刘洋河·第二场

浅谈流量劫持与防治

刘洋河 美团点评高级前端工程师

本次分享主要和大家聊一下Web中的三种常见流量劫持，以及如何去防治它们。

1. 典型的上网会经历哪些阶段
2. DNS 投毒与防治、HTTP (S) 流量劫持与防治

浅谈流量劫持与防治

美团金融 刘洋河

美团点评

自我介绍

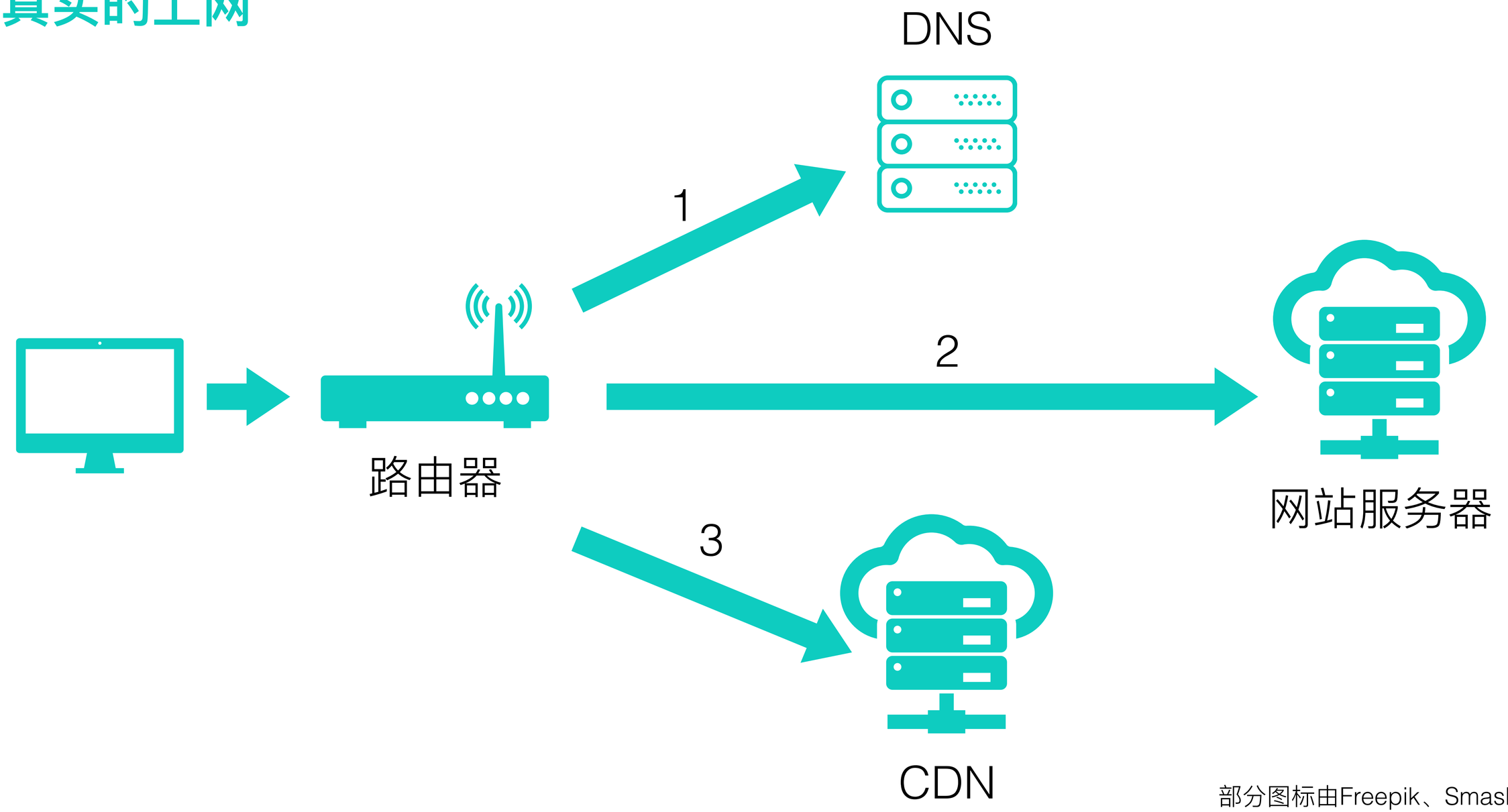
刘洋河
美团金融 高级前端工程师

2014年 开始前端职业生涯
2017年 加入美团

目录

理想的上网

真实的上网



部分图标由Freepik、Smashicons绘制

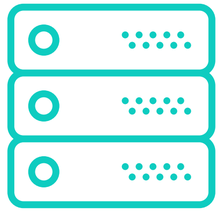
流量劫持

- 链路本身不安全
  - 从设计上未考虑安全性
  - 随着计算力发展，安全链路变得不安全
- 干扰安全链路，迫使链路使用弱安全方案

DNS 是如何工作的

用户设备

- 浏览器可能会缓存域名解析
- 用户系统中可以有自己的域名映射表

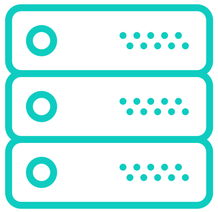


公共域名服务器

- 通常由ISP提供
- 缓存上一级域名服务器的结果

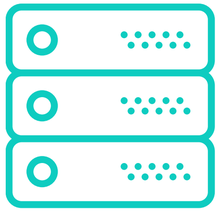
www.meituan.com的IP地址是?

DNS 是如何工作的



公共域名服务器

TTL到期



顶级域名服务器

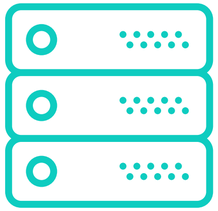
- DNS 级联的特性决定了中途可以有更多域名服务器

- 由顶级域名经营机构维护
- 可细分为与国家、通用

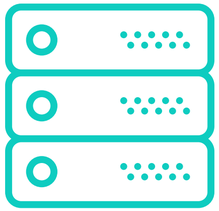
我也不知道，

先问问com：meituan.com?

DNS 是如何工作的



公共域名服务器

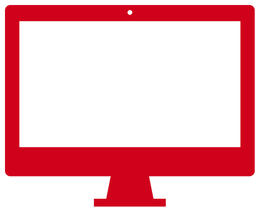


权威域名服务器

- 通常由专业的域名服务机构提供
- 购买域名时一般会提供

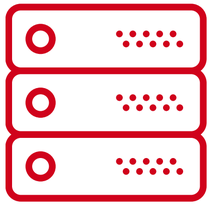
所以www.meituan.com的IP是?

如何污染 DNS



- 篡改 hosts 文件
- 拦截 DNS 请求

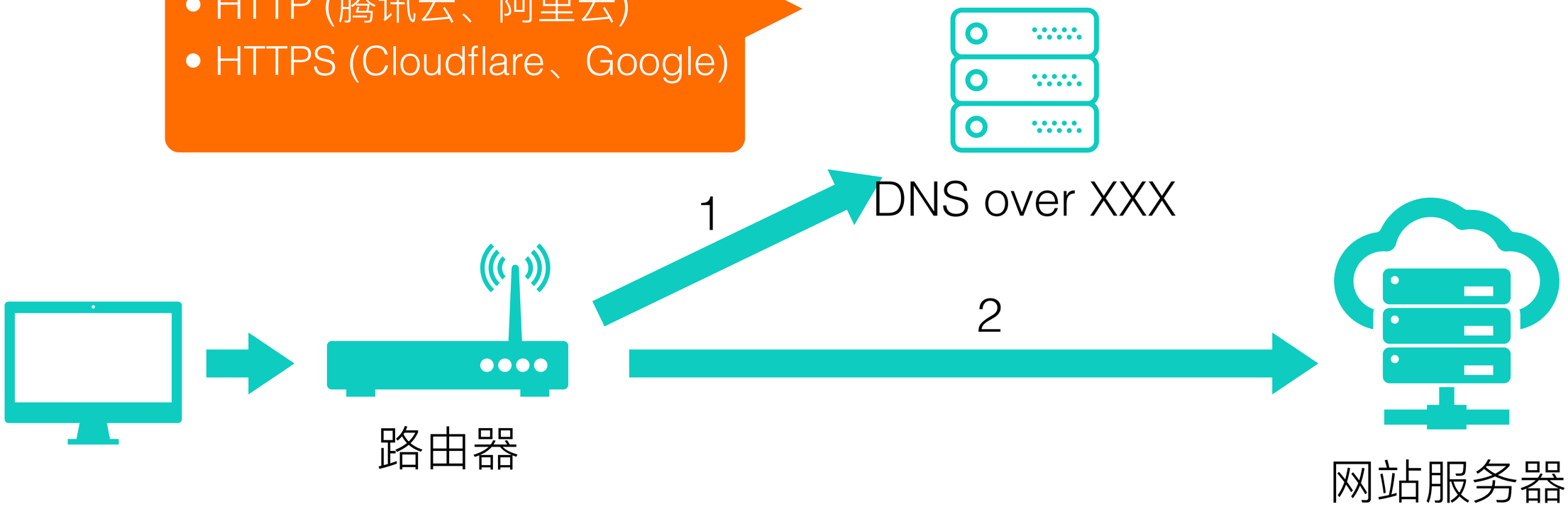
- 污染链路设备
- 中间人攻击



- 利用 DNS 服务漏洞
- 污染上一级 DNS

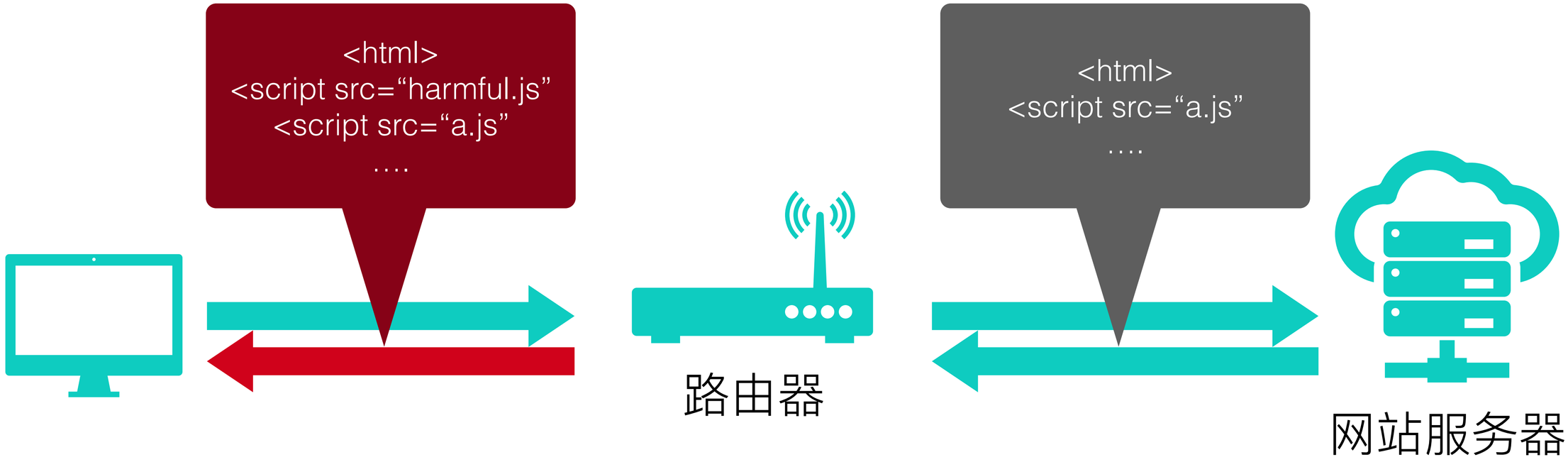
如何抵御 DNS 投毒

- TLS (Cloudflare)
- HTTP (腾讯云、阿里云)
- HTTPS (Cloudflare、Google)



很遗憾 Web 无法直接使用

HTTP 流量劫持


<html>
<script src="harmful.js"
<script src="a.js"
....
路由器
<html>
<script src="a.js"
....
网站服务器

Content Security Policy

Content-Security-Policy:
directive: rules;

default-src 'none';
script-src 'self';
connect-src 'self';
img-src 'self';
style-src 'self';

作用：

- 指定每种资源类型可以加载执行的条件
- 主要用于防御XSS攻击
- 也可以用于强迫资源使用 https 加载

缺点：

- 用于 http 页面时无法抵抗中间人攻击
- 规则比较复杂
- 影响动态创建脚本的使用

Subresource Integrity

```
<script crossorigin="anonymous"
    integrity=“sha256-+Ec97…E=“
    src=“https://a.com"></script>
```

作用：

- 只执行匹配相应hash的资源

缺点：

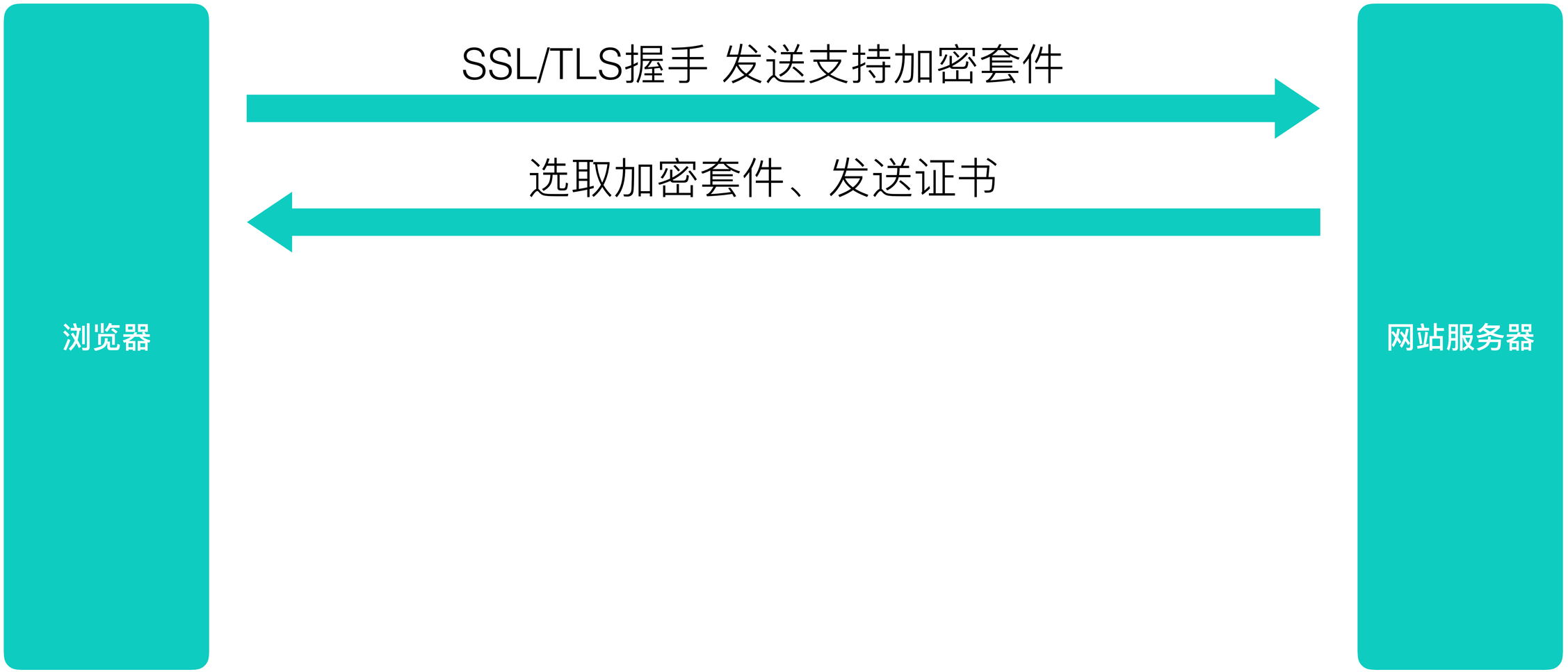
- 用于 http 页面时无法抵抗中间人攻击
- 影响动态创建脚本的使用
- 校验失败时影响可用性
- 兼容性有限，iOS Safari不支持

走向 HTTPS

SSL/TLS 上的http

HTTPS 是如何工作的

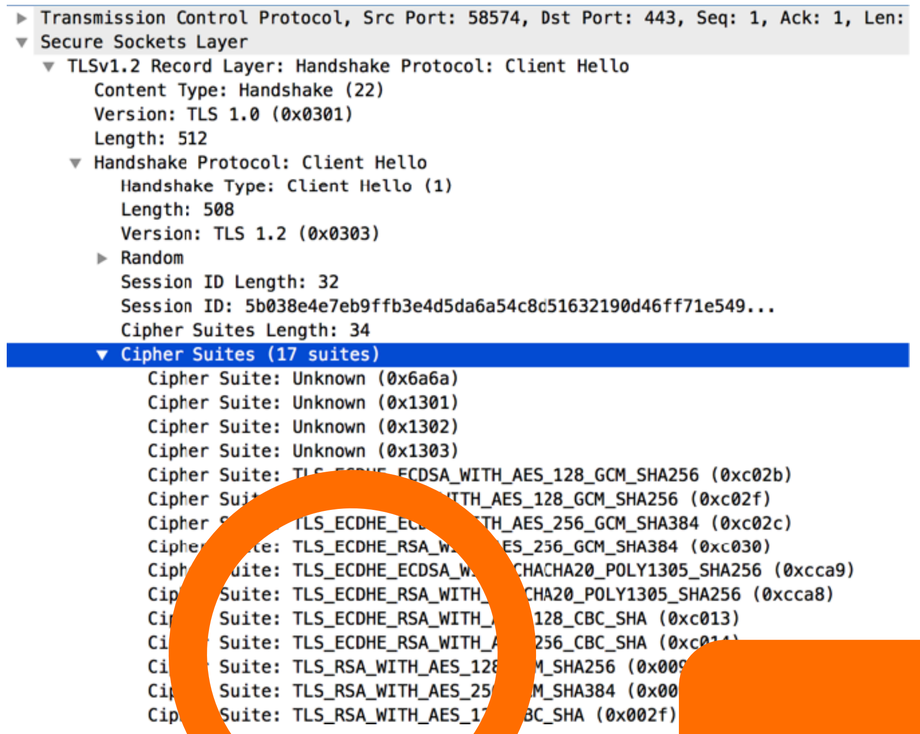
HTTPS 是如何工作的



SSL / TLS

	时间	状态	兼容性	备注
SSL 1.0	N/A	N/A	N/A	
SSL 2.0	1995	危险		
SSL 3.0	1996	危险	IE 6及以上	
TLS 1.0	1999	危险	IE 6及以上	和SSL 3.0差异很小
TLS 1.1	2006	危险	IE 8及以上	
TLS 1.2	2008	推荐	IE 8及以上	
TLS 1.3	2018	草案	Chrome 64及以上	

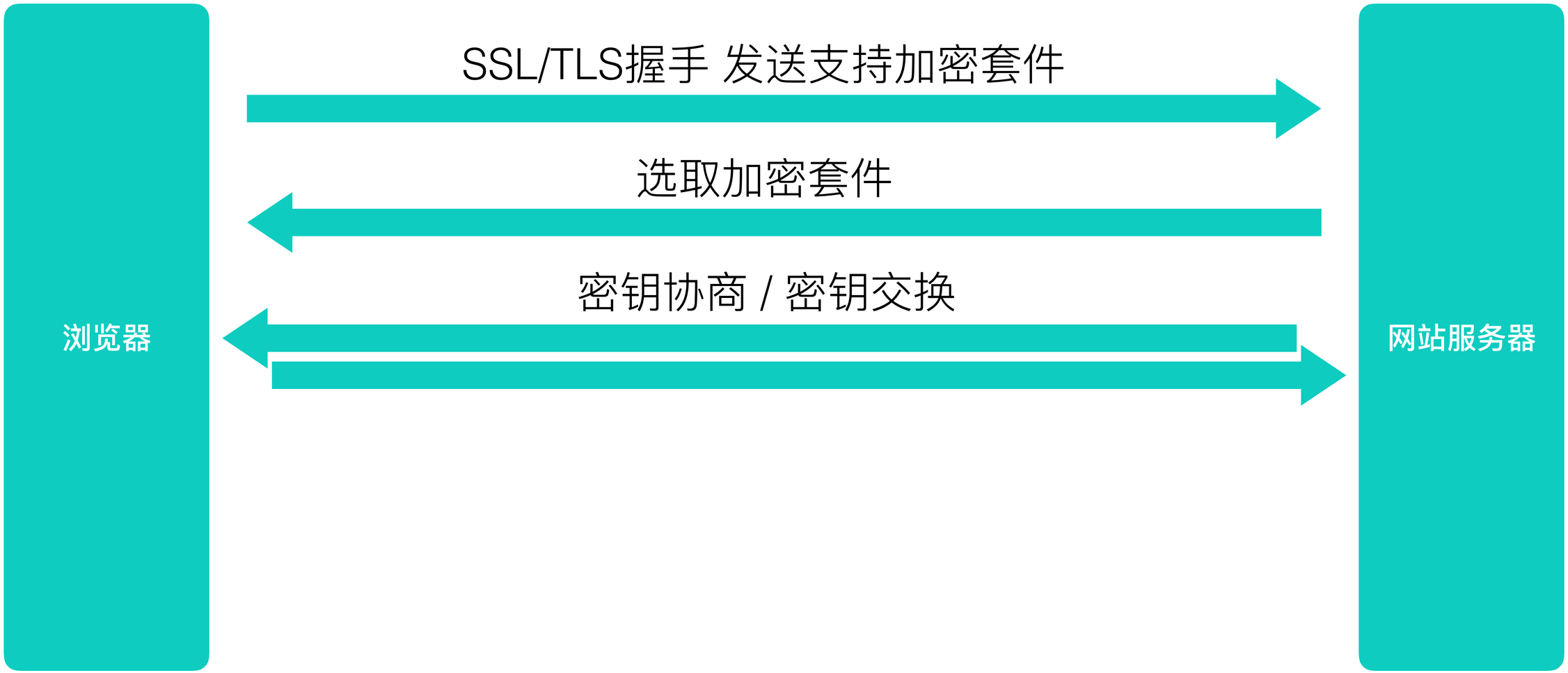
加密套件



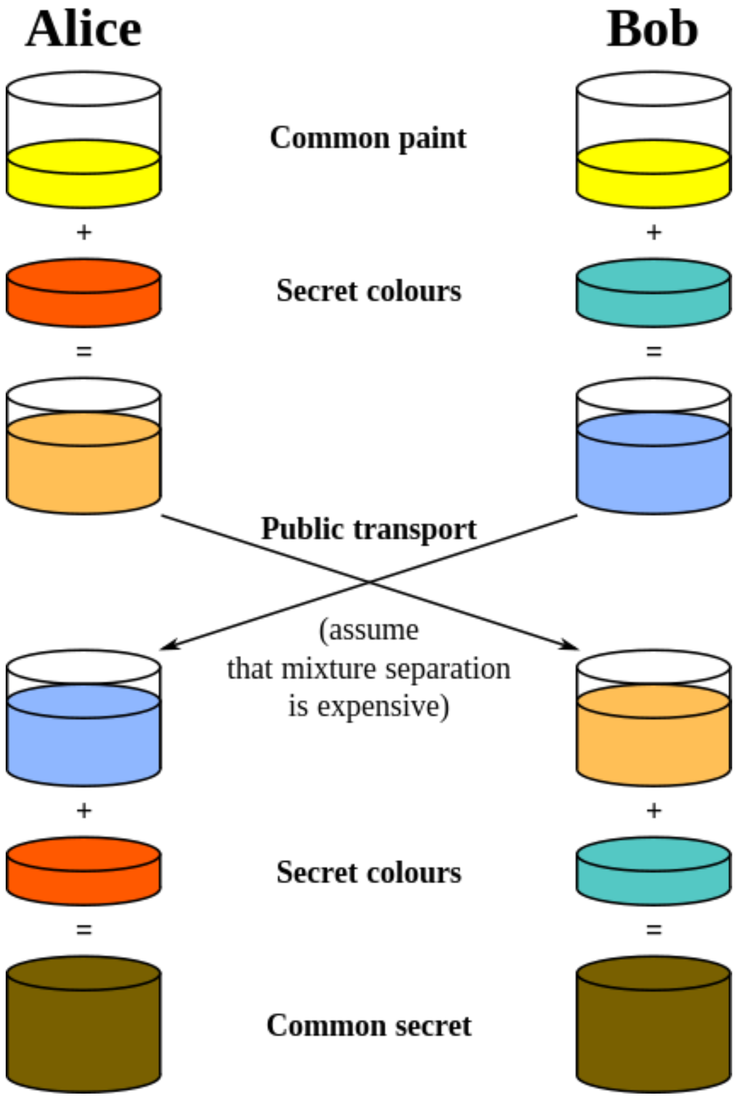
- 协议前缀: TLS
- 密钥交换 / 密钥协商: RSA
- 数据加密算法: AES_128_GCM
- 数据校验算法: SHA256

TLS_RSA_WITH_AES_128_GCM_SHA256

HTTPS 是如何工作的



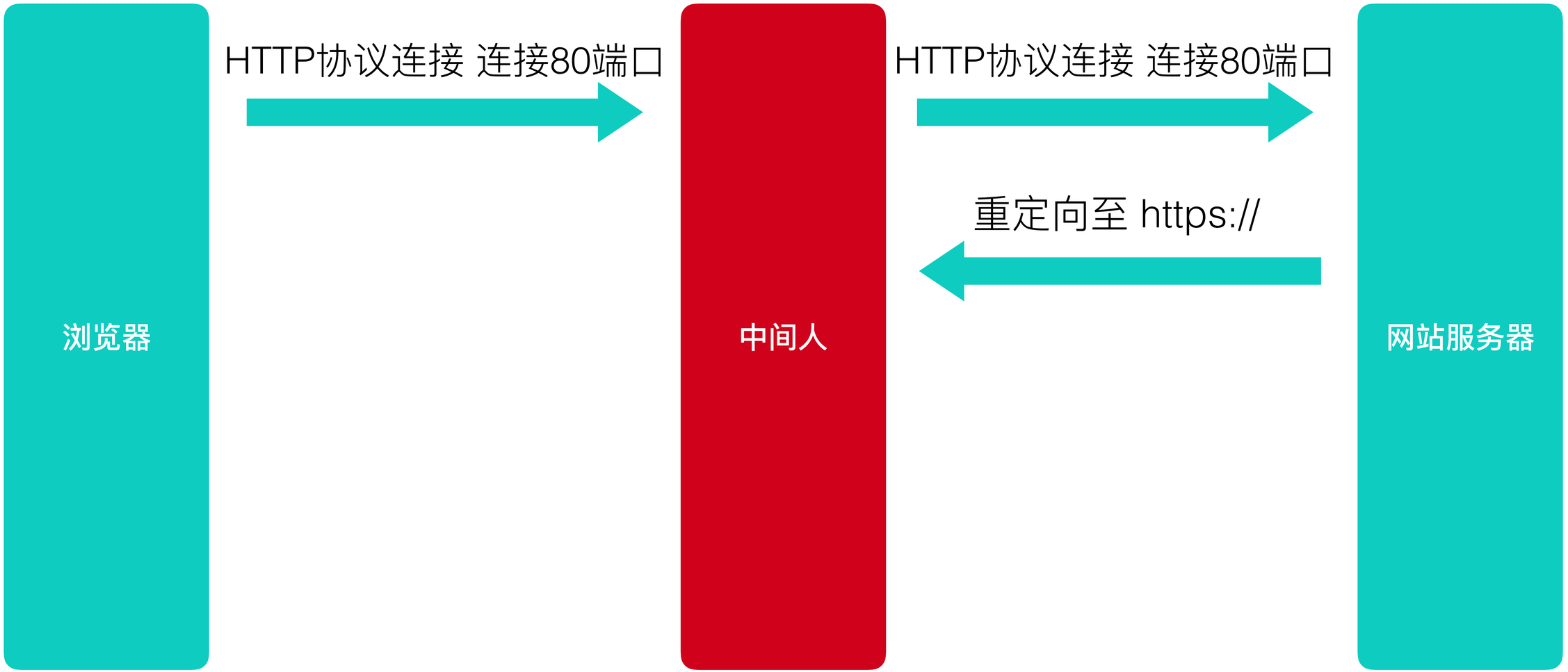
密钥协商



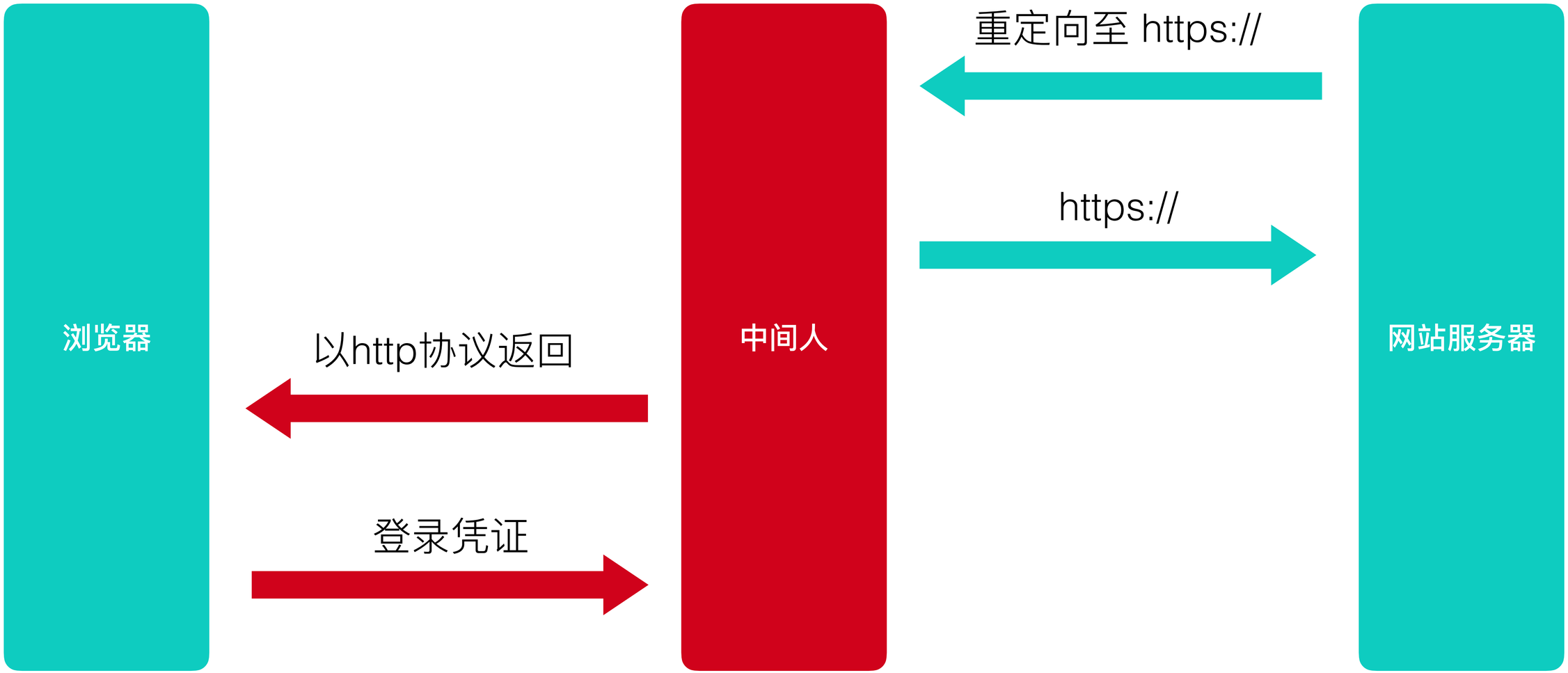
- 不是交换“密钥”，而是交换“生成密钥的信息”
- 密钥协商的过程可以被任意第三方围观
- 密钥协商需要认证身份，这是引入证书体系的关键

HTTPS 是如何工作的

SSL strip



SSL strip



HTTP Strict-Transport-Security

Strict-Transport-Security: max-age=<expire-time>

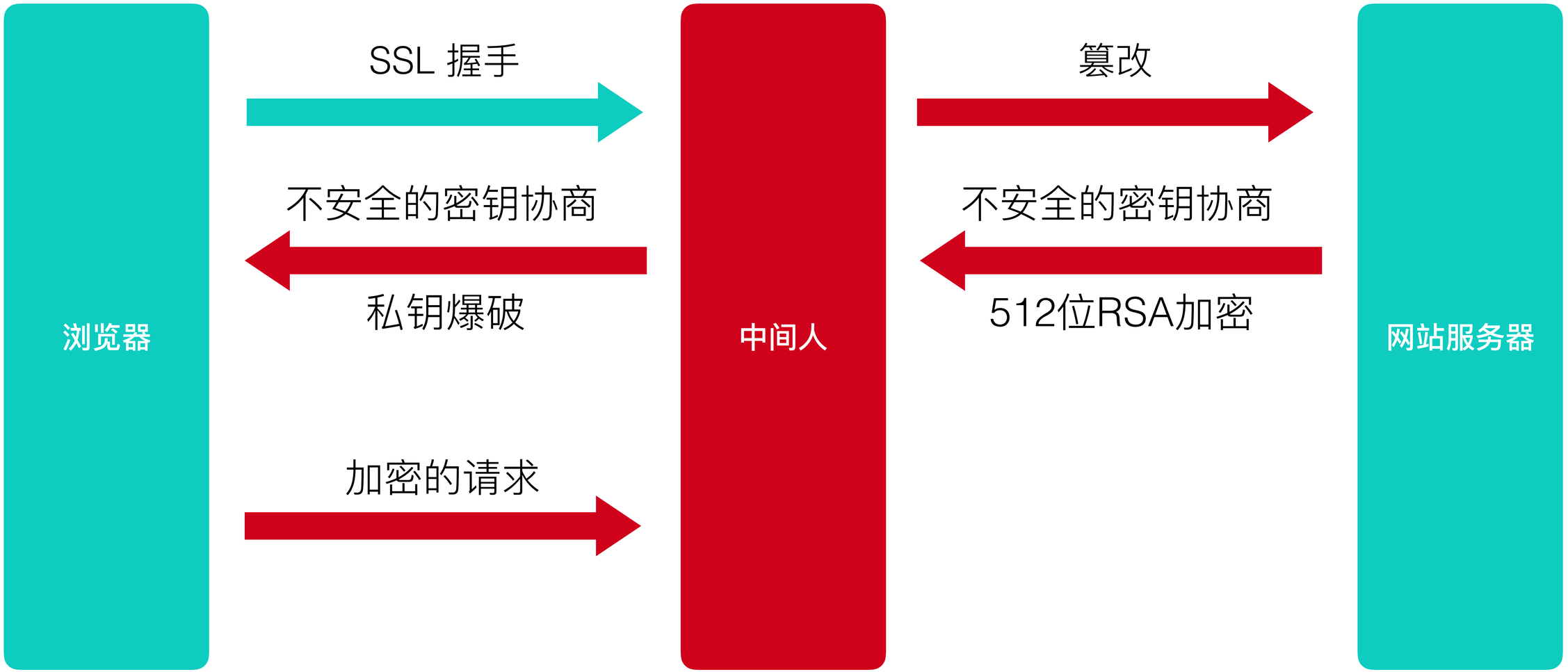
```
expires: Tue, 22 May 2018 13:10:24 GMT
npm-cost: 1
npm-remaining: 99
server: cloudflare
status: 200
strict-transport-security: max-age=31536000
vary: x-requested-with, x-spiferack, Accept-Encoding
x-clacks-overhead: terry pratchett
```

- 下次请求时必定使用 HTTPS
- 失效时间为1年
- Chrome 可以内置一部分域名

问题：

- HTTP 协议无效
- 用户的第一次访问不受控
- Chrome 的“预加载”列表名额有限

基于 SSL / TLS 的攻击 (FREAK)



512位的RSA

“

The purpose of the FaaS (Factoring as a Service) project is to demonstrate that 512-bit integers can be factored in only a few hours, for less than $100 of compute time in a public cloud environment. This illustrates the amazing progress in computing power over time, and the risk of continued use of 512-bit RSA keys.

”

在公有云服务上，512位长的整数可在数小时内因式分解，耗资不超过$100

https://github.com/eniac/faas

挑选安全的加密套装

https://wiki.mozilla.org/Security/Server_Side_TLS

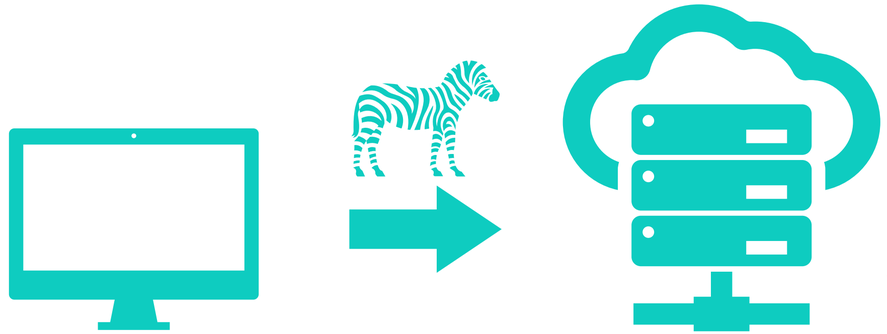
Modern compatibility

For services that don't need backward compatibility, the parameters below provide a higher level of security. This configuration is compatible wit
Opera 17, Safari 9, Android 5.0, and Java 8.

- Ciphersuites: **ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHA**
  SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDH
- Versions: **TLSv1.2**
- TLS curves: **prime256v1, secp384r1, secp521r1**
- Certificate type: **ECDSA**
- Certificate curve: **prime256v1, secp384r1, secp521r1**
- Certificate signature: **sha256WithRSAEncryption, ecdsa-with-SHA256, ecdsa-with-SHA384, ecdsa-with-SHA512**
- RSA key size: **2048** (if not ecdsa)
- DH Parameter size: **None** (disabled entirely)
- ECDH Parameter size: **256**
- HSTS: **max-age=15768000**
- Certificate switching: **None**

```
0xC0,0x2C   -   ECDHE-ECDSA-AES256-GCM-SHA384   TLSv1.2   Kx=ECDH   Au=ECDSA   Enc=AESGCM(256)     Mac=AEAD
0xC0,0x30   -   ECDHE-RSA-AES256-GCM-SHA384     TLSv1.2   Kx=ECDH   Au=RSA     Enc=AESGCM(256)     Mac=AEAD
0xCC,0xA9   -   ECDHE-ECDSA-CHACHA20-POLY1305   TLSv1.2   Kx=ECDH   Au=ECDSA   Enc=ChaCha20(256)   Mac=AEAD
0xCC,0xA8   -   ECDHE-RSA-CHACHA20-POLY1305     TLSv1.2   Kx=ECDH   Au=RSA     Enc=ChaCha20(256)   Mac=AEAD
0xC0,0x2B   -   ECDHE-ECDSA-AES128-GCM-SHA256   TLSv1.2   Kx=ECDH   Au=ECDSA   Enc=AESGCM(128)     Mac=AEAD
0xC0,0x2F   -   ECDHE-RSA-AES128-GCM-SHA256     TLSv1.2   Kx=ECDH   Au=RSA     Enc=AESGCM(128)     Mac=AEAD
0xC0,0x24   -   ECDHE-ECDSA-AES256-SHA384       TLSv1.2   Kx=ECDH   Au=ECDSA   Enc=AES(256)        Mac=SHA384
0xC0,0x28   -   ECDHE-RSA-AES256-SHA384         TLSv1.2   Kx=ECDH   Au=RSA     Enc=AES(256)        Mac=SHA384
0xC0,0x23   -   ECDHE-ECDSA-AES128-SHA256       TLSv1.2   Kx=ECDH   Au=ECDSA   Enc=AES(128)        Mac=SHA256
0xC0,0x27   -   ECDHE-RSA-AES128-SHA256         TLSv1.2   Kx=ECDH   Au=RSA     Enc=AES(128)        Mac=SHA256
```

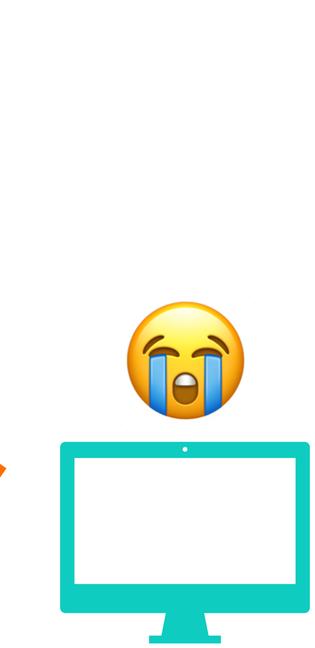
什么是 CDN



北京用户

网站服务器

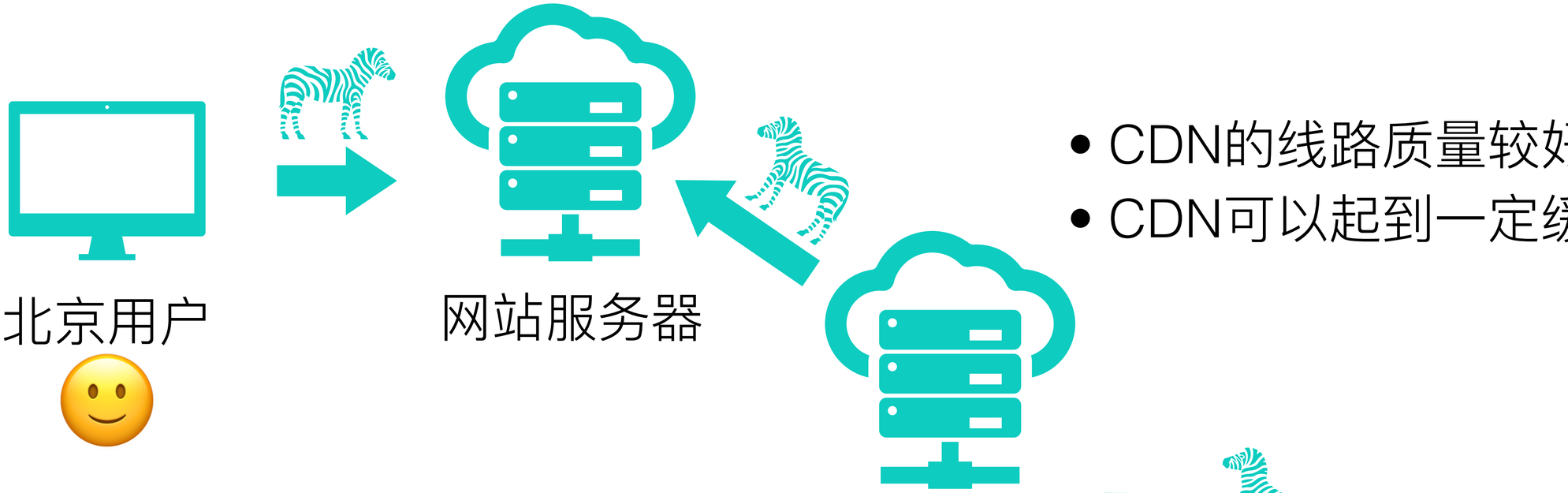
- 用户网络环境千差万别
- 每个宽带商线路不同



海南用户

用户和机房的距离客观存在

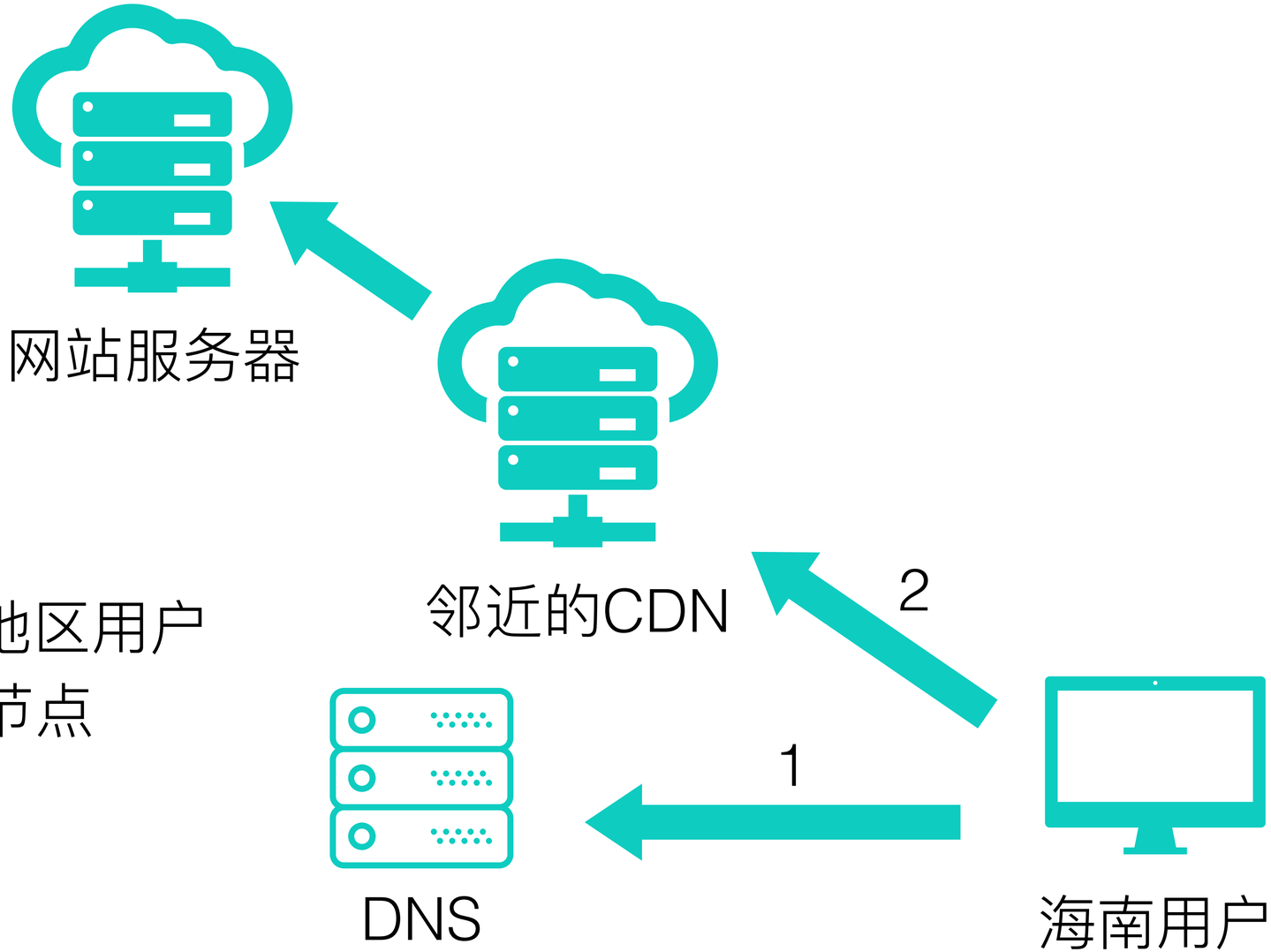
什么是 CDN



- CDN的线路质量较好
- CDN可以起到一定缓存作用

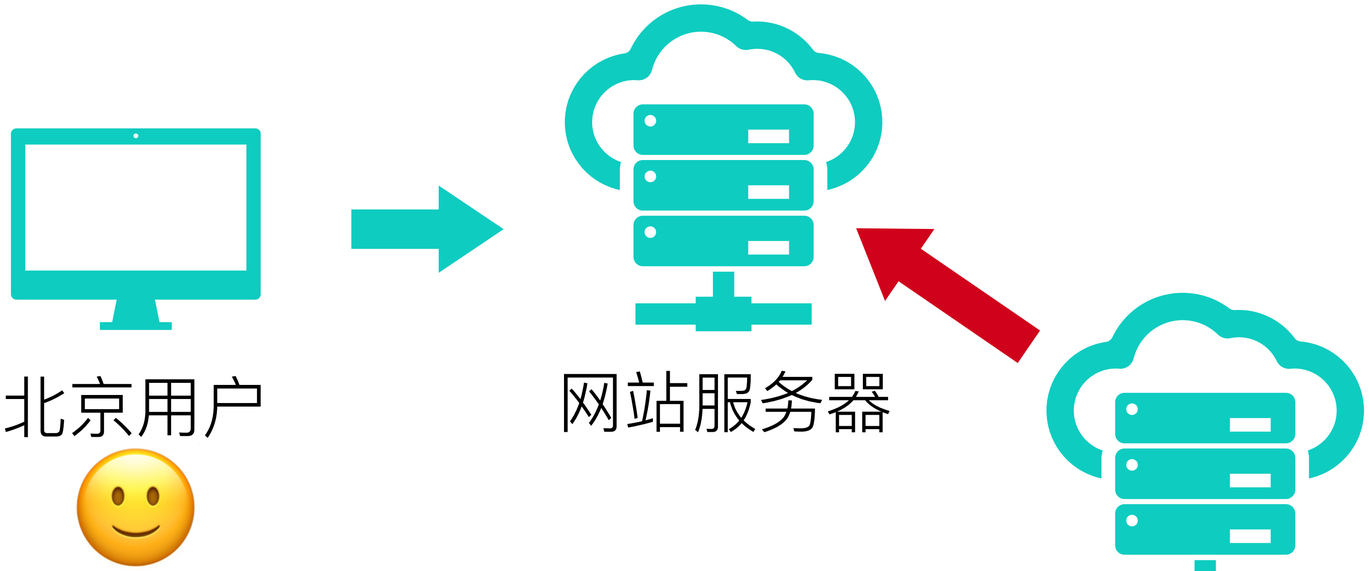
利用邻近的服务器加速网站内容的访问

引流 CDN



利用 DNS 引导不同地区用户
至不同的 CDN 节点

CDN 上的流量劫持

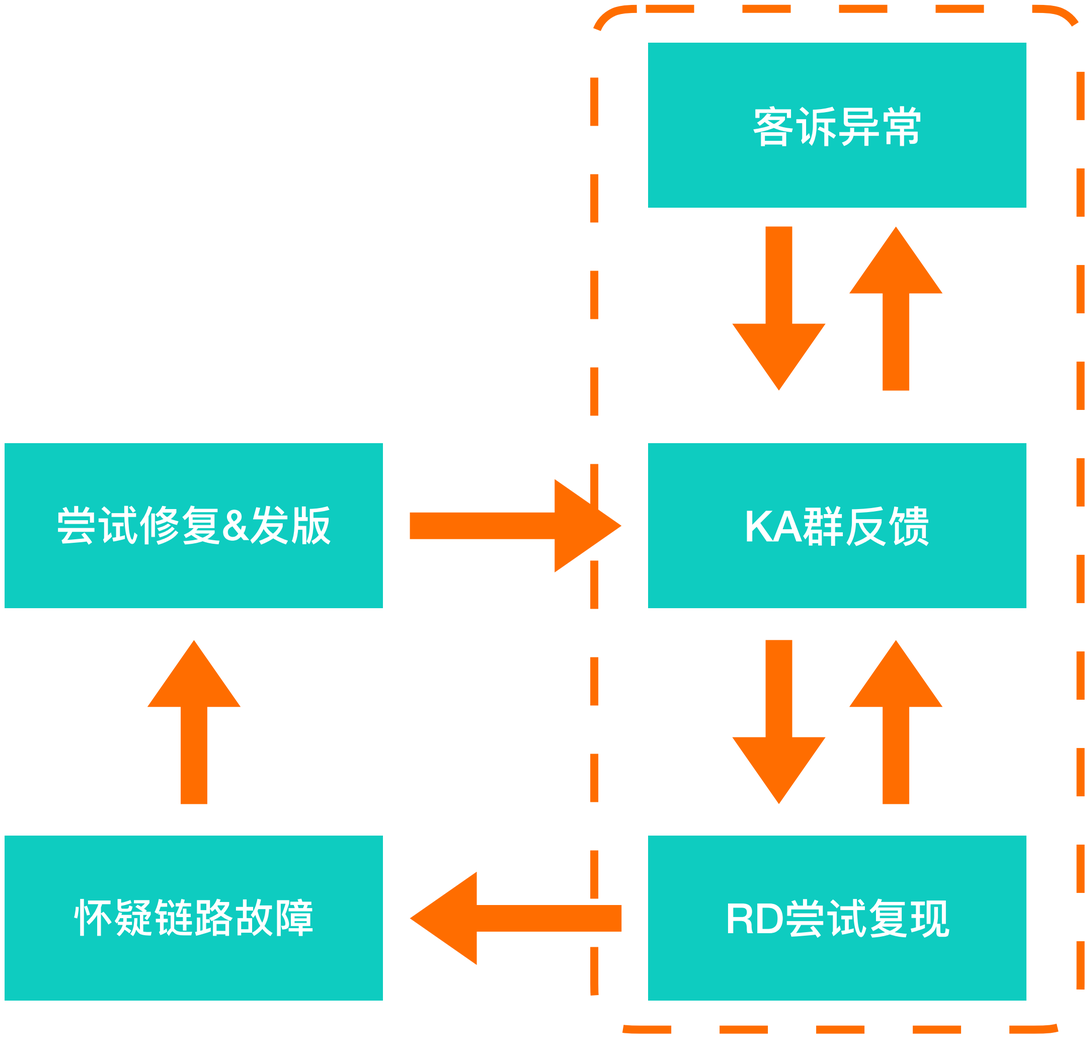


- 一次影响大批量用户
- 维护者难以发现

CDN 的回源链路如果是 HTTP 线路?

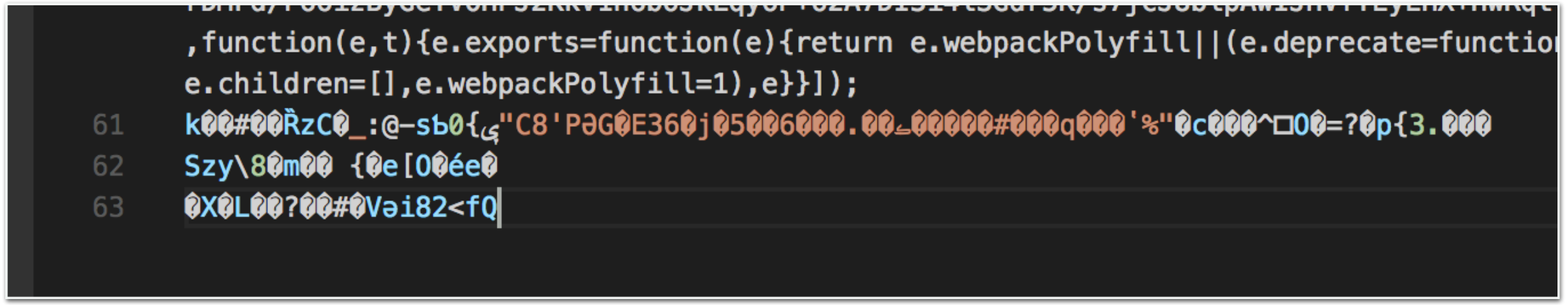
链路问题的排查

- 故障的现象和兼容性问题、跨域问题相似
- 排查过程需要反复和最终用户沟通
- 反复回滚、发布可能导致更多问题



链路故障的特点

- 本来应该返回200的请求，实际变成0（http不成功）
- 突然发生在某一地区、省份的运营商服务区域
- 没有统一的解决方案，**但不代表不能及时发现**

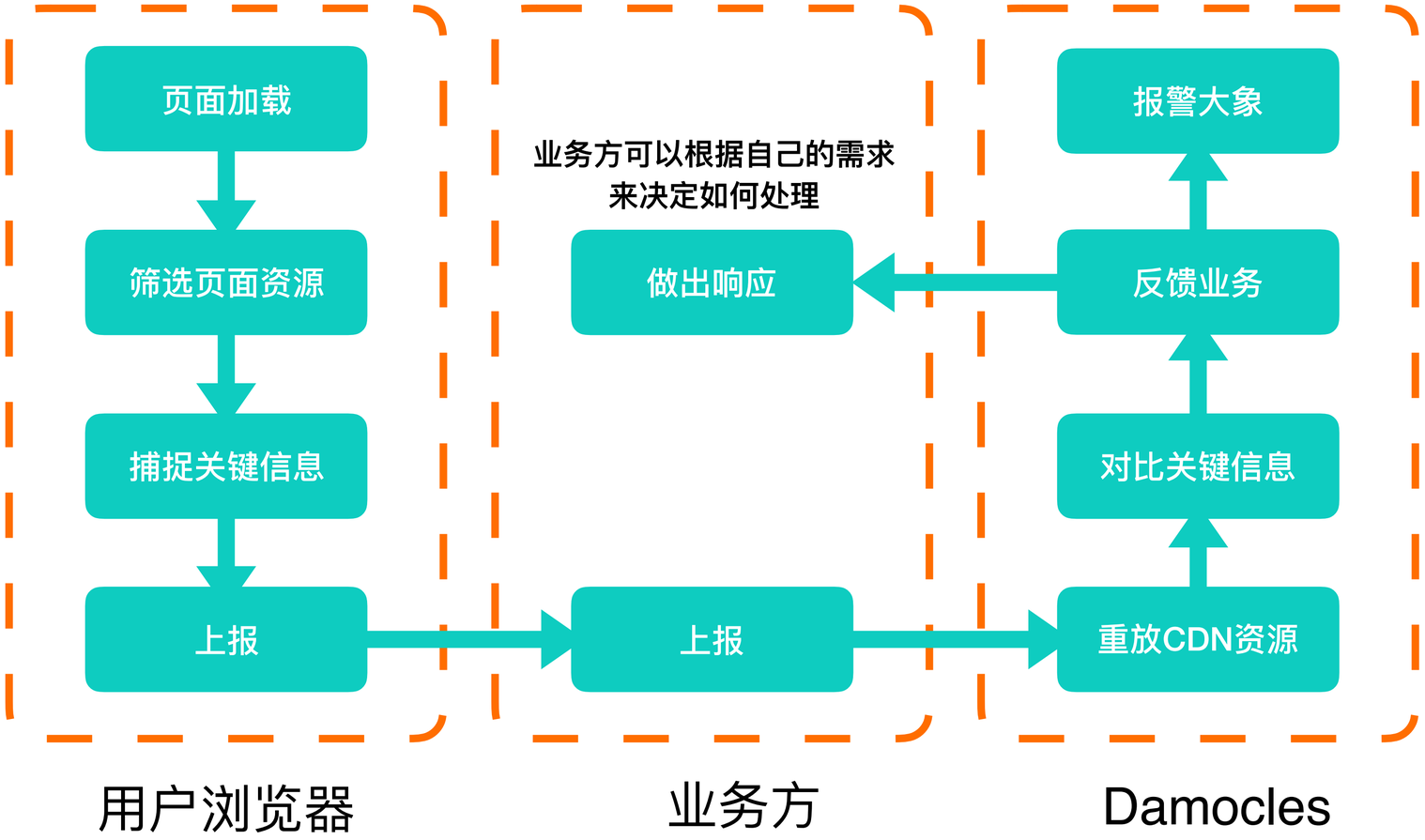


某CDN节点返回的资源，因gzip出错，实际返回的内容已彻底无法使用

现有方案

- **方案A：在某些省份、地区自建监测站，定期抓取固定资源**
  - 问题：资源太固定，监测站数量也远远不够
- **方案B：业务方在自己的html中监听资源的error事件**
  - 问题：无法确认问题在于链路，也可能只是普通的js出错
- **方案C：使用第三方企业服务进行监控**
  - 问题：服务越多成本越高
- **方案D：CSP、SRI**
  - 问题：兼容性和灵活性差，无法进行自定义逻辑

基于代码校验的防治方案



- 监控级别是业务级，而不是某个URL
- 给业务方足够的空间来做降级
- 通过集中分析来降低误判、警报风暴的可能性
- Damocles 服务故障不影响业务

浏览器下载->发送信息->服务器下载->比对信息

效果

- 3个月检测文件超过**10,480,084**次
- 3月13日北京地区联通线路发生资源劫持，我们在21点发现并及时通知SRE进行了CDN切换，通知PM做好了客诉预备

Q&A

- 美团金融持续招收大前端
- Node.js? Service Worker? WebAssembly?
- 欢迎一起实践