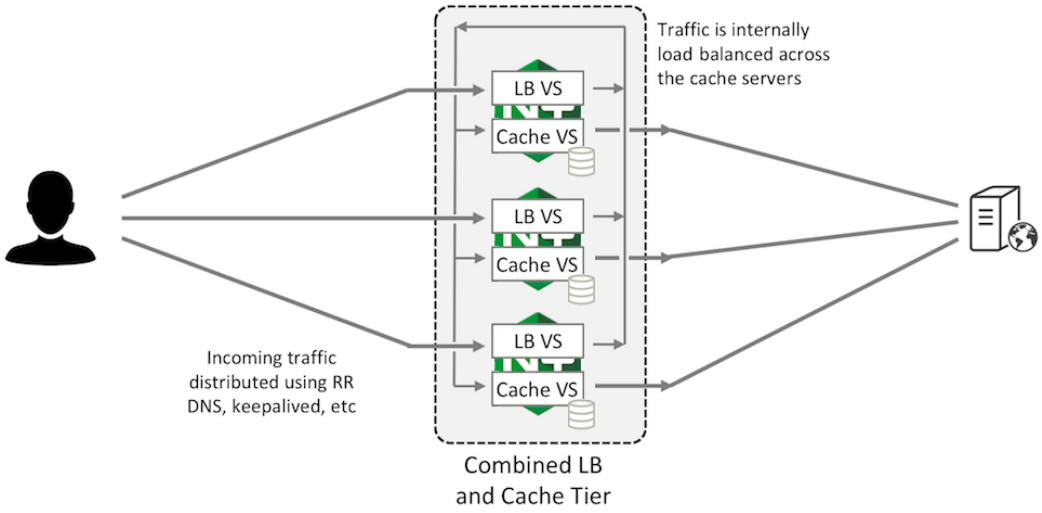
一、 整体架构



- 利用多台 GCE，每一台 GCE 运行两个 nginx 实例（一个 Nginx 开启二个不同的端口）， 80 端口负责流量的分发， 8080 负责缓存管理
- Nginx 的左边是 Google GFE，右边是源站（AWS S3 或客户的源站）
- Nginx 实例部署在靠近源站的区域
- 分区域部署方式
    - Zone1 部署一套带缓存的集群，根据客户流量部署服务器，一般至少三台
    - Zone2 部署一套不带缓存纯代理的当成备用机器，平时只接 1%的流量（直接回源到 S3，不回源到二级缓存），Zone1 如果挂了纯代理所有流量（通过 gclb 的调度机制可以做到流量的分配）； 服务器配置为 4 核 8G，至少部署两台，并且通过实例组实现自动扩展。此外，在大流量的域名切换的初期，此集群还能起到分流 cache 节点无法处理的请求的作用。

二、方案效果

某客户使用二级缓存方案前，CDN 命中率 98% 以上，但由于 CDN 量非常大，回源 AWS 源站流量一直保持在几个 GB 级别

使用二级缓存方案后，CDN 命中率不变，但是，所有回源请求都缓存在二级缓存磁盘中，只有第一次请求会回客户源站一次，后续都不再回客户源。目前此客户的回源流量稳定在 10MB/s 以下。

三、 Nginx 配置样例

```
user    www-data;
worker_processes    auto;
worker_rlimit_nofile 1048576;

#error_log    logs/error.log;
#error_log    logs/error.log    notice;
#error_log    logs/error.log    info;

#pid              logs/nginx.pid;


events {
      worker_connections    81920;
      multi_accept on;
}


http {
      include             mime.types;
      default_type    application/octet-stream;
      server_names_hash_bucket_size 96;
      send_timeout                  15s;



      proxy_cache_path      /var/cache/nginx      levels=1:2      keys_zone=mem-cache:512m
inactive=365d max_size=5500g use_temp_path=off;
      proxy_temp_path /var/cache/nginx/tmp;
      proxy_max_temp_file_size 2M;

      resolver 169.254.169.254 ipv6=off;

      upstream cache {
         hash $uri consistent;
         server 10.148.0.9:8080;
         server 10.148.0.12:8080;
      }


      # add default server here
```

```
server {
   listen 80 reuseport;

   server_name ~^(.+)$;

   location / {
         proxy_buffering off;
         proxy_set_header Connection "";
         proxy_http_version 1.1;
         proxy_set_header Host $http_host;
         proxy_pass http://cache;
   }
}


server {
      listen 8080 reuseport;

      server_name xxxx.yyyyyy.com;

      resolver 8.8.8.8;
      set $upstream_endpoint http://cdodl.xx.com.s3.amazonaws.com;

      location / {
      proxy_cache mem-cache;
      proxy_buffering on;

      proxy_set_header Host cdodl.xxx.com.s3.amazonaws.com;

      add_header X-Proxy-Cache $upstream_cache_status;

      add_header Cache-Control "public, max-age=25920001";
      proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
      proxy_pass $upstream_endpoint$request_uri;
      slice                       2097136;
      proxy_cache_key       $uri$slice_range;
      proxy_set_header      Range $slice_range;
      proxy_cache_revalidate on;
      proxy_cache_valid           200 206 301 302 304 7d;
      proxy_cache_lock           on;
      proxy_cache_lock_timeout    5s;

      }
}
```