VUE SSR 性能优化实践

微医-云服务中心　齐云雷

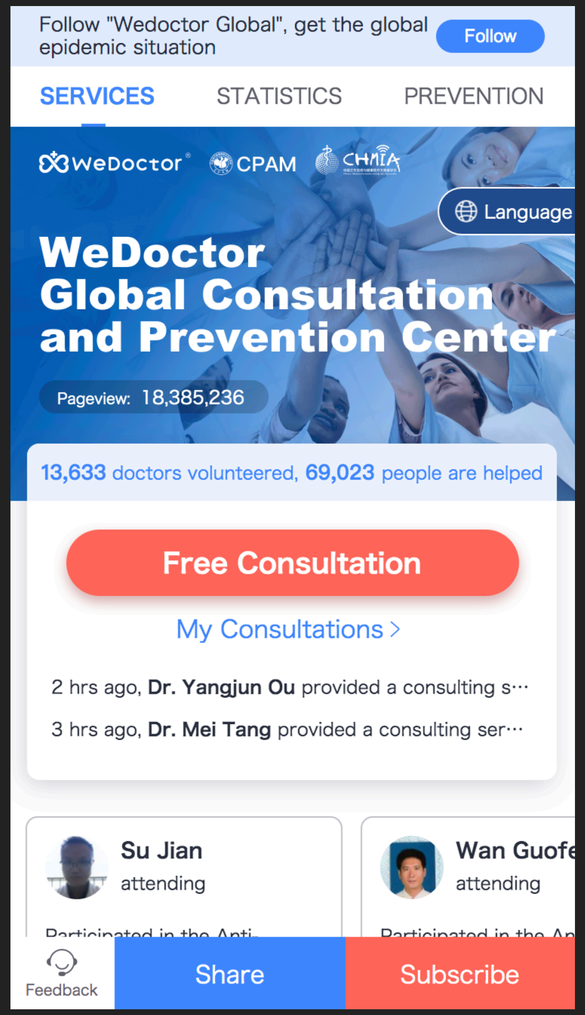
目录

一、实践背景

背景

全国战疫

200万例免费问诊



全球战疫

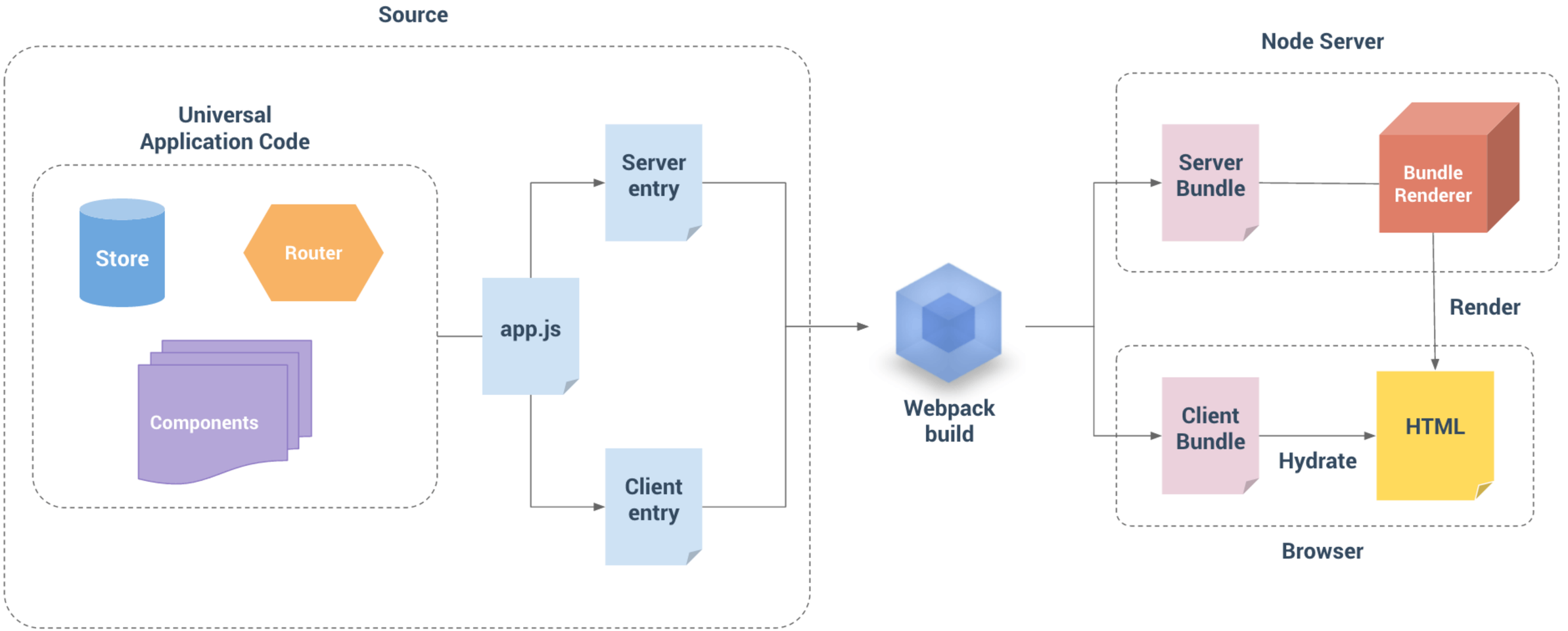
160家驻外使领馆推介

业务背景 - 抗疫

承接更大的流量

服务更远的用户

技术背景 - SSR


Source
Universal
Application Code
Store
Router
Components
app.js
Server
entry
Client
entry
Webpack
build
Node Server
Server
Bundle
Bundle
Renderer
Render
Client
Bundle
Hydrate
HTML
Browser

SSR 优势

减少首页白屏

SEO 友好

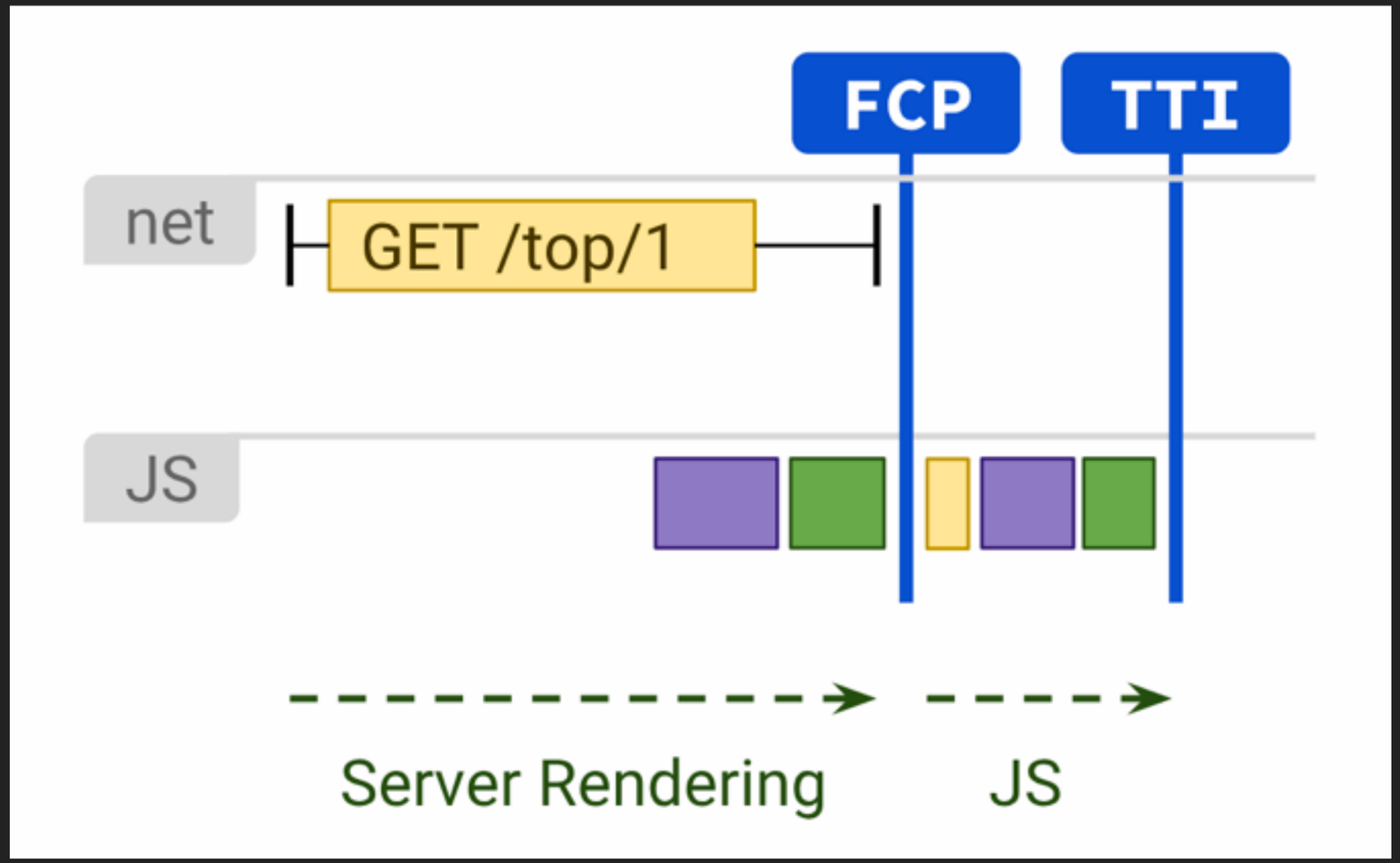
SSR 缺陷

服务端压力

开发体验

二、优化方案

渲染前　　渲染中　　渲染后



渲染中

VDOM -> 字符串拼接

- 重写 vue-loader

- VUE 3.0

（一） 常规优化

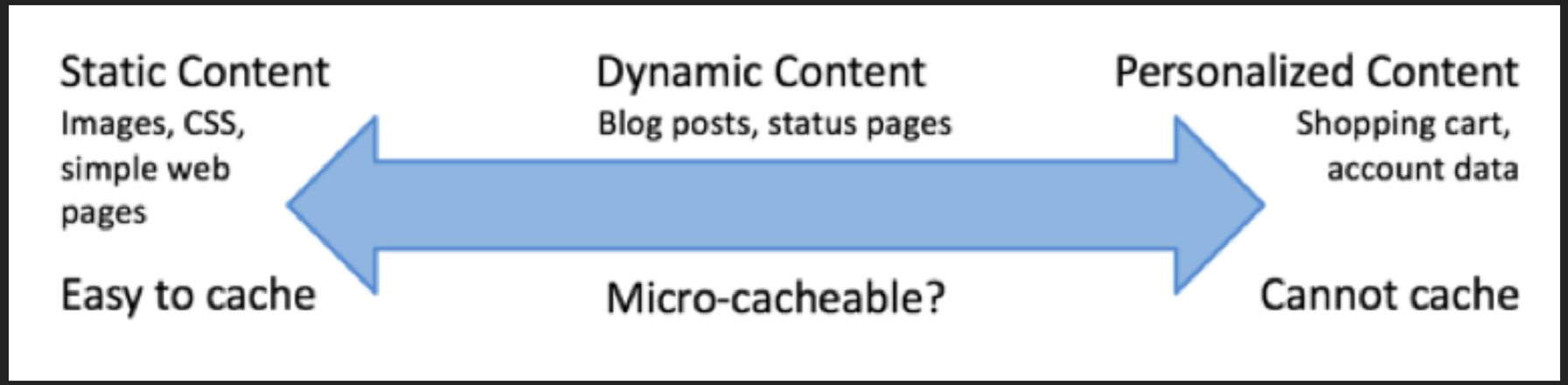
渲染前

缓存：数据、组件、页面

请求：http keep-alive

降级：客户端渲染

页面缓存不能解决所有问题



渲染后

CDN 加速静态资源

压缩响应体

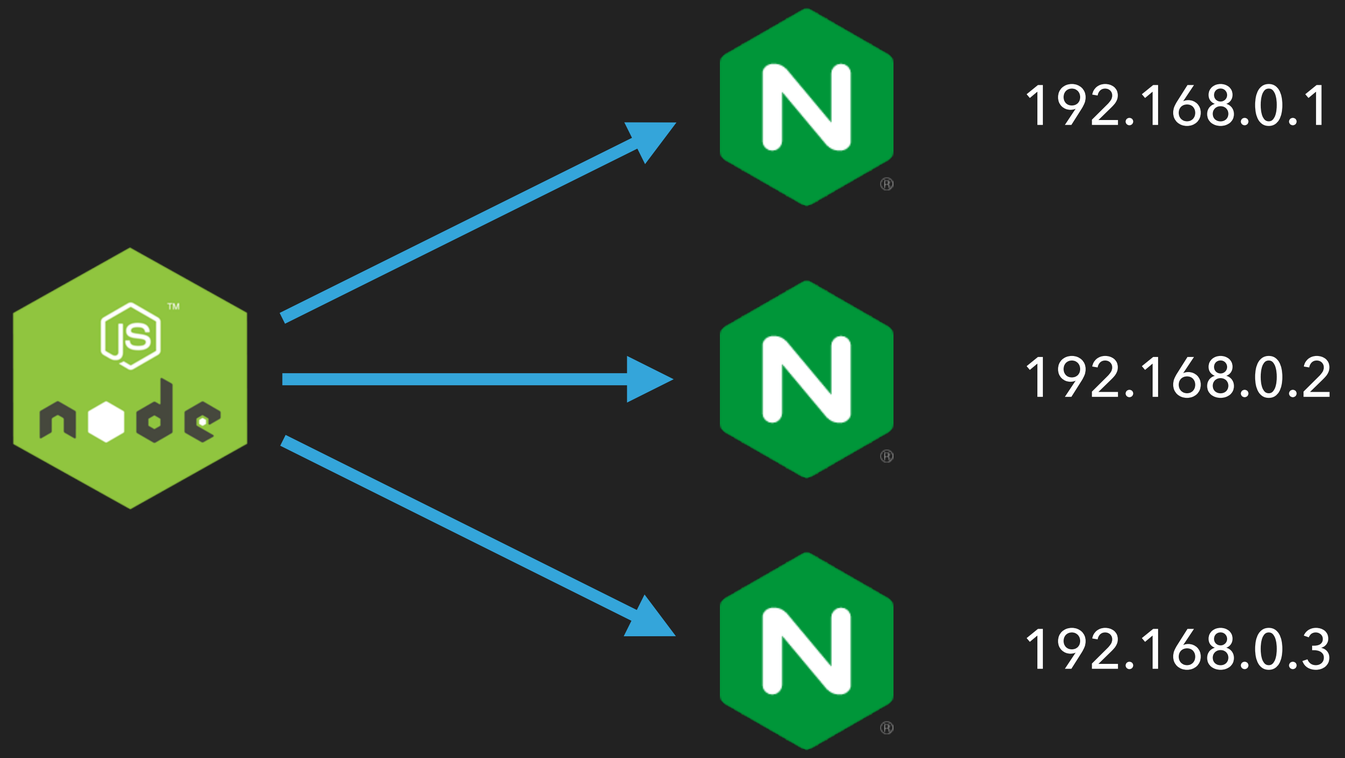
（二） 深度实践

渲染前——基础网络调优

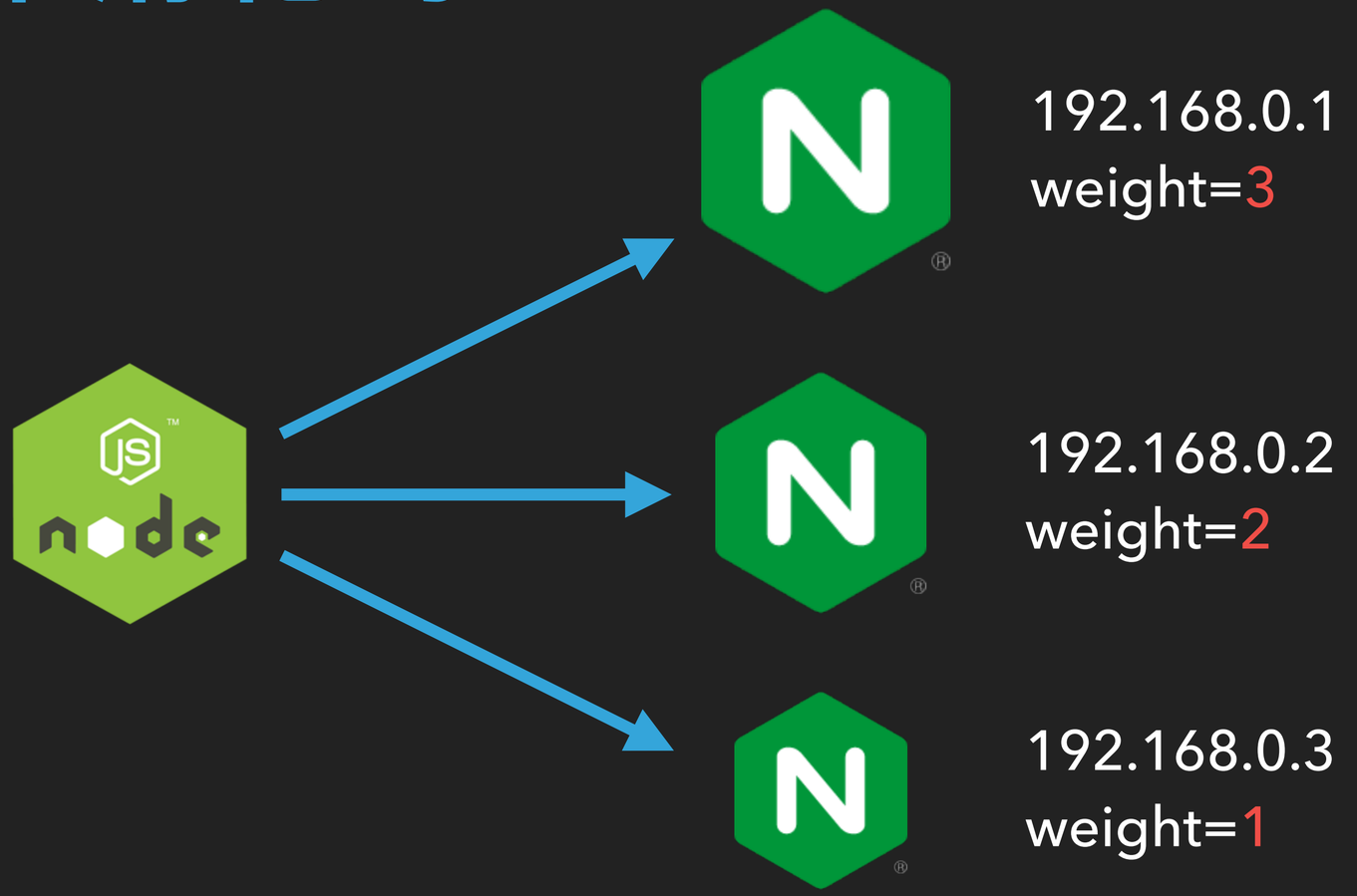
网关域名 -> 网关 IP(HTTPS) -> 内网域名(HTTP)

250 ms 90 ms 10 ms

IP 负载均衡与容灾



加权平滑轮询

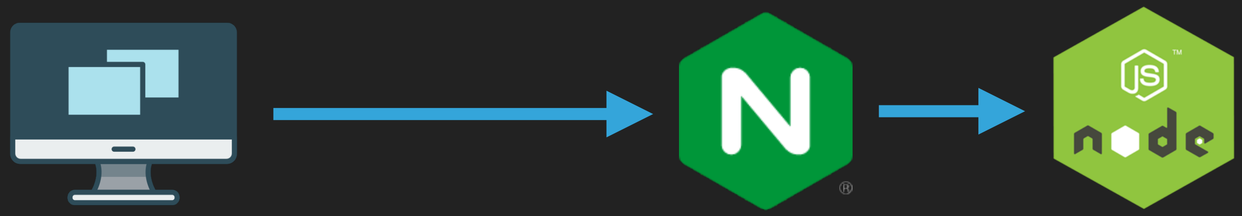


渲染后——扩展多级缓存

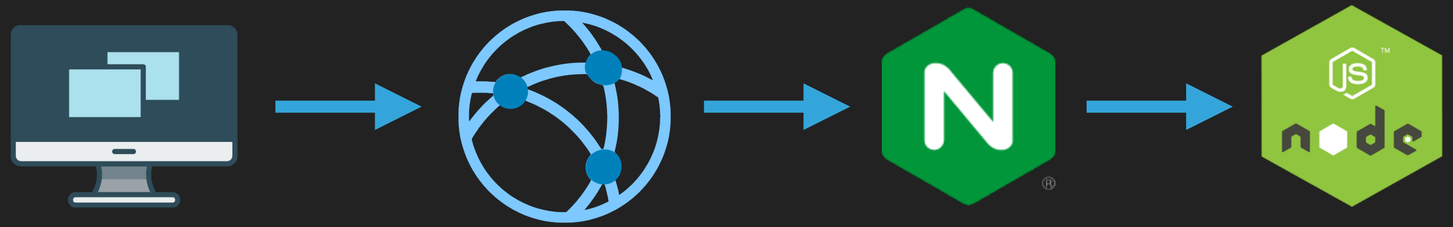
(1) CDN 缓存介绍

为什么接入 CDN？

接入前



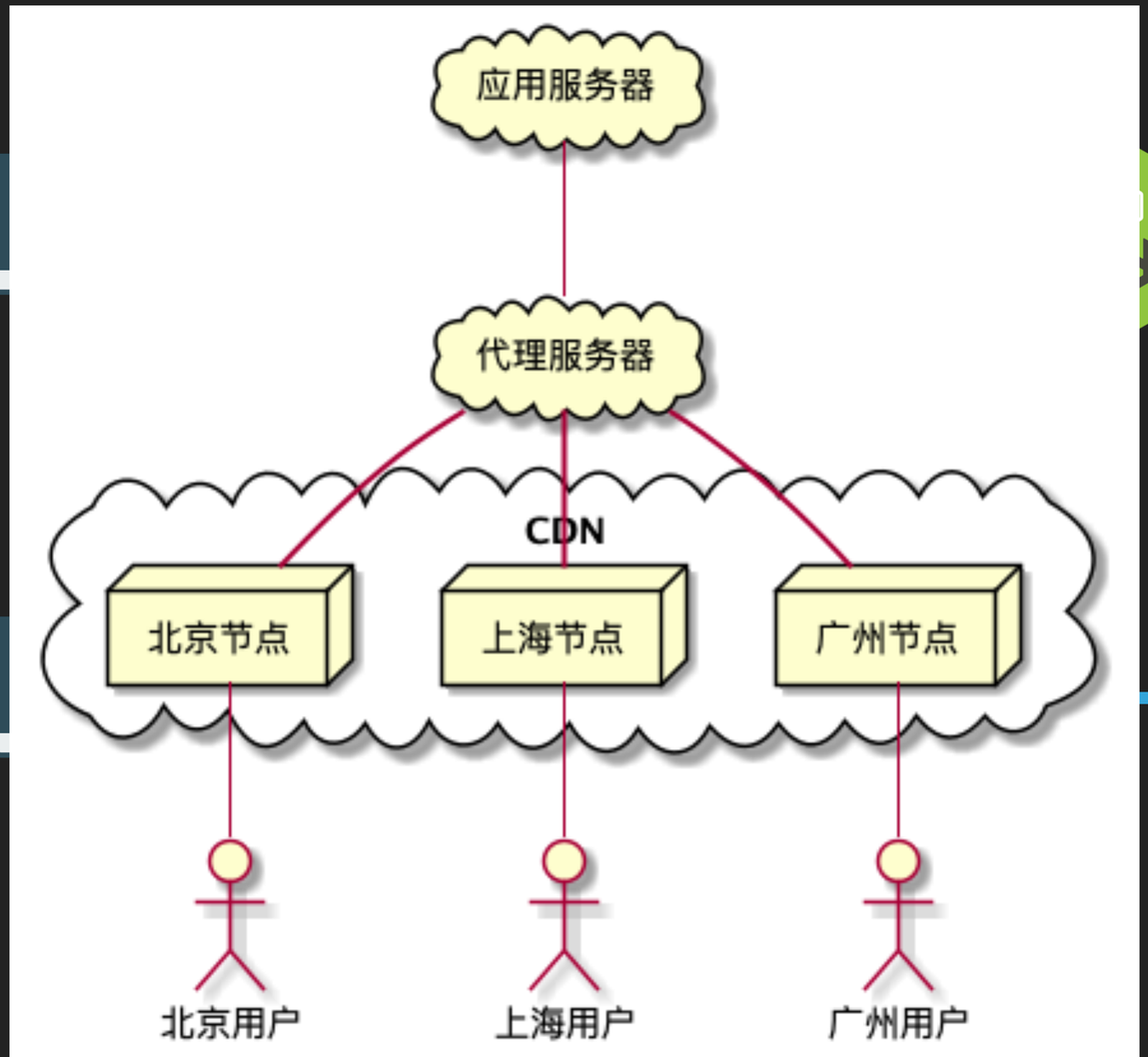
接入后



为什么接入 CDN？

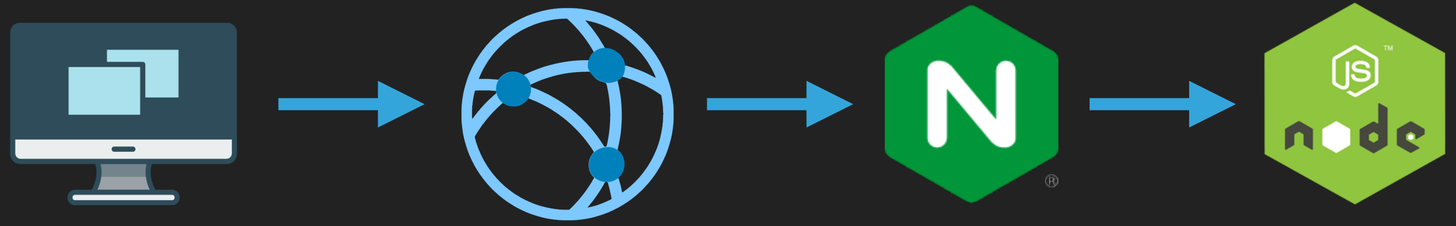
接入前

接入后

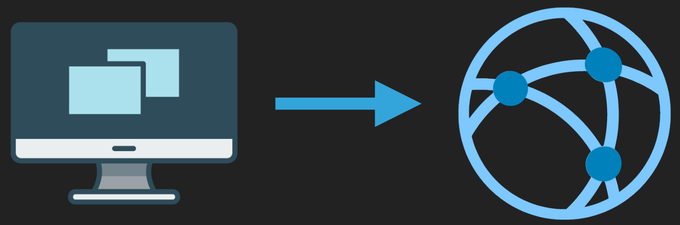


为什么开启 CDN 缓存?

开启前



开启后



如何开启 CDN 缓存?

- 域名接入 CDN 服务，并启用缓存
- 源站响应头设置 Cache-Control

什么服务可以开启 CDN 缓存?

- 无用户状态

- 低时效性

缓存优化

- 缓存时间
- 忽略参数
- 主动缓存

(2) 应用代码演进

掌控缓存

```
app.use((ctx, next) => {
  ctx.set('Cache-Control', 'max-age=300')
})
```

控制路径

```
app.use((ctx, next) => {
  if (ctx.path === '/foo') {
    ctx.set('Cache-Control', 'max-age=300')
  }
})
```

补充路径

```
app.use((ctx, next) => {
  if ([ '/foo', '/foo/' ].includes(ctx.path)) {
    ctx.set('Cache-Control', 'max-age=300')
  }
})
```

忽略降级页面

```
app.use(async (ctx, next) => {
  if ([ '/foo', '/foo/' ].includes(ctx.path)) {
    ctx.set('Cache-Control', 'max-age=300')
  }

  await next()

  // 页面降级时，取消缓存
  if (ctx._degrade) {
    ctx.set('Cache-Control', 'no-cache')
  }
})
```

Cookie 和状态治理

```
app.use(async (ctx, next) => {
  const enableCache = [ '/foo', '/foo/' ].includes(ctx.path)

  if (enableCache) {
    ctx.set('Cache-Control', 'max-age=300')
  }

  await next()

  // 页面降级时，取消缓存
  if (ctx._degrade) {
    ctx.set('Cache-Control', 'no-cache')
  }
  // 缓存页面不设 Set-Cookie
  else if (enableCache) {
    ctx.res.removeHeader('Set-Cookie')
  }
})
```

未解决的需求

- 只缓存 /user/:id，而不是缓存 /user/*
- 不同路径不同缓存时间

定制缓存路径

```
// src/router/example.js

import NotFound from '@/views/404/index.vue'

export default [
  {
    path: '/notfound',
    name: 'not-found-404',
    component: NotFound,
    meta: {
      // 页面级缓存
      cacheControlMaxAge: 604800,
      domain: {
        useWhiteList: true
      }
    }
  }
]
```

缓存失效

- 缓存刷新时机

- 应对缓存穿透

(3) 页面静态化

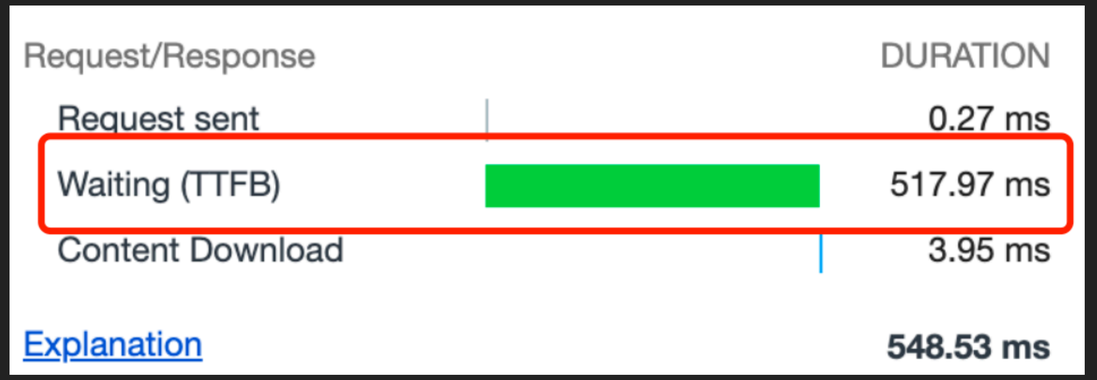
动态预渲染

- 定时主动式

- 被动触发式

三、总结

首字节时间变化


Request/Response
DURATION
Request sent
0.27 ms
Waiting (TTFB)
517.97 ms
Content Download
3.95 ms
Explanation
548.53 ms


Request/Response
DURATION
Request sent
0.15 ms
Waiting (TTFB)
18.33 ms
Content Download
2.56 ms
Explanation
24.48 ms

CDN 缓存预备步骤

1. 页面无状态改造
2. 清除 set-cookie
3. 待缓存页面设置 Cache-Control
4. 全局 no-store / no-cache

挑战

1. 路径控制
2. 页面降级
3. 状态治理
4. 缓存失效

Thanks!