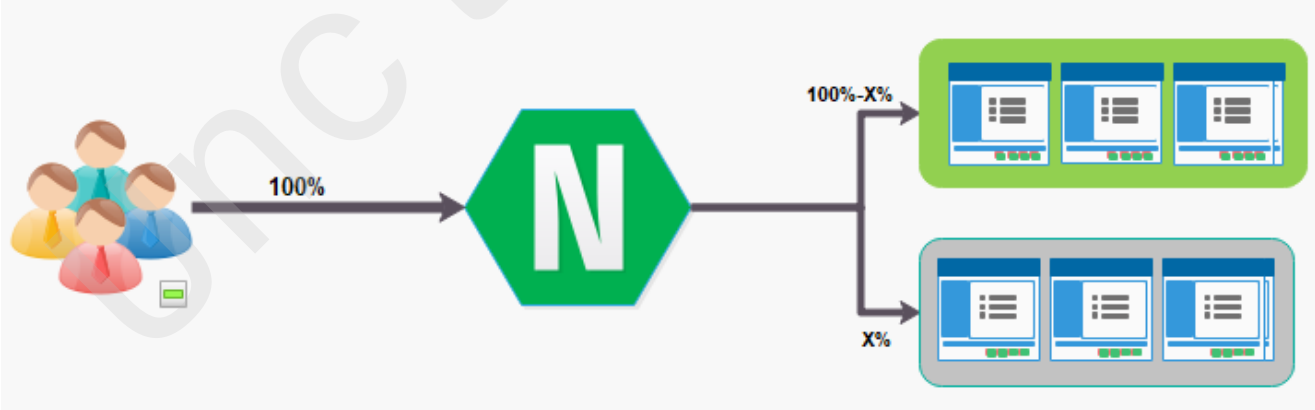
如何能够获取新版本的指标数据

回滚策略

新版本表现不佳，如何快速回滚到老版本。如果是app 回滚可能代价比较大。

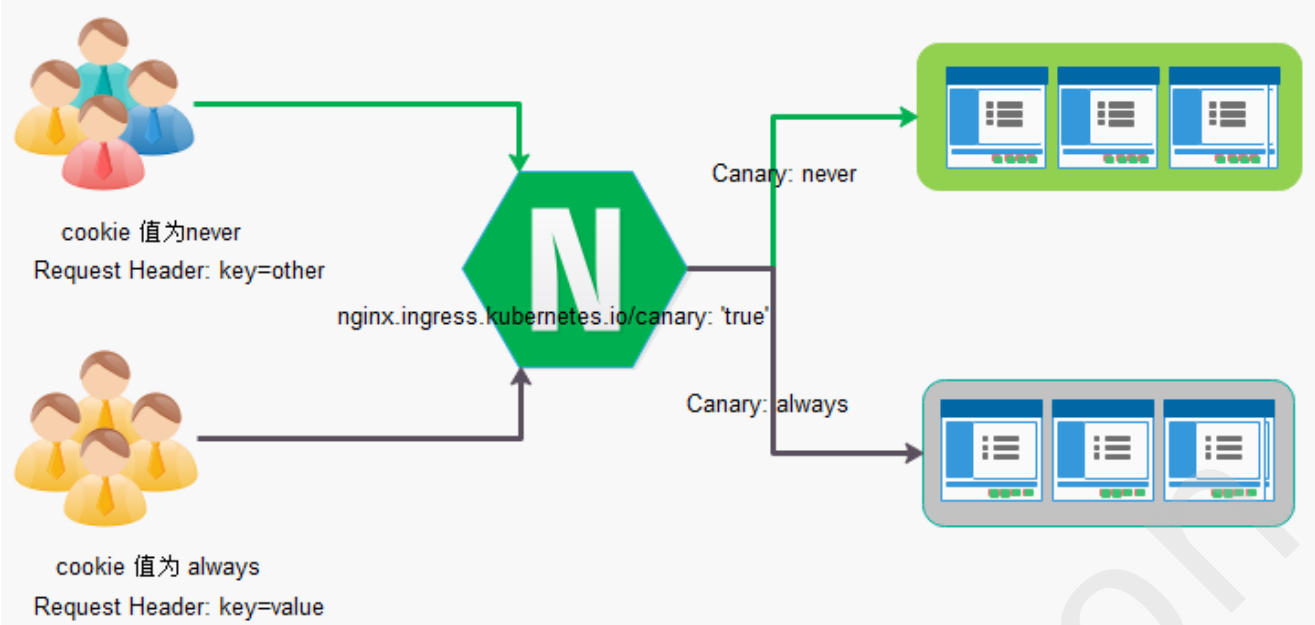
灰度策略

按流量比例



基于流量就是设置流量权重，实现上最简单，但是流量不稳定，会使用户使用有割裂感。

按请求内容



这个分为两种：基于 cookie 和 请求 Header;
这种相对于前面一种就会相对复杂点，需要请求端设置 Header或cookie。这种用户体验会好很多，能够保证用户体验的连贯性。

> 注意：金丝雀规则具有以下优先级：

> `canary-by-header` - > `canary-by-cookie` - > `canary-weight`

灰度发布实践

我们商用环境k8s 管理是使用的 `KubeSphere` ，所以下面的实践都是利用 `KubeSphere` 来实现

简单自定义一个接口服务：

```
const main = async function(ctx) {

 ctx.body ="this is canary test";;

};



const about = ctx => {

 ctx.body = "this is version 2";
 // ctx.body = "this is version 1";
```

```
};
```

分别构建了两个镜像：

```
# Production 版本
repository.anxinyun.cn/base-images/demo:0.1

# Canary 版本
repository.anxinyun.cn/base-images/demo:0.3
```

基于 `ingress-nginx`

1. 我们先部署一个 `Production` 版本的应用

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: gated-demo
  namespace: fs-gtest
spec:
  replicas: 1
  selector:
    matchLabels:
      app: gated-demo
  template:
    metadata:
      labels:
        app: gated-demo
    spec:
      containers:
      - name: gated-demo
        image: repository.anxinyun.cn/base-images/demo:0.1
        ports:
        - containerPort: 5000
          name: http-port

---

apiVersion: v1
kind: Service
metadata:
```

```
  name: gated-demo
  namespace: fs-gtest
  labels:
    app: gated-demo
spec:
  ports:
  - port: 15001
    targetPort: 5000
    protocol: TCP
    name: http-port
  selector:
    app: gated-demo
```

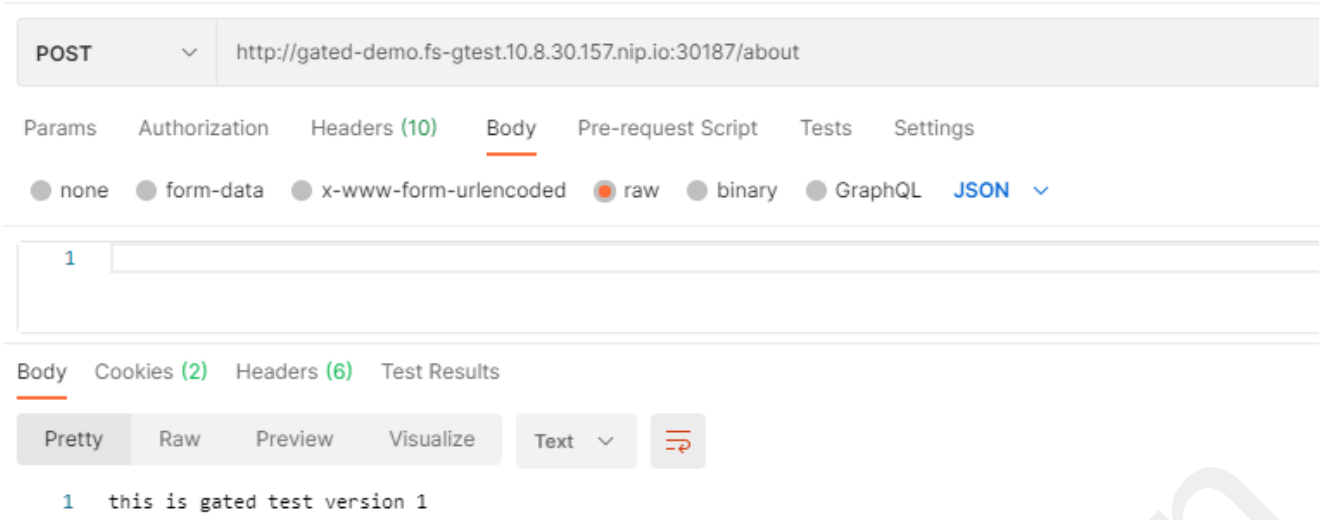
```
$ kubectl appy -f production.yaml

deployment.apps/production created
service/production created
```

2. 创建 `Production` 版本的应用路由 (Ingress)

```
kind: Ingress
apiVersion: extensions/v1beta1
metadata:
  name: gated-demo
  namespace: fs-gtest
  annotations:
    kubernetes.io/ingress.class: nginx
    kubesphere.io/creator: admin
spec:
  rules:
    - host: gated-demo.fs-gtest.10.8.30.157.nip.io
      http:
        paths:
          - path: /
            pathType: ImplementationSpecific
            backend:
              serviceName: gated-demo
              servicePort: 15001
```

3. 访问`Production` 接口



4. 下面部署 `Canary` 版本

yaml 文件 参考上面的版本，在 `gated-demo` 加上 `-canary`

5. 设置 Ingress-Nginx Annotation 规则

> 要开启灰度发布机制，需设置 `nginx.ingress.kubernetes.io/canary: "true"` 启用 Canary

基于权重

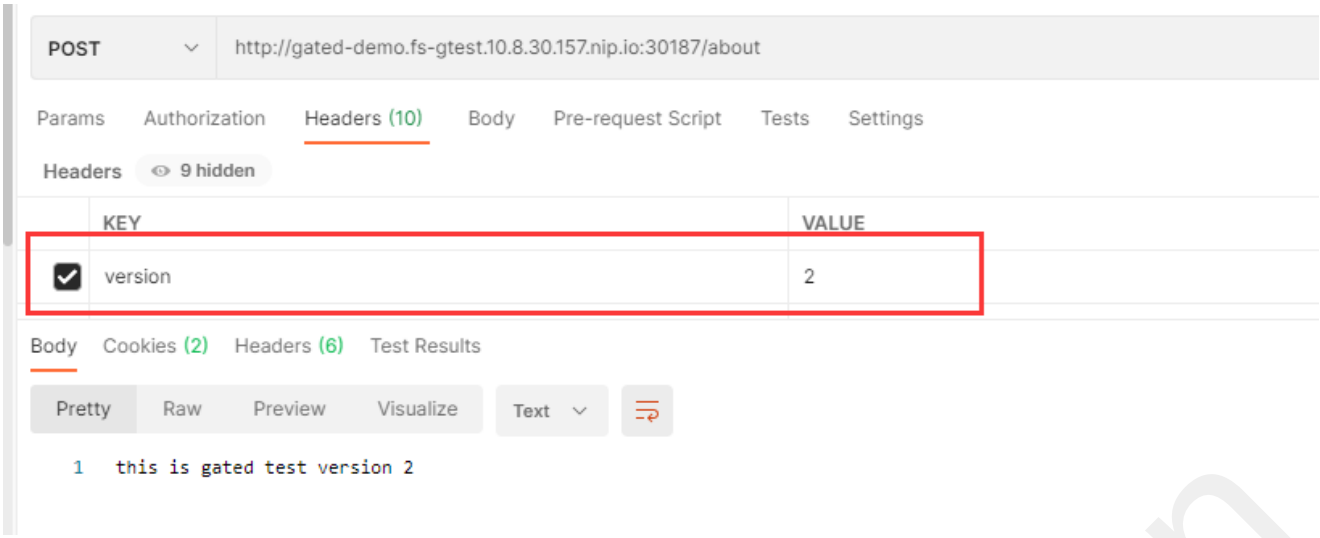
```
kind: Ingress
apiVersion: extensions/v1beta1
metadata:
  name: gated-demo-canary
  namespace: fs-gtest
  annotations:
    kubernetes.io/ingress.class: nginx
    kubesphere.io/creator: admin
    nginx.ingress.kubernetes.io/canary: 'true'
    nginx.ingress.kubernetes.io/canary-weight: "30"
spec:
  rules:
    - host: gated-demo.fs-gtest.10.8.30.157.nip.io
      http:
        paths:
          - path: /
            pathType: ImplementationSpecific
            backend:
```

```
              serviceName: gated-demo-canary
              servicePort: 15002
```

> 上面 Ingress 示例的 Canary 版本使用了**基于权重进行流量切分**的 annotation 规则，将分配 **30%** 的流量请求发送至 Canary 版本

Request Header

```
kind: Ingress
apiVersion: extensions/v1beta1
metadata:
  name: gated-demo-canary
  namespace: fs-gtest
  annotations:
    kubernetes.io/ingress.class: nginx
    kubesphere.io/creator: admin
    nginx.ingress.kubernetes.io/canary: 'true'
    nginx.ingress.kubernetes.io/canary-by-header: version
    nginx.ingress.kubernetes.io/canary-by-header-value: '2'
spec:
  rules:
    - host: gated-demo.fs-gtest.10.8.30.157.nip.io
      http:
        paths:
          - path: /
            pathType: ImplementationSpecific
            backend:
              serviceName: gated-demo-canary
              servicePort: 15002
```



当 header 'version' 设置 为 '2' 时，所有请求被转到 Canary 版本

总结

上面是简单的灰度测试，实际商用还是考虑灰度策略问题如何与用户结合起来控制，可以指定哪些用户参与灰度测试。