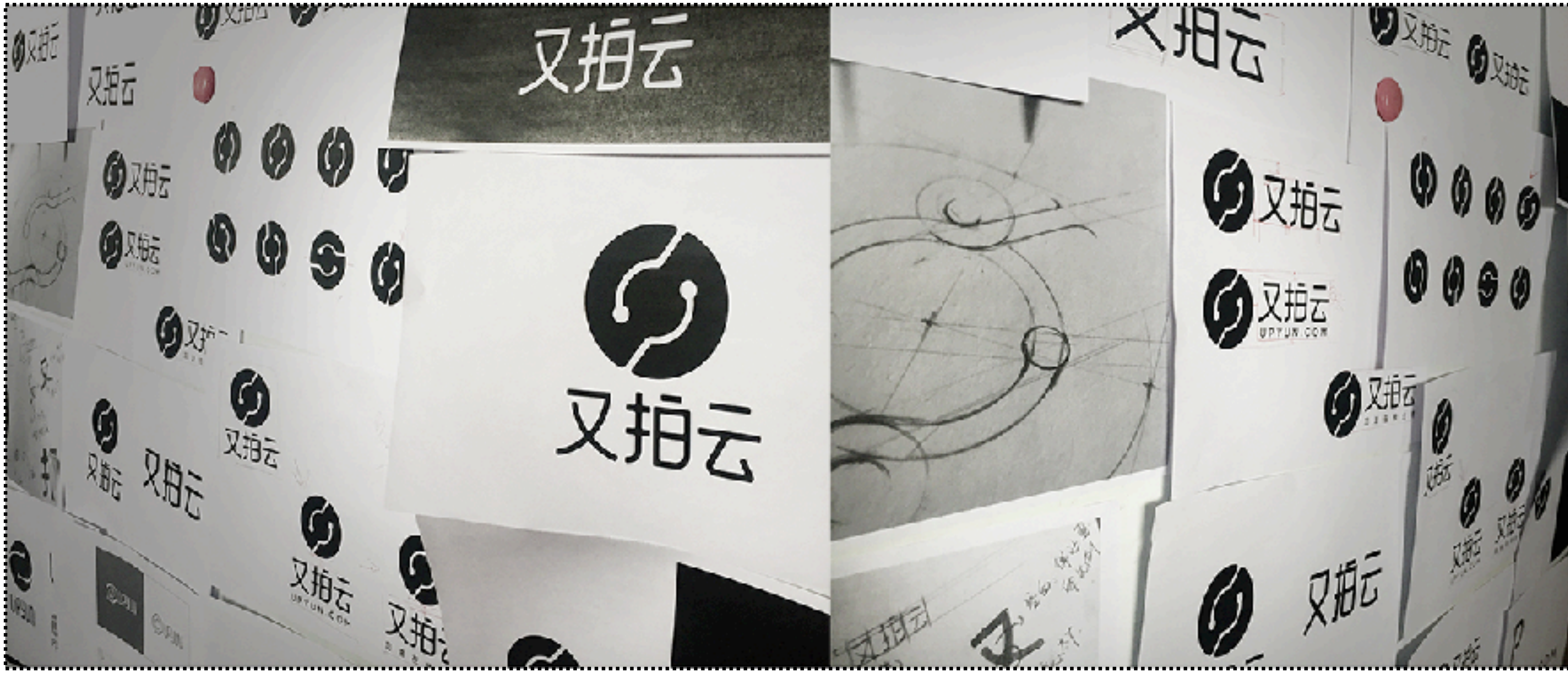
又拍云 CDN 技术特点与优势

2017.04 @ 杭州

又拍云



新 LOGO，**新开始**

云分发

云处理

云存储

又拍云

直播云

点播云

全站加速

流量营销服务

SSL 证书服务

融合云存储

图片鉴黄

视频转码

...

又拍云 CDN 关键技术介绍

- 自定义 SSL 服务
- 自主研发 DNS 系统
- **CDN 文件分段缓存**
- **CDN 支持简单编程**
- CDN 常规功能特性
- CDN 实时日志分析
- **CDN 技术架构演进**

自定义 SSL 服务:

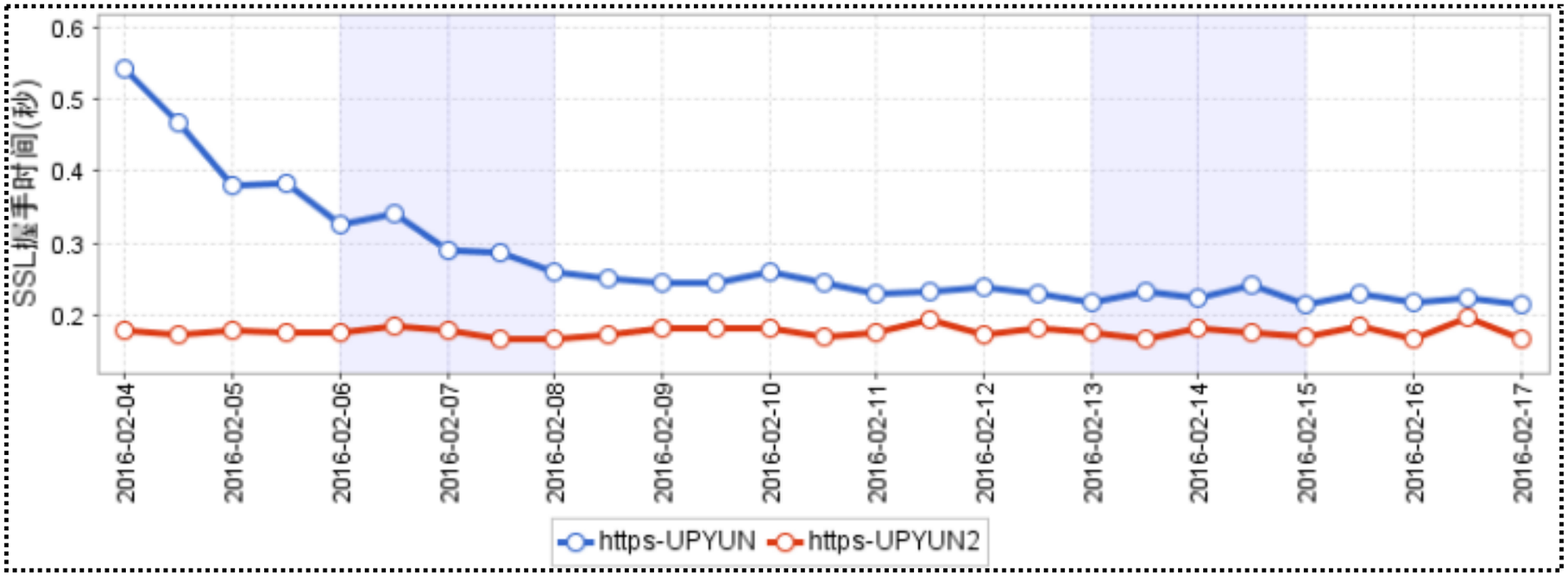
Certificates Load & OCSP stapling


```
server {
    listen 443 ssl;
    server_name upyun.com;

    ssl_certificate       upyun.com.pem;
    ssl_certificate_key   upyun.com.key;

    ssl_stapling on;
    ssl_stapling_verify on;
    ssl_trusted_certificate /etc/ssl/private/ca-certs.pem;
}
```

HTTPS **性能**和**兼容性**优化

- HTTP/2
- OCSP Stapling
- HTTP Strict Transport Security (**HSTS**)
- SSL Session ID 非阻塞分布式缓存实现
- 证书链完整性检测和自动补全
- **Let's Encrypt** & Symantec & GeoTrust & **亚洲诚信**

OCSP Stapling



Enable OCSP Stapling	Average SSL Connection Time
Yes	0.176s
No	0.267s

使用 OCSP Stapling 可以在 SSL 握手阶段节约 90+ 毫秒， 性能提升了 **34.08%** 左右。

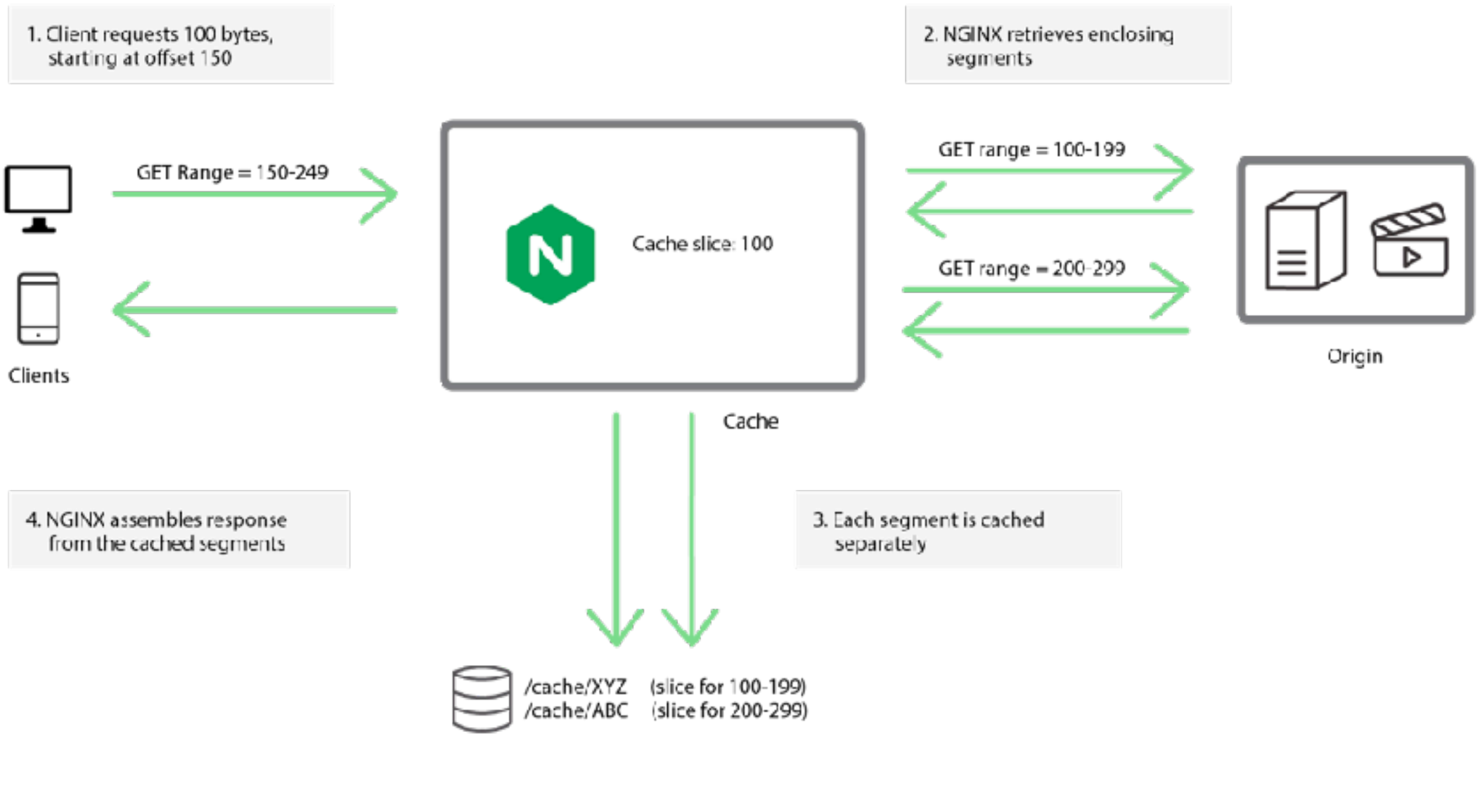
自主研发 DNS 系统

- 全面支撑又拍云 CDN 的 DNS 智能解析
- 完整的**自动化**监控、报警、上线流程
- 基于纯 C 开发，**高性能**（QPS 200w/s）、高可靠、易扩展
- DNS 分层调度策略
- HTTP DNS
- Web 管理后台

大文件点播 CDN 加速**一般面临的问题**

- 单个文件需要支持 GB 级别的缓存，对单机磁盘容量是个考验。
- 一致性哈希缓存策略下，热门文件下载容易拖垮某一命中机器。
- 下载未完成中途断开的情况需要重新回源缓存，**且重试代价高。**
- Range 请求支持不友好，**会造成回源带宽浪费过多。**
- 视频文件点播需要服务端支持拖拉，视频过大可能存在性能问题。
- 客户本地 DNS 混乱，导致域名解析覆盖不准，**对下载速度影响大。**

CDN 文件分段缓存


1. Client requests 100 bytes, starting at offset 150
2. NGINX retrieves enclosing segments
GET Range = 150-249
GET range = 100-199
GET range = 200-299
Cache slice: 100
Clients
Origin
Cache
4. NGINX assembles response from the cached segments
3. Each segment is cached separately
/cache/XYZ (slice for 100-199)
/cache/ABC (slice for 200-299)

又拍云 CDN 如何实现**大文件快速分发**?

- **大文件分段缓存：**每个大文件按 64KB 大小切割成 N 个分片，每个分片在 CDN 内部是一个独立的缓存单元，同一个文件的不同分片都拥有唯一的 URL 标记，在节点内部这些分片会根据 URL 哈希进行负载均衡，最终每个分片请求以 HTTP Range 形式回客户源或又拍云存储。

- **分段缓存预加载：**文件下载的时候多个分片会并发进行加载，以便避免因为回源或上游服务器慢导致的客户端下载卡顿问题。

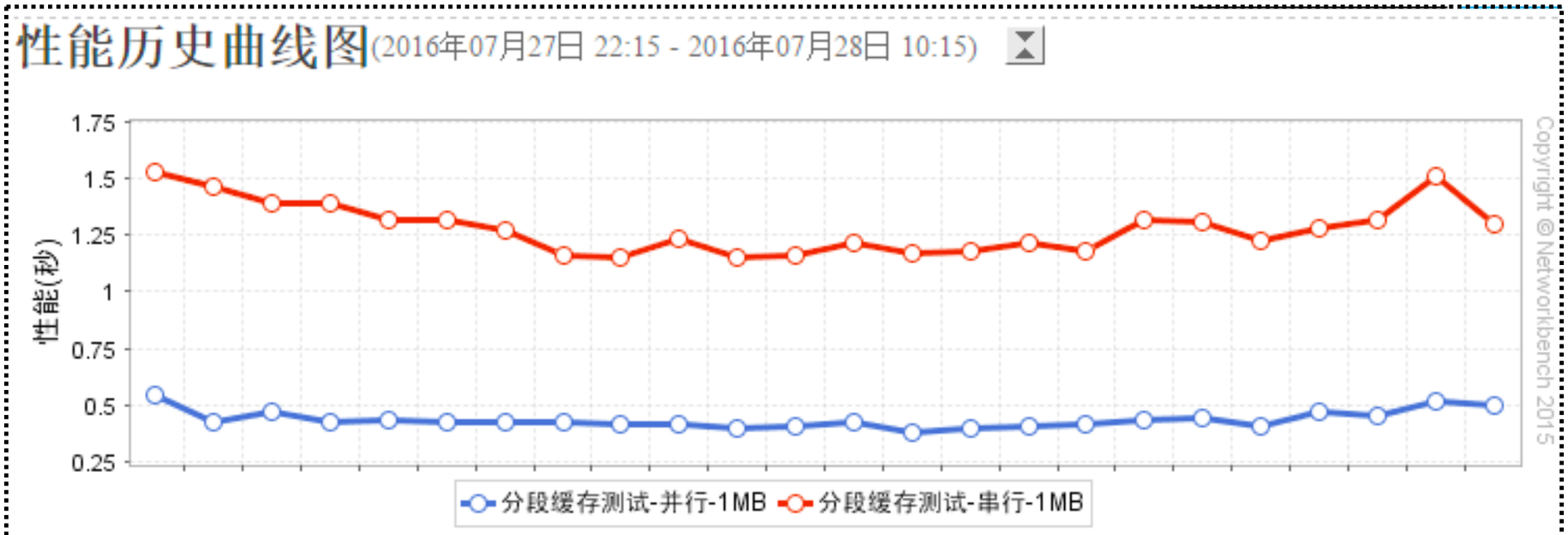
- **DNS 纠正调度：**客户端解析到错误的 CDN 节点，服务端会进行自动纠正触发 302 跳转，使客户端连接到最快的 CDN 节点去下载文件。

- **CDN 文件预热：**我们提供了预热 API 接口，可以对大文件提前进行预热，避免首次回源速度不理想的情况。

大文件分段缓存的**一些优势**

- 按需回源，可以有效降低源站带宽压力，避免不必要的带宽浪费。
- 由于不需要去下载整个文件，并且配合分段缓存预加载功能，可以有效提高了大文件下载的速度。
- 按块缓存，可以提高文件在 CDN 节点的缓存命中率。
- 由于文件按块缓存，分块文件可以散落在 CDN 节点集群的不同机器上，可以有效缓解热点文件导致的机器负载问题。
- **开启分段缓存后，视频拖拉也更加实时高效，对点播业务效果提升明显。**
- **结合分片重试技术，还能有效降低全网文件下载卡顿率。**

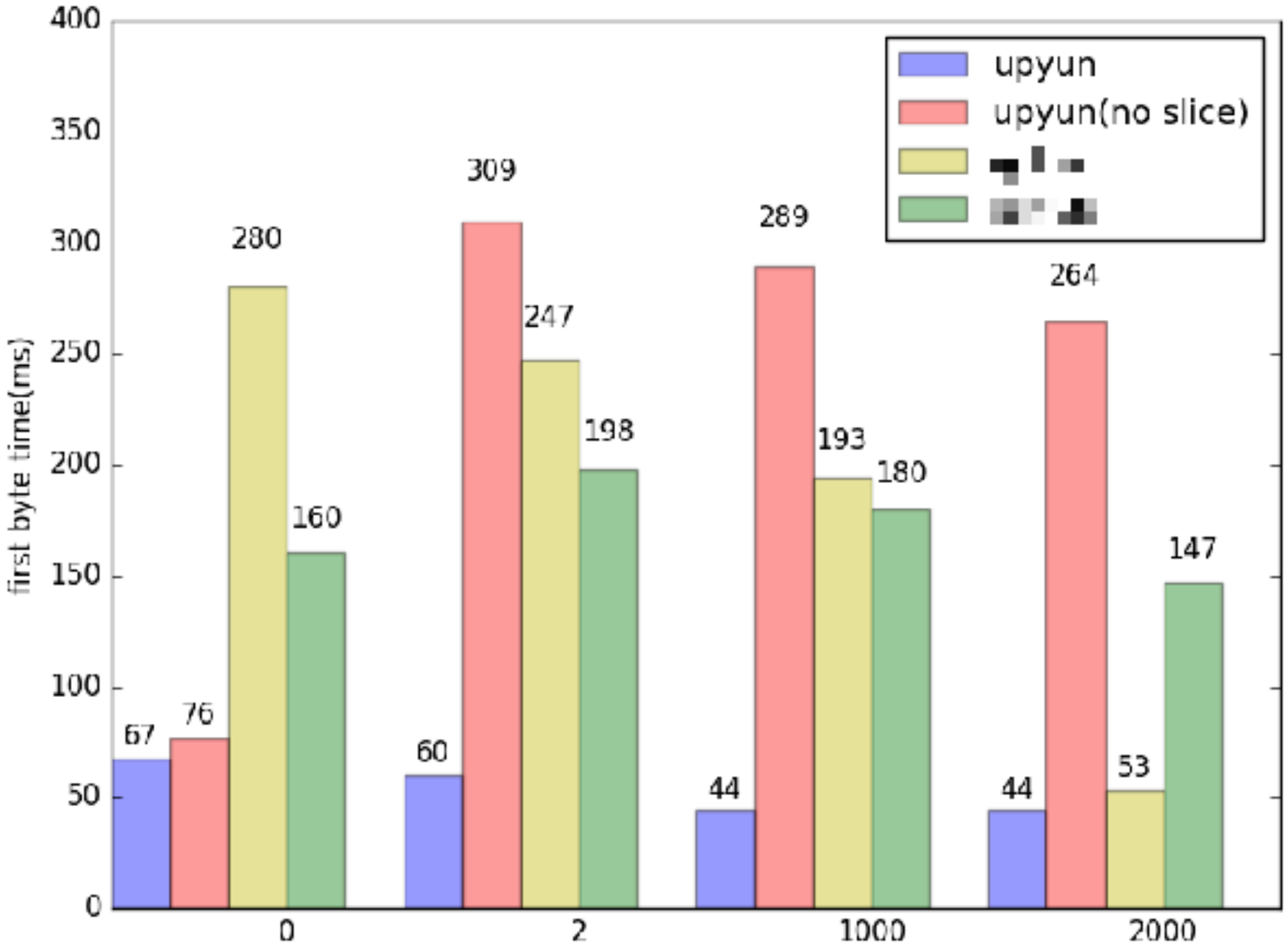
分段缓存**预加载**技术



文件大小	MISS-串行	MISS-并行	HIT-串行	HIT-并行
1MB	1.779	1.065	0.358	0.447
5MB	7.698	2.436	1.769	1.731
50MB	72.619	20.461	18.554	18.329
10KB	0.211	0.186	0.037	0.048

当客户端请求第一个分块文件时，CDN **会提前去下载**后面几个分块文件，**进而提高文件下载速度**，这个特性主要在**缓存未命中的情况下效果明显**，可以很好的**提升首次访问速度，降低首播延迟**，对视频点播类客户特别有效。

大文件视频拖拉性能对比



- 测试视频大小：248MB
- 测试视频类型：MP4
- 时间偏移：0,2,1000,2000

横轴表示视频进度条的拖拽时间点，纵轴是响应的首包响应时间（ms），每个测试点测试了 100 遍，去除一个最高值一个最低值之后计算获得平均值。**结论：又拍云 CDN 大文件视频拖拉在分段缓存模式下性能优势明显。**

又拍云 CDN 如何节省大文件回源带宽?

- **分段缓存技术**对 Range 请求很友好，按需加载，同时也有助于提升文件整体的缓存命中率。

- 区域回源特性让多个 CDN 中心节点**能集中回源**，降低重复回源次数。

- **镜像迁移**功能，确保热点文件平滑迁移到云存储，避免 CDN 缓存过期后再次回源。

大视频文件**实测**

1. 第一个用户完整下载（**未缓存**）：**1.64 MB/s**

```
test.mp4      100%[=====================================================>]   1.95G   1.82MB/s    in 20m 19s

2017-02-28 11:54:38 (1.64 MB/s) - ‘test.mp4’ saved [2096821300/2096821300]
```

2. 第二个用户完整下载（**已缓存**）：**3.98 MB/s**

```
test.mp4      100%[=====================================================>]   1.95G  3.89MB/s    in 8m 22s

2017-02-28 12:03:15 (3.98 MB/s) - ‘test.mp4’ saved [2096821300/2096821300]
```

3. 第一个用户在**未缓存情况下**下载到 50%，第二个用户**立马从头完整下载**：可以看到前面 50% 的下载速率保持在已缓存情况下比较快的水平，达到 4 MB/s 左右，后 50% 的下载速率因为没有缓存的原因退化到 2 MB/s 左右，符合未缓存情况下的速率指标。整体速率：**3.01 MB/s**

```
test.mp4                      40%[====================>                ] 801.64M  4.27MB/s    eta 4m 52s
```

```
test.mp4                      62%[================================>    ]   1.23G  1.96MB/s    eta 3m 22s
```

```
test.mp4      100%[=====================================================>]   1.95G  2.21MB/s    in 11m 5s

2017-02-28 13:39:40 (3.01 MB/s) - ‘test.mp4’ saved [2096821300/2096821300]
```

大视频文件实测

第一个用户首播时间（**未缓存**）： **2.378s**

协议类型	http
HTTP Server	nuocoo/0.25
HTTP Via	T:65139.M:1, V:407=zj-lna=133, S:mix-hz-fdi=166, T:101172.M:1, V:mix-hz-fdi=168, T:3525.17:1, M:ctr=zj-lna3=719
文件大小	2,096,821,300 (1999.7M)
视频/音频长度	5591.000s
网络性能指标	
总下载字节数	285,801,116 (272.6M)
DNS时间	0.008s
TCP连接时间	0.014s
流媒体请求时间	0.000s
首包时间	0.019s
内容下载时间	118.788s
流媒体性能指标	
卡缓冲次数	0
缓冲前准备时间	1.533s
缓冲时间	0.845s
再缓冲时间	0.000s
等待时间	2.378s
比特率	18614.600 Kbps
码流	2929.961 Kbps
监测时长	118.879s
首次播放时长	116.501s
播放时长	116.501s
点击流媒体首播时间	2.378s
错误代码	--

首播时间

大视频文件实测

第二个用户首播时间（**已缓存**）： **1.025s**

协议类型	http
HTTP Server	marco/0.2.5
HTTP Via	T 5207.M.1, V403-zj-fuid-202, S mix-hz-6ii-171, T 101166.M.1, V mix-hz-5ii-169, T 356.T.1, M ctn-zj-lne2-005
文件大小	2,096,821,300 (1999.7M)
视频/音频长度	5591.000s
网络性能指标	
总下载字节数	649,246,449 (619.2M)
DNS时间	0.052s
TCP连接时间	0.010s
流媒体调度时间	0.000s
首包时间	0.046s
内容下载时间	115.737s
流媒体性能指标	
再缓冲次数	0
缓冲前准备时间	0.777s
缓冲时间	0.248s
再缓冲时间	0.000s
等待时间	1.025s
比特率	42268.649 Kbps
码流	2929.561 Kbps
监测时长	119.892s
首次播放时长	118.870s
播放时长	118.870s
真正流媒体首播时间	1.025s 首播时间
错误代码	--

DNS 纠正调度

某个天津电信的用户由于 DNS 配置问题解析到了江苏移动 ...

```
timebug@harmless:~$ http http://img.huaban.com/test.mp4

HTTP/1.1 302 Moved Temporarily
Connection: keep-alive
Content-Length: 161
Content-Type: text/html
Location: http://123.150.200.130/img.huaban.com/test.mp4?
_up_sum=a738dc&_up_id=0f8b04a82480906a2a09bfda8d864c84&_up_from=112.21.160.135
Server: marco
```

江苏常州 移动（112.21.160.135）-> **天津市 电信**（123.150.200.130）

CDN 文件预热

当源站增加新的资源文件时，可将这些新文件**提前缓存至 CDN 节点**，最终用户**第一次访问这些资源文件时**，CDN 节点**不需要回源站**请求文件资源了。一方面**可减少用户突发访问时的回源量**，另一方面可以**加快资源文件的首次访问速度**。

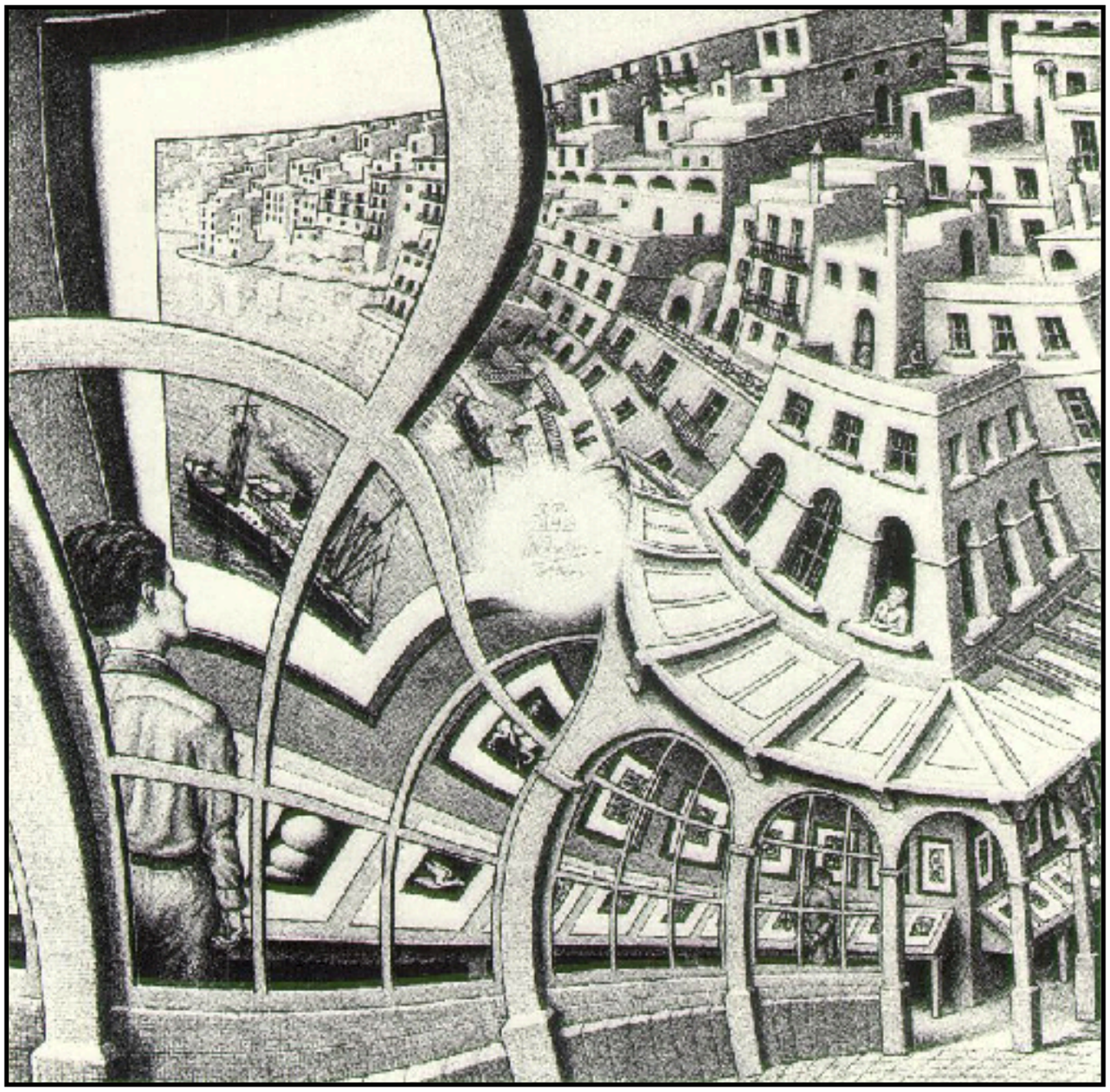
延伸问题：如何实现未完成大文件完整缓存时，同一 CDN 节点上第一个用户访问和第二个用户访问的快速命中分片资源?

答：利用又拍云 CDN **分段缓存技术**，一个大文件实际上是从头到尾分成 N 个 64KB 的小文件进行回源的，并且这些分片文件都是独立且大小对齐的缓存单元，一个大文件在下载过程中，已经下载完成的分片立刻就能被 CDN 缓存起来了，即便这个大文件在客户端层面下载到一半中断了请求，第二个用户访问的时候也能直接命中到那些已经缓存住的分片，从而**不需要重新加载回源**。

分段缓存对源站的**一些要求**

- 源站支持 HTTP Range 请求 **Accept-Ranges: bytes**。
- 源站支持 ETag 输出，ETag 为文件内容 **MD5** 或能表示文件唯一性的字符串。
- 源站支持**非 chunked** 形式的响应体。

又拍云存储完美兼容。



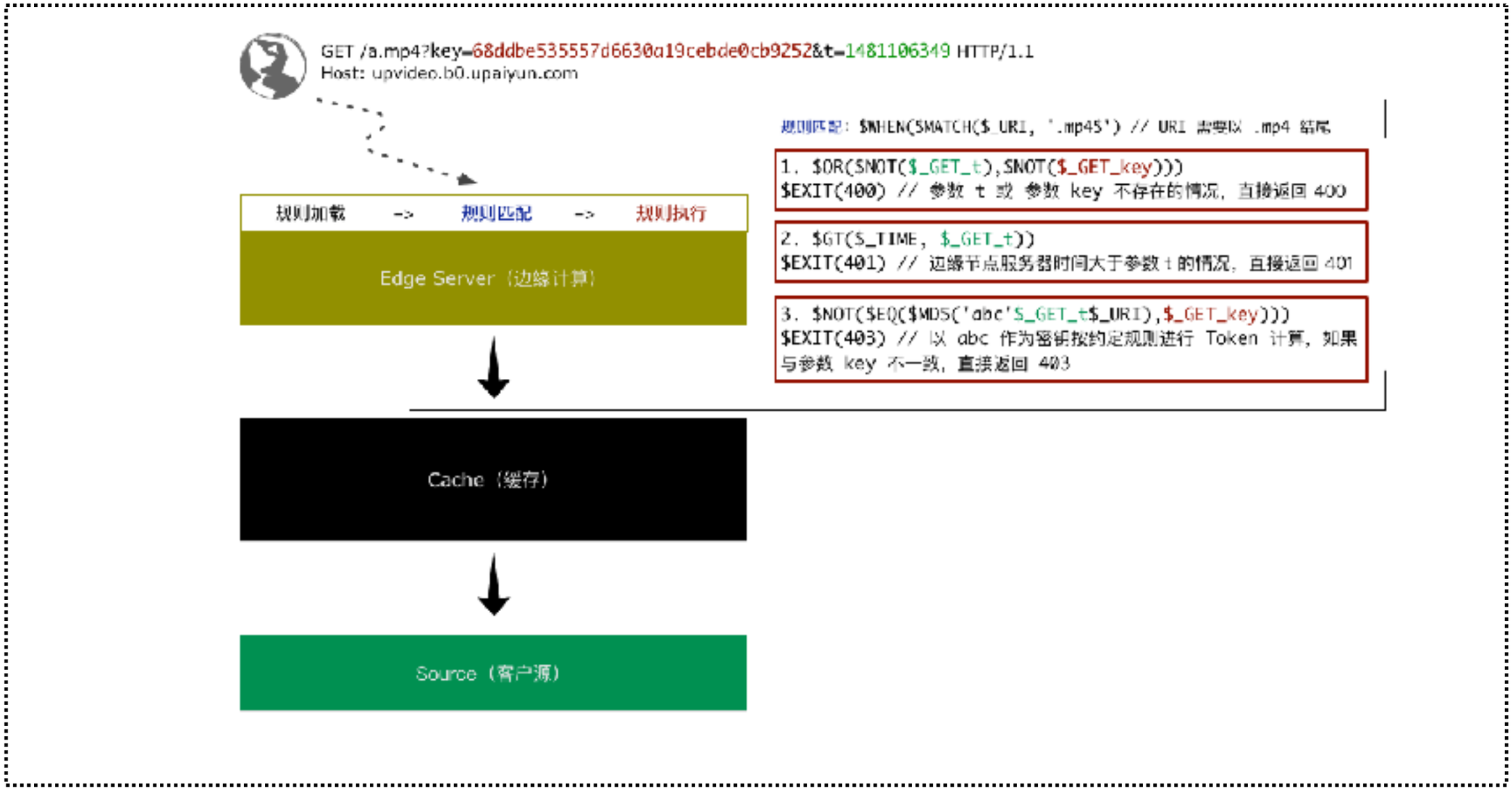
需求变化那么快

我们需要**以不变应万变**。

又拍云 CDN 支持**简单编程**

- 大量的服务器部署在各个省份
- 边缘计算
- 2014 年正式提出 **CDN + Cloud** 概念
- 适用场景

可编程 CDN 架构


GET /a.mp4?key=68ddbe535557d6630a19cebde0cb9252&t=1481106349 HTTP/1.1
Host: upvideo.b0.upaiyun.com
规则加载 -> 规则匹配 -> 规则执行
Edge Server（边缘计算）
Cache（缓存）
Source（客户源）
规则匹配：$WHEN($MATCH($_URI, '.mp4$') // URI 需要以 .mp4 结尾
1. $OR($NOT($_GET_t),$NOT($_GET_key)))
$EXIT(400) // 参数 t 或 参数 key 不存在的情况，直接返回 400
2. $GT($_TIME, $_GET_t))
$EXIT(401) // 边缘节点服务器时间大于参数 t 的情况，直接返回 401
3. $NOT($EQ($MD5('abc'$_GET_t$_URI),$_GET_key)))
$EXIT(403) // 以 abc 作为密钥按约定规则进行 Token 计算，如果
与参数 key 不一致，直接返回 403

可编程 CDN

http://io.upyun.com/2015/03/09/hello-world/?foo=bar

[scheme] [host] [path] [query]

可编程 CDN

目前为止：

- 支持 **40** 个实用函数
- 支持 **15** 个常用变量
- 覆盖多种不同业务场景

NGINX Rewrite

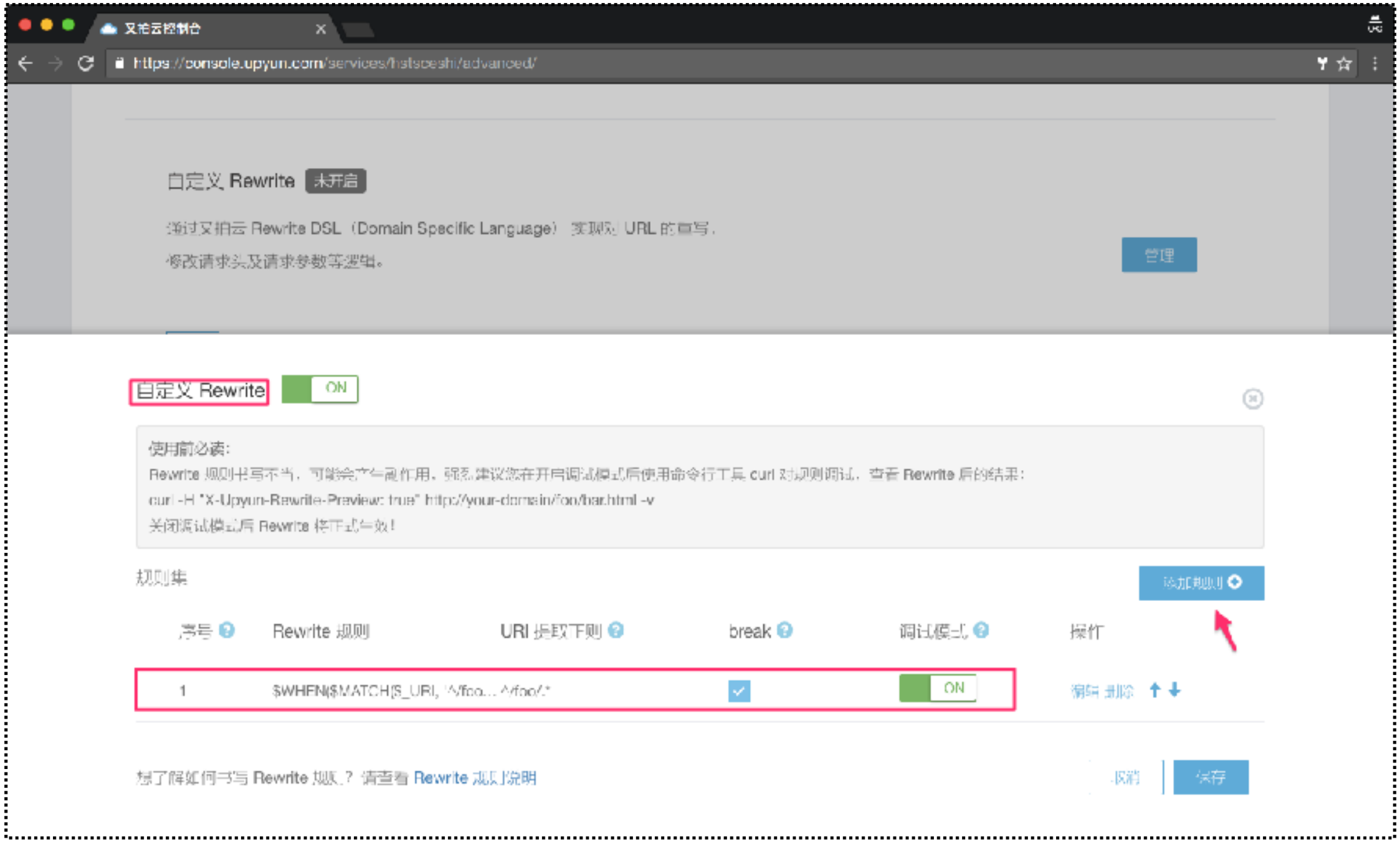
- http://nginx.org/en/docs/http/ngx_http_rewrite_module.html

Example:

```
server {
    ...
    rewrite ^(/download/.*)/media/(.*)\..*$ $1/mp3/$2.mp3 last;
    rewrite ^(/download/.*)/audio/(.*)\..*$ $1/mp3/$2.ra  last;
    return  403;
    ...
}
```

But if these directives are put inside the "`/download/`" location, the `last` flag should be replaced by `break`, or otherwise nginx will make 10 cycles and return the 500 error:

```
location /download/ {
    rewrite ^(/download/.*)/media/(.*)\..*$ $1/mp3/$2.mp3 break;
    rewrite ^(/download/.*)/audio/(.*)\..*$ $1/mp3/$2.ra  break;
    return  403;
}
```



服务 > 功能配置 > 高级功能 > **自定义 Rewrite**

规则格式说明


自定义 Rewrite
Rewrite 规则
$WHEN($MATCH($_URI, '^/add.*'))$EXIT(200, $ADD($_GET_a, $_GET_b))
URI 提取正则
可不填，最多支持输入 256 个字符。
break
调试模式
ON
取消
保存

可编程的 CDN:

变量部分

$_METHOD

$_SCHEME

$_IP

$_HOST

$_POST_name

$_SYM_sym

$_HEADER_name

$_HOST_n

$_COOKIE_name

$_URI

$_GET_name

$_RANDOM_n

$_RANDOM

$_QUERY

. . .

变量详解

- $_GET_name - 使用 Query String 中的变量（无需 urldecode）

```
rule: $WHEN($MATCH($_URI,'/echo_var'))$EXIT(200, $_GET_foo)
```

```
test:

$ http http://httpdump.b0.upaiyun.com/echo_var?foo=bar
HTTP/1.1 200 OK
Connection: keep-alive
Content-Type: application/octet-stream
Date: Wed, 07 Dec 2016 06:44:58 GMT
Server: marco/0.20
Transfer-Encoding: chunked
Via: M.ctn-zj-lna2-005
X-Cache: Unknown from ctn-zj-lna2-005
X-Request-Id: 3ebc2760275f4a03c4294cfdaafeeb0b

bar
```

可编程的 CDN:

函数部分

$ENCODE_BASE64(E)

$ALL(E1, E2, ...)

$UPPER(E)

$DECODE_BASE64(E)

$ANY(E1, E2, ...)

$LOWER(E)

$WHEN(E1, E2, ...)

$SUB(E1, from, to)

$PCALL(E)

$MATCH(E1, E2)

$GT(E1, E2)

$GE(E1, E2)

$EQ(E1, E2)

$ADD_REQ_HEADER(E1, E2)

$DEL_REQ_HEADER(E1)

$ADD_RSP_HEADER(E1, E2)

. . .

函数详解：分类

- 条件选择和判断，例如 **$WHEN(E1, E2, ...)**
- 字符匹配和捕获，例如 **$MATCH(E1, E2)**
- 请求/响应修改，例如 **$ADD_REQ_HEADER(E1, E2)**
- 数值计算、字符串操作等一些功能类函数，例如 **$MD5(E)**
- 限速相关，例如 **$LIMIT_RATE(E1, E2)**

函数详解：限速相关

```
rule: $WHEN($MATCH($_URI, '/limit'))/$LIMIT_RATE_AFTER(20, k)
$LIMIT_RATE($_GET_rate, k)/bytes/102400
```

```
test:

$ wget 'http://httpdump.b0.upaiyun.com/limit?rate=30'

limit?rate=30                       100%
[=================================================================>] 100.00K
50.8KB/s    in 2.0s

2016-12-07 17:56:05 (50.8 KB/s) - 'limit?rate=30' saved [102400/102400]

$ wget 'http://httpdump.b0.upaiyun.com/limit?rate=2'

limit?rate=2                        100%
[=================================================================>] 100.00K
2.11KB/s    in 40s

2016-12-07 17:56:48 (2.49 KB/s) - 'limit?rate=2' saved [102400/102400]
```

经典案例分析：自定义防盗链

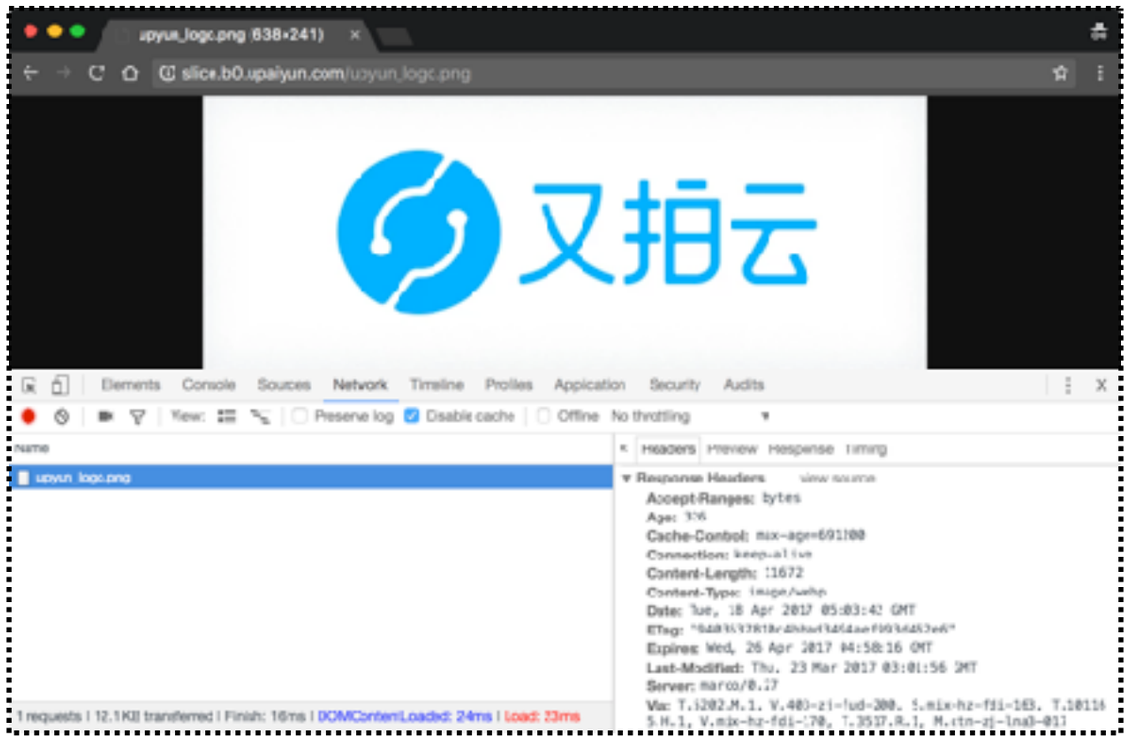
```
token script:

<?php
$etime = time() + 600; // 授权 10 分钟后过期
$key = 'abc'; // token 防盗链密钥
$path = '/test.ts'; // 文件相对路径
$url = $path.'?key='.md5($key.$etime.$path).'&t='.$etime;
echo $url;
?>
```

```
token url:

/test.ts?key=68ddbe535557d6630a19cebde0cb9252&t=1481106349
```

经典案例分析：自定义防盗链

```
rules:

{
  "rule": "$WHEN($MATCH($_URI, '.ts$'),$OR($NOT($_GET_t),$NOT($_GET_key)))
$EXIT(403)",
},
{
  "rule": "$WHEN($MATCH($_URI, '.ts$'),$GT($_TIME, $_GET_t)) $EXIT(401)",
},
{
  "rule": "$WHEN(($MATCH($_URI, '.ts$'),$NOT($EQ($MD5('abc'$_GET_t$_URI),
$LOWER($_GET_key)))) $EXIT(403)",
}
```

经典案例分析：自动 WebP




```
rules:

{
  "rule": "$WHEN($MATCH($_URI,’.jpg|.png$'))$ADD_RSP_HEADER(Vary,Accept)",
},
{
  "rule": "$WHEN($MATCH($_URI,'.jpg|.png$'),$MATCH($_HEADER_accept,'image/webp'))$_URI!webp",
}
```

又拍云 CDN 常规功能**特性增强**

- 更实用的 CC 防护
- 更好用的缓存配置
- 更智能的源站管理
- 更完善的目录刷新
- 更多样的访问控制：地区访问限制、回源鉴权、IP 访问限制
- 以及更多贴心功能：重定向优化、**DNS 纠正**等

功能类别	功能名称
服务管理	管理方式 | API 管理 | 域名管理 | 回源配置
内容管理	缓存管理 | 内容刷新 | URL 预热 | 参数跟随
访问控制	黑白名单 | 防盗链 | WAF 防护 | CC 防护 | 请求大小限制
统计分析	带宽统计 | 流量统计 | 请求数统计 | 导出历史记录
性能监控	健康度 | 缓存命中率 | 平均下载速度 | 拦截攻击次数
高级服务	自定义 SSL 服务 | 视频拖拉 | 镜像存储 | CORS 跨域共享
日志管理	日志下载 | 日志分析
辅助工具	IP检测工具 | 网络诊断工具

又拍云 CDN 关键技术介绍

- 自定义 SSL 服务
- 自主研发 DNS 系统
- CDN 文件分段缓存
- CDN 支持简单编程
- CDN 常规功能特性
- **CDN 实时日志分析**
- **CDN 技术架构演进**

全网 CDN **日志规模**有多大?

保守估计每天约几十 TB 原始日志。

日志的**多种用途**

- 日志每日归档；按不同服务名称提供下载。
- **日志实时分析**；供近实时多维分析、问题排查和监控报警。
- **日志聚合计算**；数据结果提前根据需求确定，实时后台展示。
- 日志离线分析；复杂的数据模型计算和分析，定期生成报表。

日志收集常见的几种方式

```
access_log /path/to/access.log combined buffer=4096 flush=5s;
```

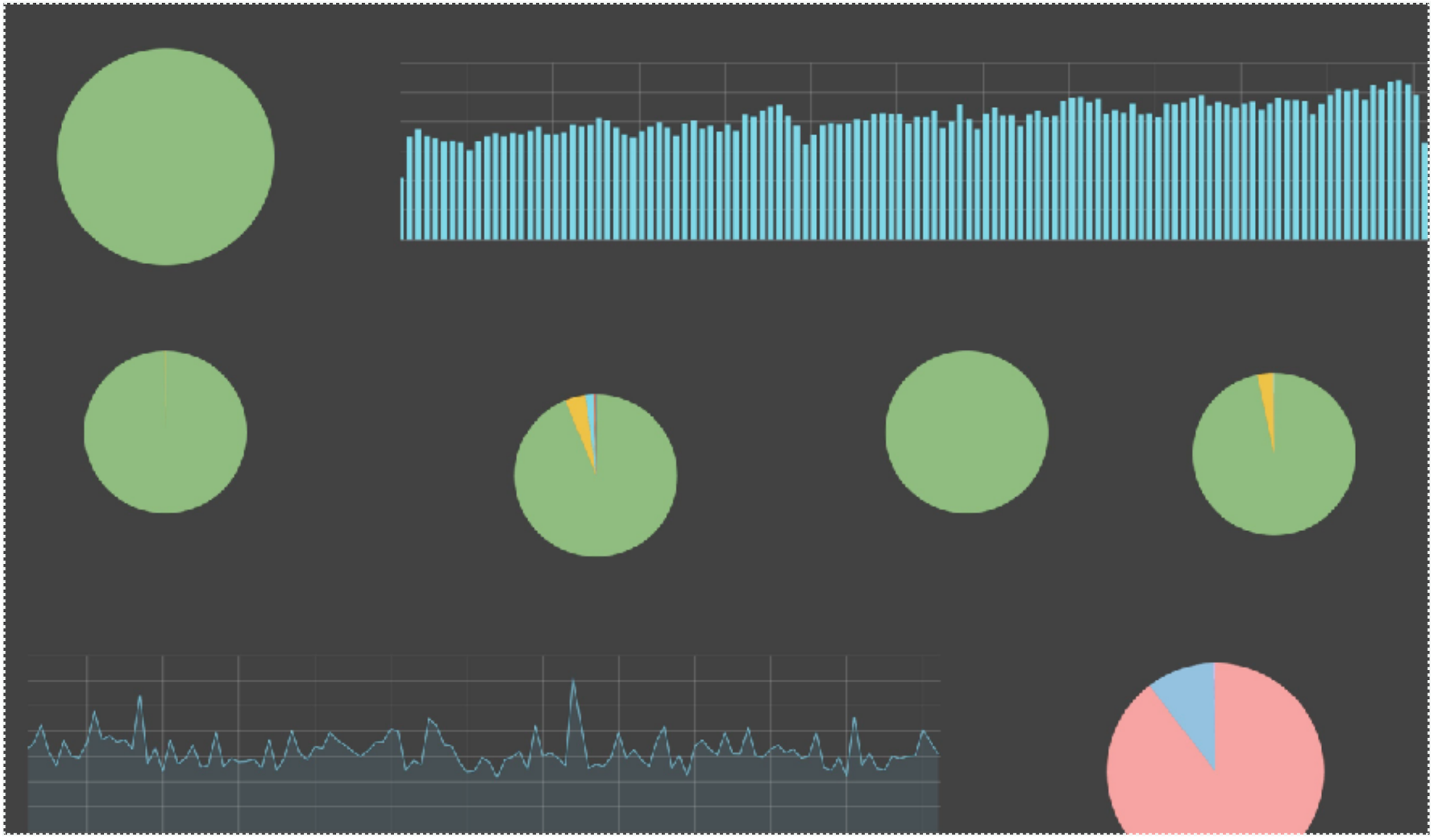
- 传统运维脚本：日志文件**定时**或**定量**切割上报
- **Logstash** / **Heka** Agent：Input File 模块（tail -f）
- **NGINX** 1.7.1+： Logging to **syslog**
- 更多选择？以及日志一定要直接落到**磁盘**吗？

关于**日志实时分析**

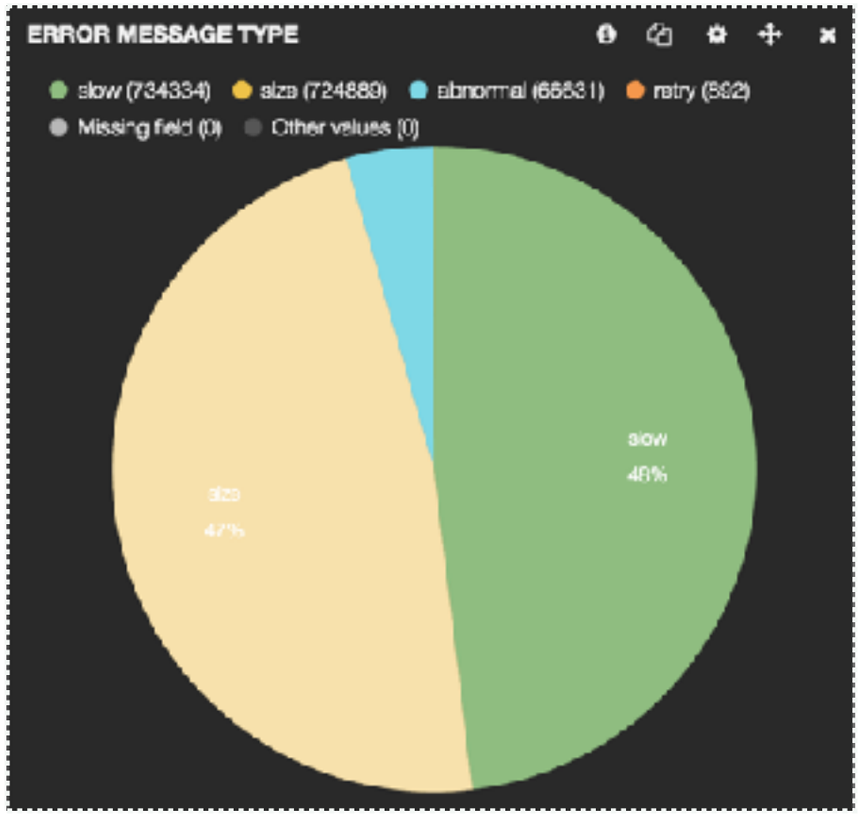
- 很多场景中，不需要全部数据都导入进来，**按需分析即可**。
- 目前分析集群（**ELK 架构**）规模约 30 台，资源有限。
- 需要一套**控制规则**来决定当前我们需要获取哪些日志数据。
- **老板说**：我想**现在**抽样看下某个大客户的实时访问情况。
- 当然，我们默认还是收集了全量的**回源日志**和边缘节点的**非健康日志**。

UPYUN LOG Platform:

HAProxy + Heka + Kafka + Elasticsearch + Kibana

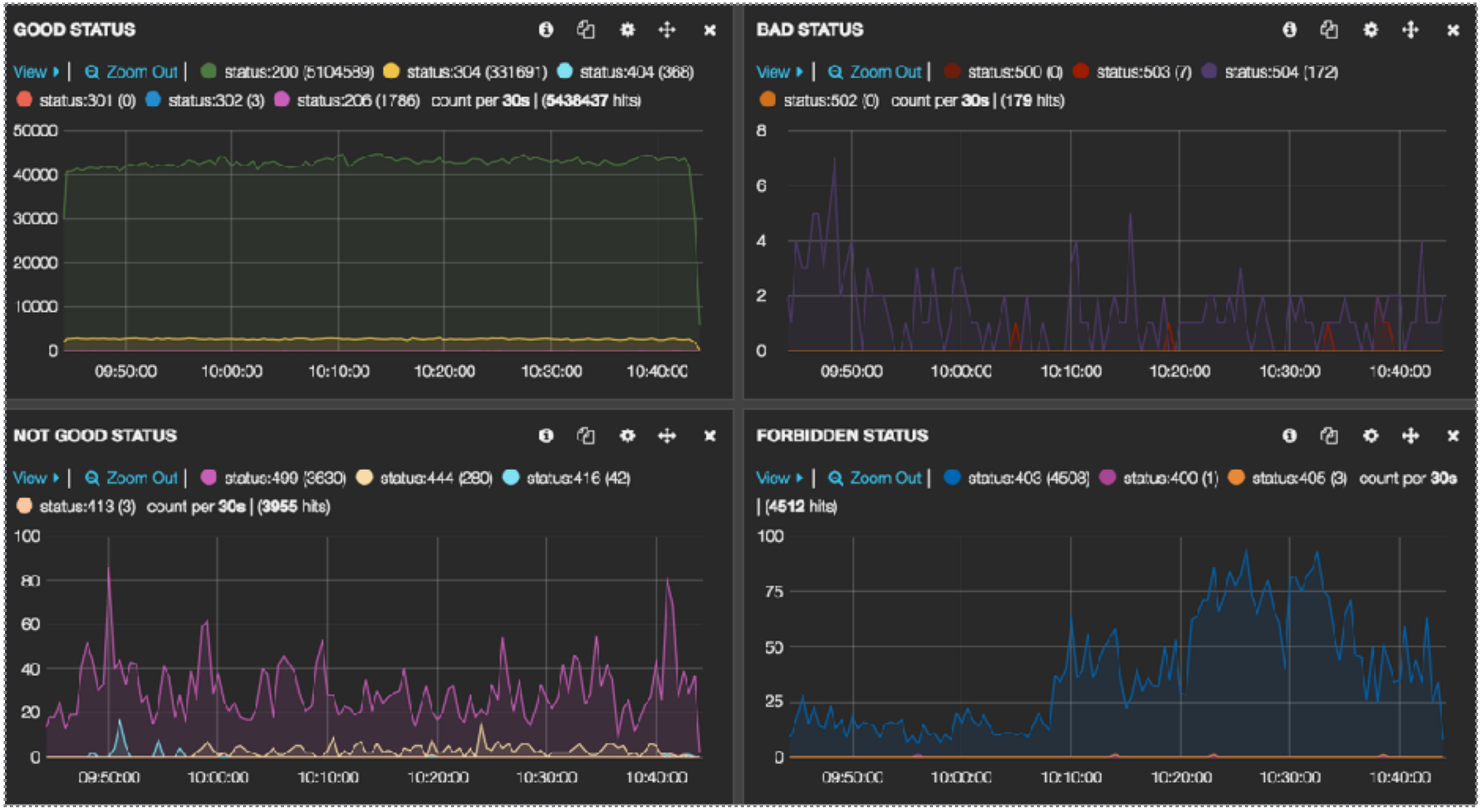


非健康日志有哪些

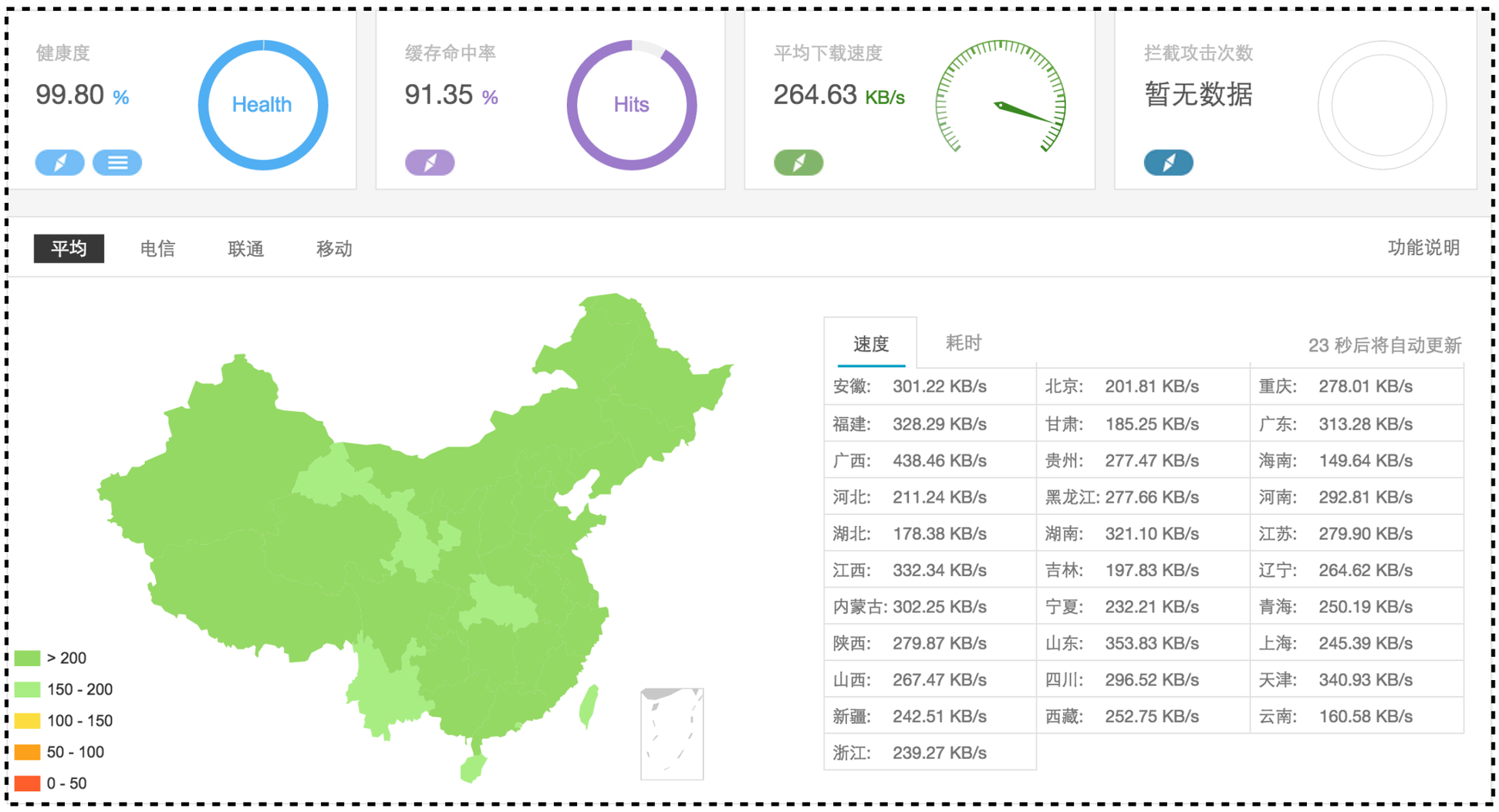


- **show** - 表示速率小于 **50kb/s** 的请求。
- **size** - 表示发送出去的字节数和文件 **Content-Length** 不一致。
- **retry** - 表示在后端节点重试了至少一次。
- **abnormal** - 表示非正常响应，这里排除客户源站返回的部分。

响应状态码分类


GOOD STATUS
View | Zoom Out | status:200 (5104589) status:304 (331691) status:404 (368)
status:301 (0) status:302 (3) status:206 (1786) count per 30s | (5436437 hits)
50000
40000
30000
20000
10000
0
09:50:00
10:00:00
10:10:00
10:20:00
10:30:00
10:40:00
BAD STATUS
View | Zoom Out | status:500 (0) status:503 (7) status:504 (172)
status:502 (0) count per 30s | (179 hits)
8
6
4
2
0
09:50:00
10:00:00
10:10:00
10:20:00
10:30:00
10:40:00
NOT GOOD STATUS
View | Zoom Out | status:499 (3630) status:444 (280) status:416 (42)
status:113 (3) count per 30s | (3955 hits)
100
80
60
40
20
0
09:50:00
10:00:00
10:10:00
10:20:00
10:30:00
10:40:00
FORBIDDEN STATUS
View | Zoom Out | status:403 (4508) status:400 (1) status:405 (3) count per 30s
| (4512 hits)
100
75
50
25
0
09:50:00
10:00:00
10:10:00
10:20:00
10:30:00
10:40:00

CDN 实时聚合系统


健康度
99.80 %
Health
缓存命中率
91.35 %
Hits
平均下载速度
264.63 KB/s
拦截攻击次数
暂无数据
平均
电信
联通
移动
功能说明
> 200
150 - 200
100 - 150
50 - 100
0 - 50
速度
耗时
23 秒后将自动更新
安徽: 301.22 KB/s
北京: 201.81 KB/s
重庆: 278.01 KB/s
福建: 328.29 KB/s
甘肃: 185.25 KB/s
广东: 313.28 KB/s
广西: 438.46 KB/s
贵州: 277.47 KB/s
海南: 149.64 KB/s
河北: 211.24 KB/s
黑龙江: 277.66 KB/s
河南: 292.81 KB/s
湖北: 178.38 KB/s
湖南: 321.10 KB/s
江苏: 279.90 KB/s
江西: 332.34 KB/s
吉林: 197.83 KB/s
辽宁: 264.62 KB/s
内蒙古: 302.25 KB/s
宁夏: 232.21 KB/s
青海: 250.19 KB/s
陕西: 279.87 KB/s
山东: 353.83 KB/s
上海: 245.39 KB/s
山西: 267.47 KB/s
四川: 296.52 KB/s
天津: 340.93 KB/s
新疆: 242.51 KB/s
西藏: 252.75 KB/s
云南: 160.58 KB/s
浙江: 239.27 KB/s

关于**日志聚合计算**

- 全量收集 CDN 边缘请求指标，**实时上报、聚合计算、查询**。
- 支持**多种时间维度**的数据展示（分钟，5分钟，小时，天）。
- 支持域名、服务名称、账号名查询。
- 支持列表、TopN、数据汇总等**复杂查询**请求。

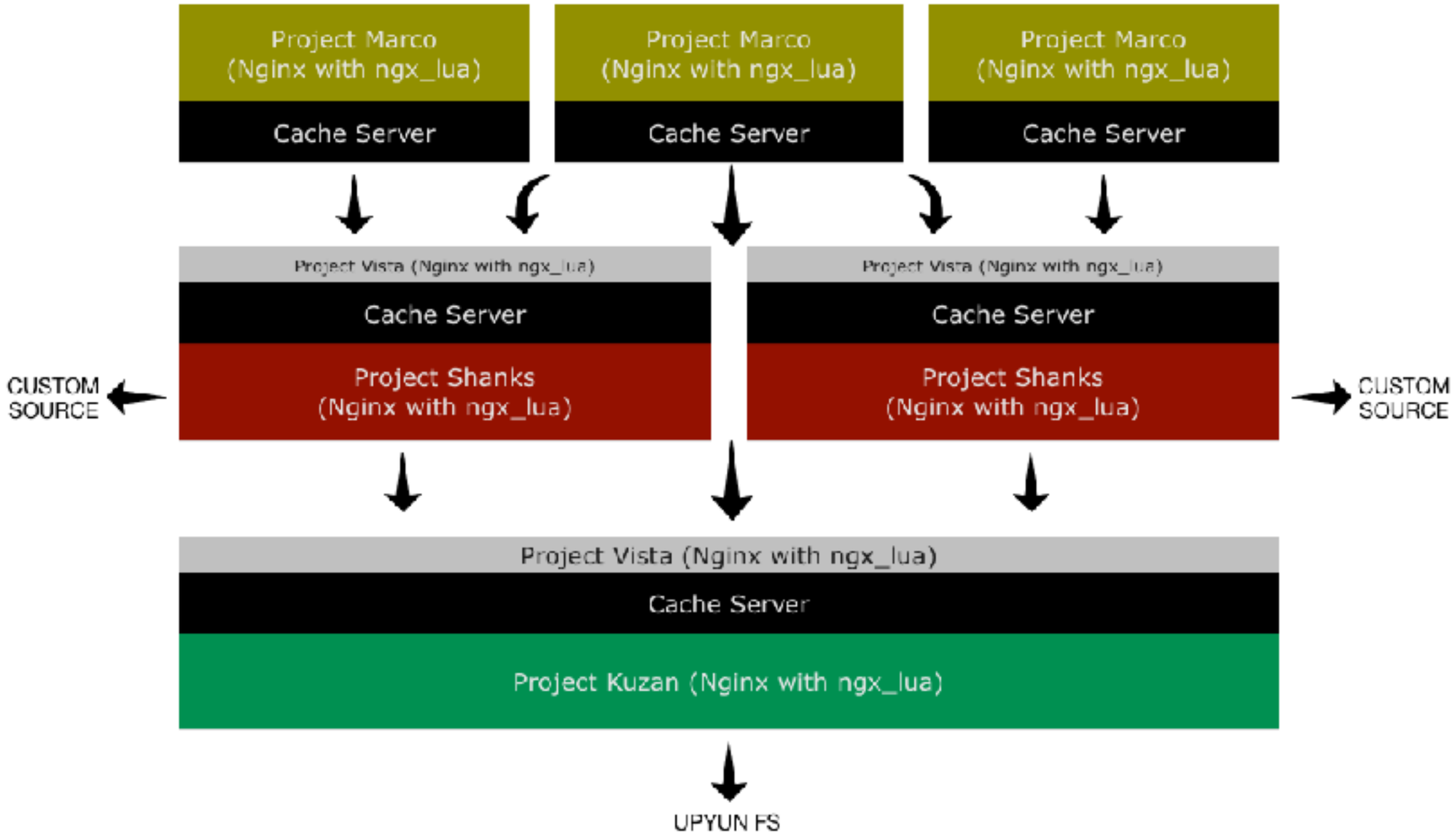
有了**日志数据**，就可以指导我们把系统**做得更好**。

一些有助于定位**请求来源**的字段

- **x-request-id**: ce4fc776afd2b74175695d239b67e3ed
- **via**: T.2428.**H**.1, V.mix-gd-can-007, T.2415.**R**.1, M.cun-ha-cgo-005
- **x-source**: C/200

又拍云 CDN 关键技术介绍

- 自定义 SSL 服务
- 自主研发 DNS 系统
- CDN 文件分段缓存
- CDN 支持简单编程
- CDN 常规功能特性
- CDN 实时日志分析
- **CDN 技术架构演进**



又拍云 CDN 架构

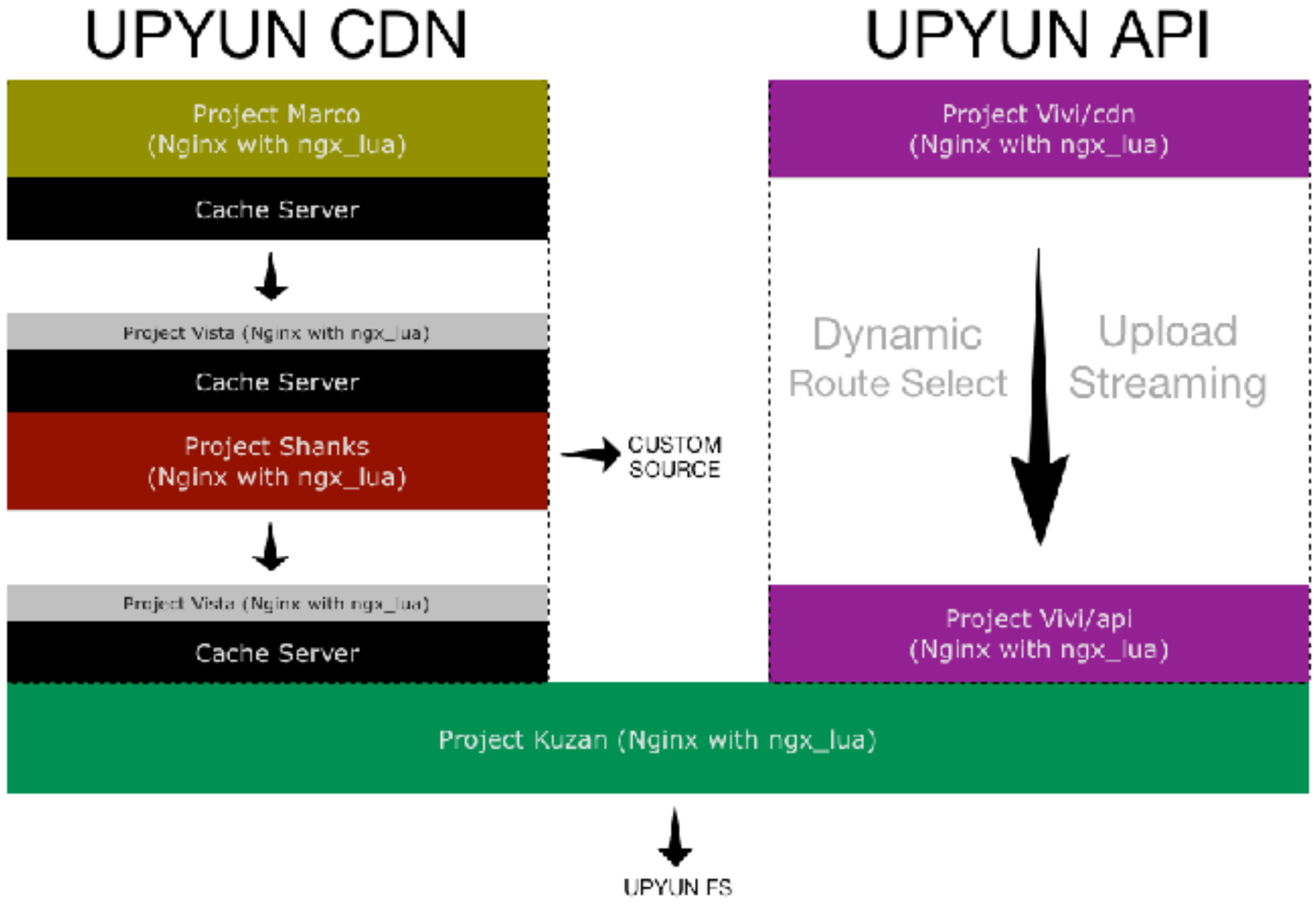
130+ 边缘节点、多个中心三线大机房、同时与云存储无缝融合

云 CDN 时代，开通个功能，是否还需要走工单?

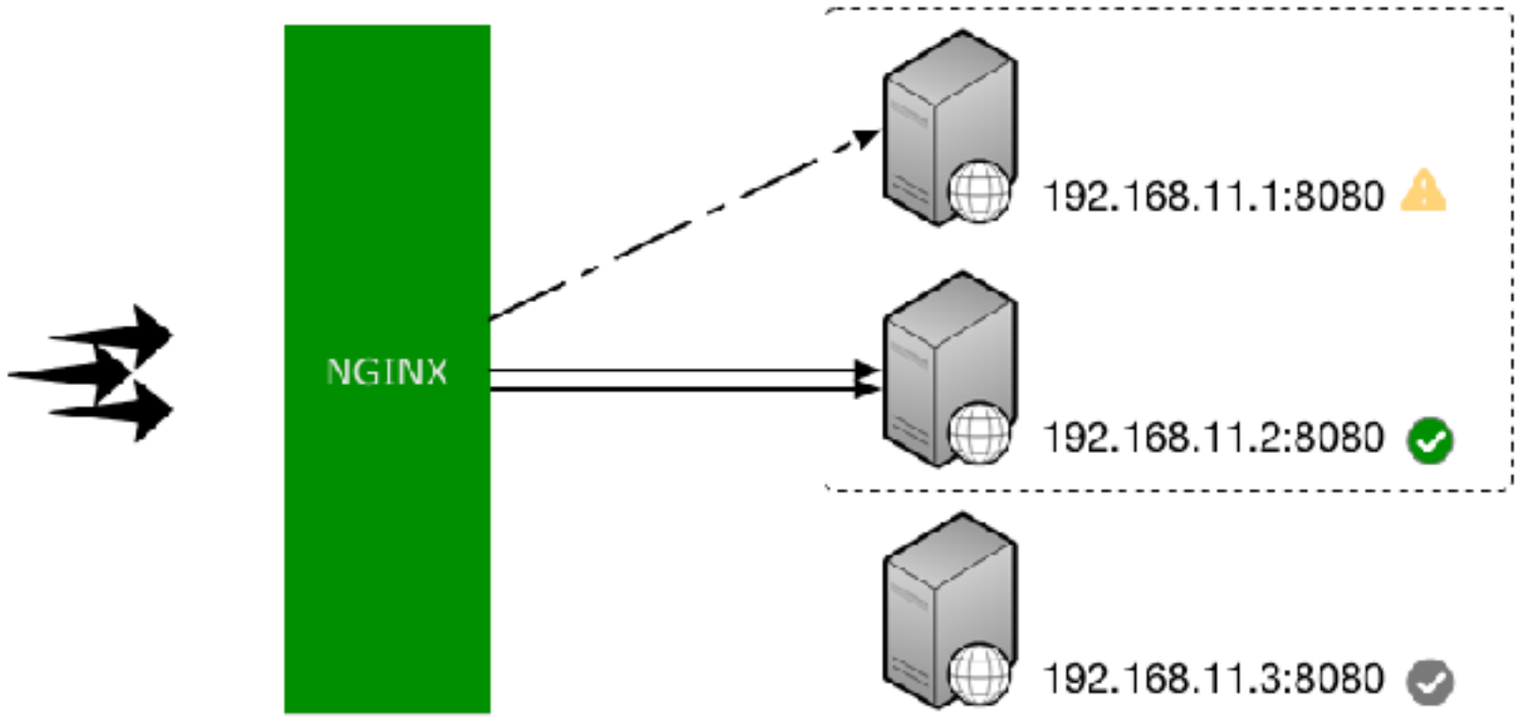
nginx.conf	service
server_name *.b0.upaiyun.com	Custom Domain Binding
valid_referers, **allow, deny**	**Custom Antileech Rules and Redirect:** ip, user-agent, referer, token etc.
expires 7d	**Custom Cache Control:** support specific URI rules etc.
ssl_certificate* **ssl_stapling***	Custom SSL
upstream { server 127.0.0.1 }	Custom CDN Origin: support multi-network routing etc.
max_fails=3 fail_timeout=30s **health_check (*)**	Custom Health Check Strategy: passive, active
round-robin, ip_hash, **hash (1.7.2+)**	Custom Load Balancing Strategy
rewrite	Custom URL rewrite
...	...

nginx.conf as a service

powered by ngx_lua



UPYUN CDN & API is built on top of NGINX with ngx_lua

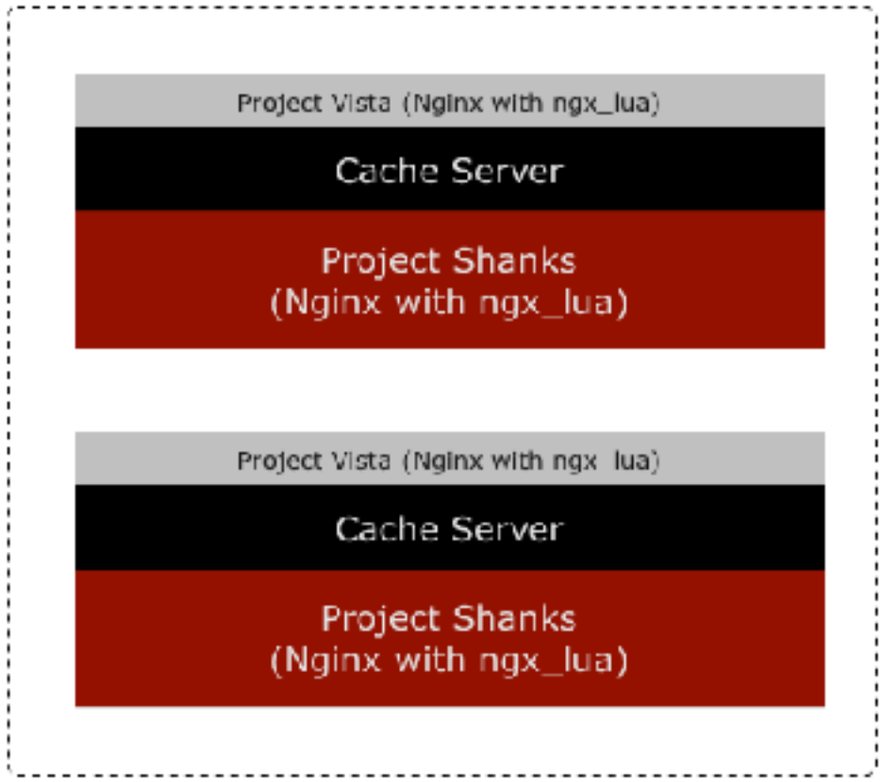



```
upstream blog.upyun.com {
    server 192.168.11.1:8080 weight=1 max_fails=10 fail_timeout=30s;
    server 192.168.11.2:8080 weight=2 max_fails=10 fail_timeout=30s;

    server 192.168.11.3:8080 weight=1 max_fails=10 fail_timeout=30s backup;

    proxy_next_upstream error timeout http_500;
    proxy_next_upstream_tries 2;
}
```

Lua Upstream Dynamically:

Configure Everything as **JSON**

```
{
    "bucket:upblog": [
        {
            "fail_timeout": 30,
            "host": "192.168.11.1",
            "max_fails": 3,
            "port": 8080,
            "weight": 1
        },
        {
            "fail_timeout": 30,
            "host": "192.168.11.2",
            "max_fails": 3,
            "port": 8080,
            "weight": 2
        },
        {
            "backup": true,
            "fail_timeout": 30,
            "host": "192.168.11.3",
            "max_fails": 3,
            "port": 8080,
            "weight": 1
        }
    ]
}
```



master

slave

回源管理

回源 Host：blog.upyun.com

- 可选配置项，可自定义域名，如不填则表示使用加速域名进行回源访问

回源方式：HTTP 协议回源　HTTPS 协议回源　协议跟随

线路配置

回源地址	端口号	主/备	轮询权重	最大失败次数	静默时间(秒)
192.168.11.1:8080	8080	主线路	1	10	30
192.168.11.2:8080	8080	主线路	2	10	30
192.168.11.3:8080	8080	备用线路	1	10	30

- 最大失败次数表示在静默时间内允许回源失败的最大次数，默认为 3 次

自适应 WebP

自适应 H.265

全网流量调度

智能区域回源

弹性处理集群

数据分析预测

...

Q & A

“欢迎体验又拍云”

又拍云