负载均衡

实践教程

腾讯云

版权所有：腾讯云计算（北京）有限责任公司

【版权声明】

©2013-2025 腾讯云版权所有

本文档（含所有文字、数据、图片等内容）完整的著作权归腾讯云计算（北京）有限责任公司单独所有，未经腾讯云事先明确书面许可，任何主体不得以任何形式复制、修改、使用、抄袭、传播本文档全部或部分内容。前述行为构成对腾讯云著作权的侵犯，腾讯云将依法采取措施追究法律责任。

【商标声明】

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。未经腾讯云及有关权利人书面许可，任何主体不得以任何方式对前述商标进行使用、复制、修改、传播、抄录等行为，否则将构成对腾讯云及有关权利人商标权的侵犯，腾讯云将依法采取措施追究法律责任。

【服务声明】

本文档意在向您介绍腾讯云全部或部分产品、服务的当时的相关概况，部分产品、服务的内容可能不时有所调整。
您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或默示的承诺或保证。

【联系我们】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100或95716。

文档目录

版权所有：腾讯云计算（北京）有限责任公司

实践教程
负载均衡开启 Gzip 配置及检测方法说明

最近更新时间：2025-05-23 17:06:12

在公网负载均衡、公网固定 IP 型负载均衡实例中，HTTP/HTTPS 协议默认支持用户开启 Gzip 压缩功能。开启 Gzip 功能对网页进行压缩，可以有效降低网络传输的数据量，提升客户端浏览器的访问速度。在使用过程中，需要注意如下事项：

注意事项

- **需要后端 CVM 同步开启 Gzip 支持**
  CLB 默认支持 Gzip 压缩透传模式，后端 CVM 开启 Gzip 压缩后，CLB 可以透传压缩包。对于常见的 Nginx 服务容器，必须在其配置文件（默认为 nginx.conf）中，开启 Gzip 并重启服务。

```
gzip on;
```

- **当前负载均衡支持的文件类型如下，您可以在 Gzip_types 配置项中指定文件类型进行压缩。**

```
application/atom+xml application/javascript application/json application/rss+xml
application/vnd.ms-fontobject application/x-font-ttf application/x-web-app-
manifest+json application/xhtml+xml application/xml font/opentype image/svg+xml
image/x-icon text/css text/plain text/x-component;
```

> ⚠ **注意：**
> 负载均衡后端 CVM 业务软件中必须同步开启对上述文件类型的 Gzip 支持。

- **客户端请求中必须带有压缩请求标记**
  需要启用压缩，还要求客户端请求时必须携带如下标记：

```
Accept-Encoding: gzip,deflate,sdch
```

后端 CVM 开启 Gzip 流程支持示例

示例云服务器运行环境：Debian 6。

1. 使用 vim 依据用户路径打开 Nginx 配置文件：

```
vim /etc/nginx/nginx.conf
```

2. 找到如下代码：

```
gzip on;
gzip_min_length 1k;
gzip_buffers 4 16k;
gzip_http_version 1.1;
gzip_comp_level 2;
gzip_types text/html application/json;
```

上述代码的语法详解：

- gzip：开启或关闭 Gzip 模块。
  语法：`gzip on/off`
  作用域：http，server，location
- gzip_min_length：设置允许压缩的页面最小字节数，页面字节数从 header 头中的 Content-Length 中进行获取。默认值是1k。
  语法：`gzip_min_length length`
  作用域：http，server，location
- gzip_buffers：设置处理 Gzip 压缩文件的缓冲区数量和大小。16k 代表以16k 为单位，按照原始数据大小以16k 为单位的4倍申请内存。
  语法：`gzip_buffers number size`
  作用域：http，server，location
- gzip_http_version ：代表可以使用 Gzip 功能的 HTTP 最低版本，设置 HTTP/1.0 代表了需要使用 Gzip 功能的 HTTP 最低版本，因此可以向上兼容 HTTP/1.1。由于腾讯云现已全网支持 HTTP/1.1，因此无需进行更改。
  语法：`gzip_http_version 1.0 | 1.1;`
  作用域: http，server，location
- gzip_comp_level：Gzip 压缩比，范围为1 - 9。1压缩比最小处理速度最快，9压缩比最大但处理最慢（传输快但比较消耗 cpu ）。
  语法：`gzip_comp_level 1..9`
  作用域: http，server，location
- gzip_types：匹配 MIME 类型进行压缩，默认"text/html" 类型是会被压缩的。 此外，Nginx 下的 Gzip 默认不压缩 javascript、图片等静态资源文件，可以通过gzip_types 指定需要压缩的 MIME 类型，非设置值则不进行压缩。**例如，如果需要对 json 格式数据进行压缩，则需要在此语句中添加 application/json 类型数据。**
  支持的类型如下：

  ```
  text/html text/plain text/css application/x-javascript text/javascript
  application/xml
  ```

  语法：`gzip_types mime-type [mime-type ...]`
  作用域: http, server, location

3. 如对配置有修改，则首先将文件保存退出，进入到 Nginx bin 文件目录，执行如下命令重新加载 Nginx：

```
./nginx -s reload
```

4. 执行以下 curl 命令测试 Gzip 是否成功开启：

```
curl -I -H "Accept-Encoding: gzip, deflate" "http://cloud.tencent.com/example/"
```

- 若命令执行后有返回结果，则表示开启成功。
- 若命令执行后无返回结果，则表示开启失败。

版权所有：腾讯云计算（北京）有限责任公司

部署证书到负载均衡

最近更新时间：2025-05-15 17:46:22

操作场景

本文档指导您将 SSL 证书部署到负载均衡。

前提条件

已登录 证书管理控制台，且成功申请获取证书（参考 如何免费申请域名型证书）。

操作步骤

> **注意：**
> 操作之前，请确认您的 负载均衡控制台 是否有实例，若没有实例，请您先创建实例。

1. 在 我的证书 > 全部 页面单击**已签发**页签，选择您需要部署的证书，并单击**证书信息列**的**证书 ID**。
2. 进入**证书详情**管理页面，在**一键部署证书**模块，单击负载均衡。

> **注意：**
> 目前不支持华南地区-深圳金融。

3. 在弹出的部署证书窗口中，**部署类型**选择**负载均衡**，选择资源实例以及监听器资源。

> **说明：**
> 部署证书的前提是证书支持的域名要和 CLB 绑定的域名一致，如您的负载均衡（CLB）资源未创建监听器资源，可参考 添加监听器 进行操作。

4. 单击**确定**，即可操作成功。如下图所示：

添加监听器

1. 登录 负载均衡控制台，选择您需要配置的监听器，并单击**实例ID**。
2. 进入实例**基本信息**页面，选择**监听器管理**页签。
3. 在 **HTTP/HTTPS 监听器**中单击**新建**，弹出**创建监听器**弹窗。
4. 将**监听协议端口**切换到 HTTPS，服务器证书可选择已有的证书。如下图所示：

> **说明：**
> 此处选择的**服务器证书**为需部署至负载均衡实例的证书，则无需再进行部署至负载均衡操作。

5. 单击**提交**，即可成功配置监听器。

相关文档

管理证书

部署证书到负载均衡（双向认证）

最近更新时间：2025-05-07 15:54:43

操作场景

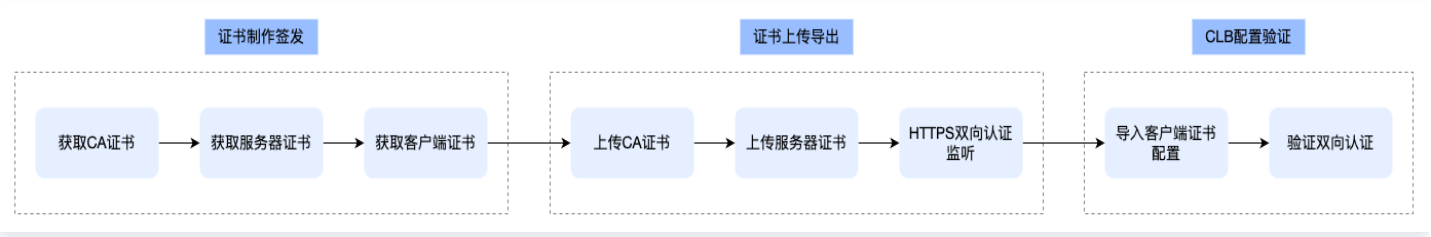
在传统的单向认证里，客户端仅验证服务器的身份，对于一些安全性要求较高的场景，单向认证无法满足安全需。双向认证则要求客户端和服务器双方都验证对方身份，这增强了通信的安全性，能有效防止中间人攻击、身份伪造和数据泄露。关于单向认证与双向认证的说明您可以参考 单向认证和双向认证说明 。

前提条件

1. 您已经创建负载均衡 CLB 实例，具体操作请参考 创建负载均衡实例 。
2. 如需域名访问，请确认您拥有域名并配置负载均衡转发域名，具体操作请参考 配置负载均衡的转发域名 。
3. 您已创建2台 CVM 实例 rs-1与 rs-2，作为 CLB 实例的后端服务器，具体操作请参考 后端服务器 。

配置步骤

下面以自签证书为例，具体流程如图所示，介绍如何配置 HTTPS 双向认证负载均衡。



1. CA 证书：即证书颁发机构证书，您可以用于签发服务器证书或客户端证书。
2. 服务器证书：您可以通过购买或自签方式获取服务器证书，证书购买流程请参考 申请证书 。
3. 客户端证书：您可以使用所获取的 CA 证书签发客户端证书。
4. 上传证书：您需要上传 CA 证书到证书平台，并上传所购买或自签的服务器证书。
5. 负载均衡配置：您在配置 HTTPS 监听器时应开启双向认证，服务器证书选择所上传的证书，CA 证书选择自签根证书。
6. 导入客户端证书：您需要在客户端导入请求时所使用的客户端证书。

步骤1：获取 CA 证书

1. 执行以下命令， 创建 CA 证书的私钥文件 ca.key。

```
# 生成 CA 私钥
openssl genrsa -out ca.key 2048
```

2. 执行以下命令， 创建 CA 证书的请求文件 ca.csr。

```
# 生成 CA 证书请求文件
openssl req -new -key ca.key -out ca.csr
```

> **注意：**
> 请自行填写以下参数，并保证其 Common Name 与服务器证书或者客户端证书的 Common Name 不相同。

```
Country Name (2 letter code) [AU]:sz
State or Province Name (full name) [Some-State]:sz
Locality Name (eg, city) []:sz
Organization Name (eg, company) [Internet Widgits Pty Ltd]:sz
Organizational Unit Name (eg, section) []:sz
Common Name (e.g. server FQDN or YOUR name) []:clb
Email Address []:1.qq.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

3. 执行以下命令，创建自签名的 CA 证书 ca.crt。

```
# 自签名生成 CA 证书，有效期为 3650 天
openssl x509 -req -in ca.csr -out ca.crt -signkey ca.key -days 3650
```

运行成功如下图：

```
Certificate request self-signature ok
subject=C = sz, ST = sz, L = sz, O = sz, OU = sz, CN = clb, emailAddress = 1.qq.com
```

步骤2：获取服务器证书

1. 执行以下命令，创建服务器证书的私钥文件 server.key。

```
# 生成服务器私钥
openssl genrsa -out server.key 2048
```

2. 执行以下命令，创建服务器证书的请求文件 server.csr。

```
# 生成服务器证书请求文件
openssl req -new -key server.key -out server.csr
```

3. 执行以下命令，使用 CA 证书签发服务器证书 server.crt。

```
# 使用 CA 证书签名生成服务器证书，有效期为 365 天
openssl x509 -req -in server.csr -out server.crt -CA ca.crt -CAkey ca.key -
CAcreateserial -days 3650
```

步骤3：获取客户端证书

1. 执行以下命令，创建客户端证书的私钥文件 client.key。

```
# 生成客户端私钥
openssl genrsa -out client.key 2048
```

2. 执行以下命令，创建客户端证书的请求文件 client.csr。

```
# 生成客户端证书请求文件
openssl req -new -key client.key -out client.csr
```

3. 执行以下命令， 使用 CA 证书签发客户端证书 client.crt。

```
# 使用 CA 证书签名生成客户端证书，有效期为 365 天
openssl x509 -req -in client.csr -out client.crt -CA ca.crt -CAkey ca.key -
CAcreateserial -days 3650
```

4. 执行以下命令，将所生成客户端证书 client.crt 转换为浏览器可识别到 p12格式文件。

```
# 客户端证书格式转换
openssl pkcs12 -export -clcerts -in client.crt -inkey client.key -out client.p12
```

步骤4：上传 CA 证书

1. 登录 负载均衡控制台 。
2. 单击**证书管理** ，再单击**新建**。

3. 在上传证书页面，证书类型选择 **CA 证书**，把 步骤1 创建的 CA 证书 ca.crt 的内容复制到**签名证书**区域，单击**确定**。

> **注意：**
> 复制内容时请将最后的换行符删除，避免报错。

步骤5：上传服务器证书

1. 登录 负载均衡控制台 。
2. 单击**证书管理**，再单击**新建**。
3. 在上传证书页面，证书类型选择**服务器证书**，把 步骤2 生成的服务器证书 server.crt 和服务器私钥 server.key 的内容复制到签名证书区域，单击**确定**。

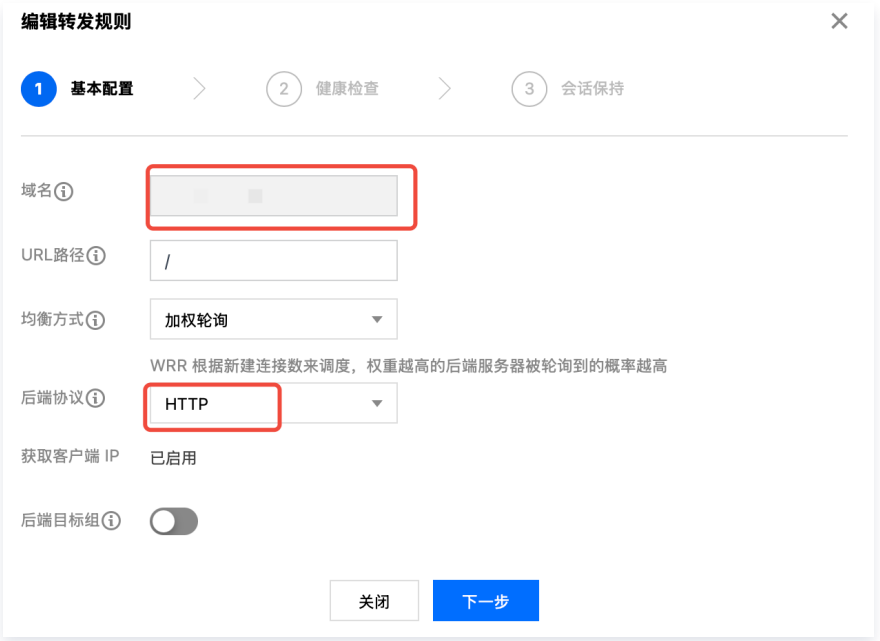
步骤6：配置 HTTPS 双向认证监听

方式一：不启用 SNI

1. 登录 负载均衡控制台；在**实例**页面，找到目标 CLB 实例，单击实例 ID；在**监听器管理**页，单击**新建**。

2. 在配置界面，监听协议选择 HTTPS，并填写指定端口；不启用 SNI 选择双向认证方式；并上传已获取服务器证书与 CA 证书；确认配置信息，然后单击**提交**。

3. 点击加号创建转发规则；填写负载均衡器域名及URL路径；选择均衡方式与后端协议完成基本配置。



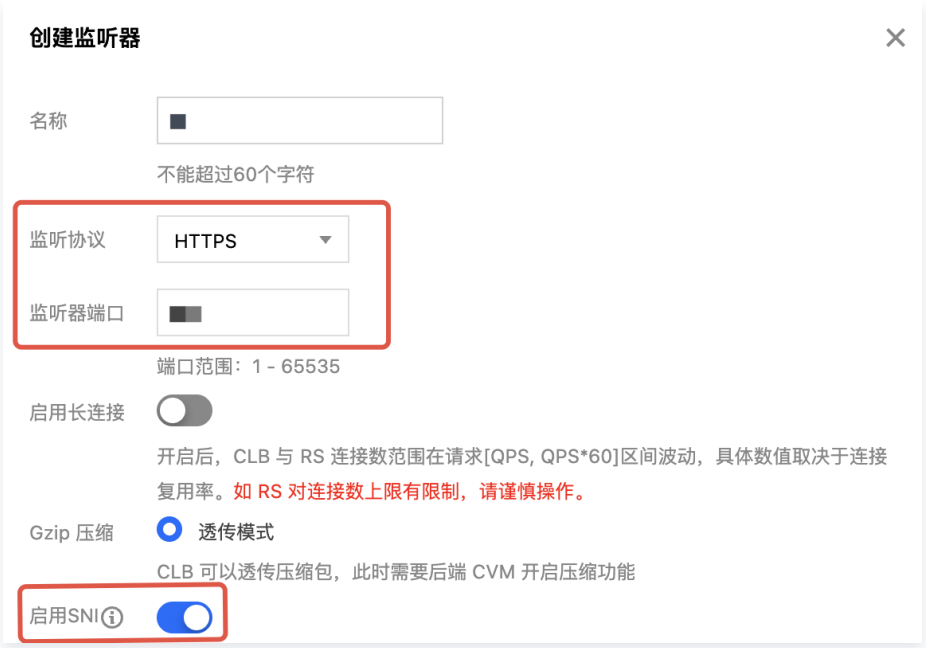
> **说明：**
> 当选用 HTTPS 监听协议时，客户端到负载均衡访问使用 HTTPS；而负载均衡到后端服务器之间转发协议，可选择 HTTP 或 HTTPS。

4. 配置健康检查端口，并根据选择配置会话保持。

5. 展开规则，单击**绑定**，选择所创建的两个后端 rs-1、rs-2。

方式二：启用 SNI

1. 在配置界面，监听协议选择 **HTTPS**，并填写指定端口；启用 SNI 然后单击**提交**。



2. 点击加号创建转发规则，填写负载均衡器域名及 URL 路径，选择均衡方式与后端协议完成基本配置；填写该域名所对应的服务器证书以及自签的 CA 证书。

3. 后续步骤与 不启用SNI 配置方式相同，SNI 具体说明您可参考 CLB 支持 SNI 多域名证书 。

步骤7：导入客户端证书

方式一：浏览器方式

1. 将所签发的客户端证书 client.p12下载导入本地。
2. 双击客户端证书，根据证书导入向导完成客户端证书安装。

方式二：命令行方式

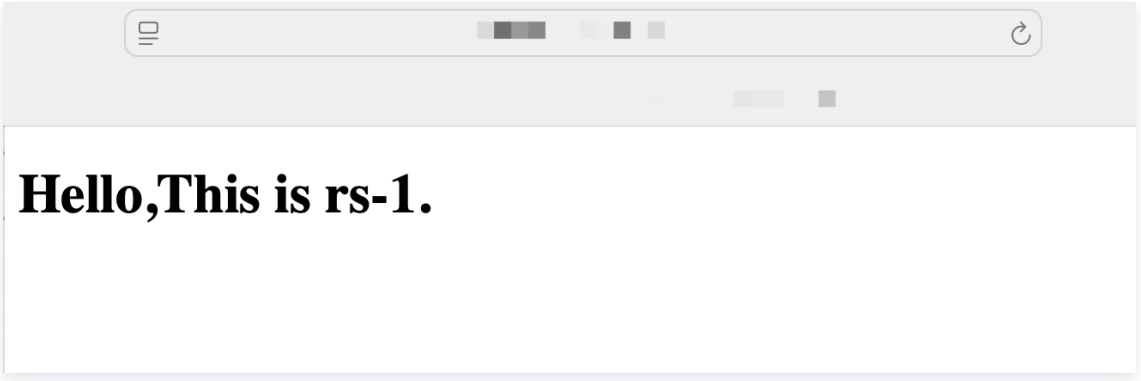
1. 把客户端证书 client.crt 和客户端私钥文件 client.key 拷贝到新目录。
2. 使用指定目录的客户端文件运行命令行验证。

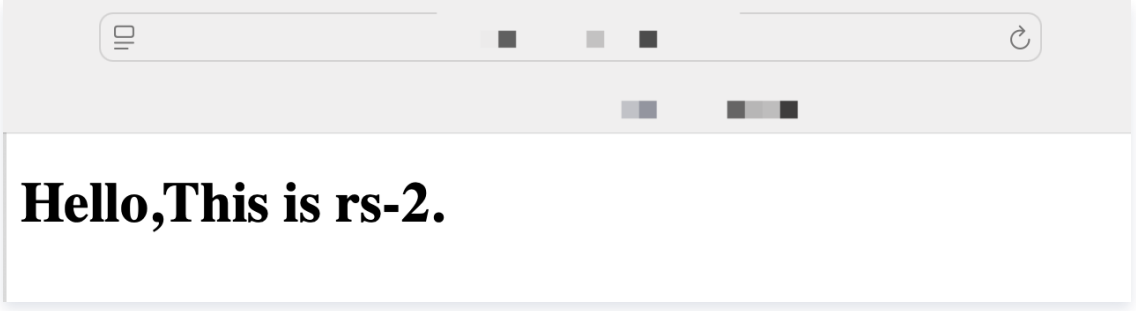
步骤8：验证双向认证

方式一：浏览器方式

1. 在浏览器输入负载均衡绑定域名，未绑定域名可访问 IP 端口，访问时请选择已导入客户端证书。

2. 可刷新浏览器，观察到客户端的请求在 rs-1和 rs-2服务器之间的转换，说明验证成功。





方式二：命令行方式

1. 在 shell 界面，输入以下命令，并确认证书地址和密钥地址，以及所访问的负载均衡器地址。

```
curl --cert client.crt --key client.key --cacert ca.crt https://xxx.xxx.xxx
```

2. 输出对应正确响应码，说明验证成功。

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <!-- ... existing code ... -->
  </head>
  <body>
    <header>
      <h1>Hello,This is rs-1.</h1>
    </header>
    <!-- ... existing code ... -->
  </body>
```

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <!-- ... existing code ... -->
  </head>
  <body>
    <header>
      <h1>Hello,This is rs-2.</h1>
    </header>
    <!-- ... existing code ... -->
  </body>
```

HTTPS 转发配置入门指南

最近更新时间：2025-05-15 17:46:22

1. 负载均衡能力说明

腾讯云 CLB 负载均衡器通过对协议栈及服务端的深度优化，实现了 HTTPS 性能的巨大提升。同时，我们也通过证书的国际合作，极大降低了证书的成本。腾讯云 CLB 在如下几个方面能够为业务带来非常显著的收益：

1. 使用 HTTPS 并不会降低 Client 端的访问速度。
2. 集群内单台服务器 SSL 加解密性能，高达 6.5W cps 的完全握手。相比高性能 CPU 提升了至少3.5倍，节省了服务端成本，极大提升了业务运营及流量突涨时的服务能力，增强了计算型的防攻击能力。
3. 支持多种协议卸载及转换。减少业务适配客户端各种协议的压力，业务后端只需要支持 HTTP1.1 就能使用 HTTP2、SPDY、SSL3.0 及 TLS1.2 等各版本协议。
4. 一站式 SSL 证书申请、监控、替换。我们和国际证书厂商 Comodo，SecureSite 展开对话，探讨合作，大幅缩减证书申请流程及成本。
5. 防 CC 及 WAF 功能。能够有效杜绝慢连接、高频定点攻击、SQL 注入、网页挂马等应用层攻击。

2. HTTP、HTTPS 头部标识

CLB 会对 HTTPS 进行代理，来自客户端的 HTTP 或者 HTTPS 请求，到达 CLB 转发给后端服务器时，CLB 与后端服务之间的协议支持选择 HTTP、HTTPS 或 gRPC，详情请参见 配置 HTTPS 监听器 。CLB 与后端服务之间的协议选择为 HTTP 时，开发者有可能无法分辨前端的请求是 HTTP 还是 HTTPS。

腾讯云 CLB 在将请求转发给后端服务器时，头部 header 会植入 X-Client-Proto：

- X-Client-Proto: http（前端为 HTTP 请求）
- X-Client-Proto: https（前端为 HTTPS 请求）

3. 入门配置

假定开发者希望用户在浏览器中输入网址时，键入 `www.example.com` 即可通过 HTTPS 协议安全访问。

负载均衡 CLB 的操作流程可参考以下文档：

- 创建负载均衡实例
- 配置 HTTP 监听器
- 管理后端服务器

用户输入的 `www.example.com` 请求转发流程如下：

1. 该请求以 HTTP 协议传输，通过 VIP 访问负载均衡监听器的80端口，并被转发到后端服务器的8080端口。
2. 下载 Nginx 后在 nginx/nginx.conf 中配置，通过在腾讯云后端服务器的 Nginx 上配置 rewrite 操作，该请求经过8080端口，并被重写到 `https://www.example.com` 页面。
3. 此时浏览器再次发送 `https://www.example.com` 请求到相应的 HTTPS 站点，该请求通过 VIP 访问负载均衡监听器的443端口，并被转发到后端服务器的80端口。

该操作在浏览器用户未感知的情况下，将用户的 HTTP 请求重写为更加安全的 HTTPS 请求。为实现以上请求转发操作，用户可以对后端服务器做如下配置：

```
server {

    listen 8080;
    server_name www.example.com;

    location / {
```

```
        #! customized_conf_begin;
        client_max_body_size 200m;
        rewrite ^/.(.*) https://$host/$1 redirect;

}
}
```

或者在 Nginx 新版本中，采用推荐的301重定向配置方法，将 Nginx HTTP 页面重定向到 HTTPS 页面：

```
server {
      listen       80;
      server_name    www.example.com;
      return       301 https://$server_name$request_uri;
}

server {
      listen       443 ssl;
     server_name    www.example.com;
    [....]
}
```

说明：您也可以通过负载均衡进行重定向配置，具体请参考 七层重定向配置 。

如何获取客户端真实 IP
后端服务器通过 CLB 获取客户端真实 IP

最近更新时间：2025-06-11 12:21:12

负载均衡获取客户端真实 IP 的说明

CLB 的四层监听器（TCP/UDP/TCP SSL/QUIC）支持直接在后端服务器上获取客户端真实 IP，无需进行额外配置。在默认情况下，后端服务器上获取的源 IP 即为真实的客户端 IP。
但是，当 CLB 和后端服务器之间存在一个或多个 NAT 网关时，后端的服务器无法接受到真实的客户端源 IP。针对此场景，在 CLB 四层监听器上可以开启 Proxy Protocol 配置，主动发起 Proxy Protocol，通过 Proxy Protocol 协议，携带真实的客户端源 IP 给到后端的服务器。

> ⚠ **注意：**
> - 使用该功能需要后端服务器同时开启 Proxy Protocol，这样后端服务器才可以获取到客户端真实的 IP 地址，如果后端服务器不具备解析 Proxy Protocol 协议能力，直接打开特性开关，很可能会导致后端服务解析异常，从而影响服务可用性。
> - 该功能不支持在线平滑迁移，切换到 ProxyProtocol 需要业务停服升级，请谨慎配置。
> - CLB 仅支持 Proxy Protocol v2 版本。Proxy Protocol v2 版本支持多种传输协议，如 TCP 和 UDP，更多信息，请参见 The PROXY protocol 。

功能说明

- 此功能仅标准账户类型支持，传统账户类型不支持，账户类型判断方式请参见 判断账户类型 。
- 仅 IPv4 和 IPv6 实例的 TCP/UDP/TCP SSL/QUIC 监听器支持该功能。
- IPv6 CLB 的TCP/UDP监听器的 Proxy Protocol 配置功能还在灰度中，如有需要请提交 工单申请 。

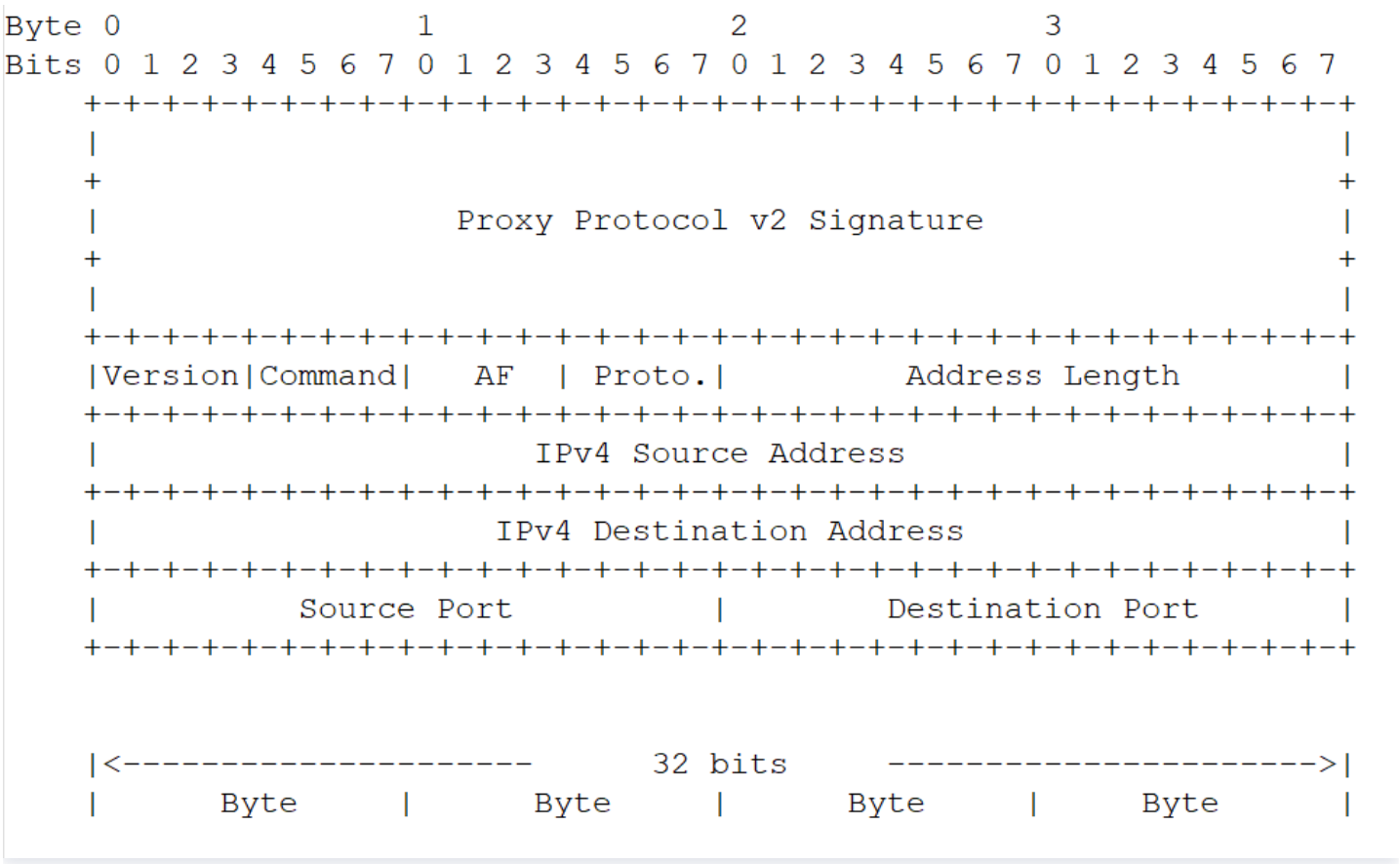
Proxy Protocol 协议说明

使用 Proxy Protocol，代理服务器在转发请求时将客户端的原始网络连接信息封装在请求头部中，发送给后端服务器。后端服务器通过解析 Proxy Protocol 头部，就可以获得客户端的真实网络连接信息，包括源 IP 地址、源端口以及传输协议等。
通过使用 Proxy Protocol，后端服务器可以准确获取客户端的原始网络连接信息，从而进行更准确的日志记录、访问控制、流量监控等操作。

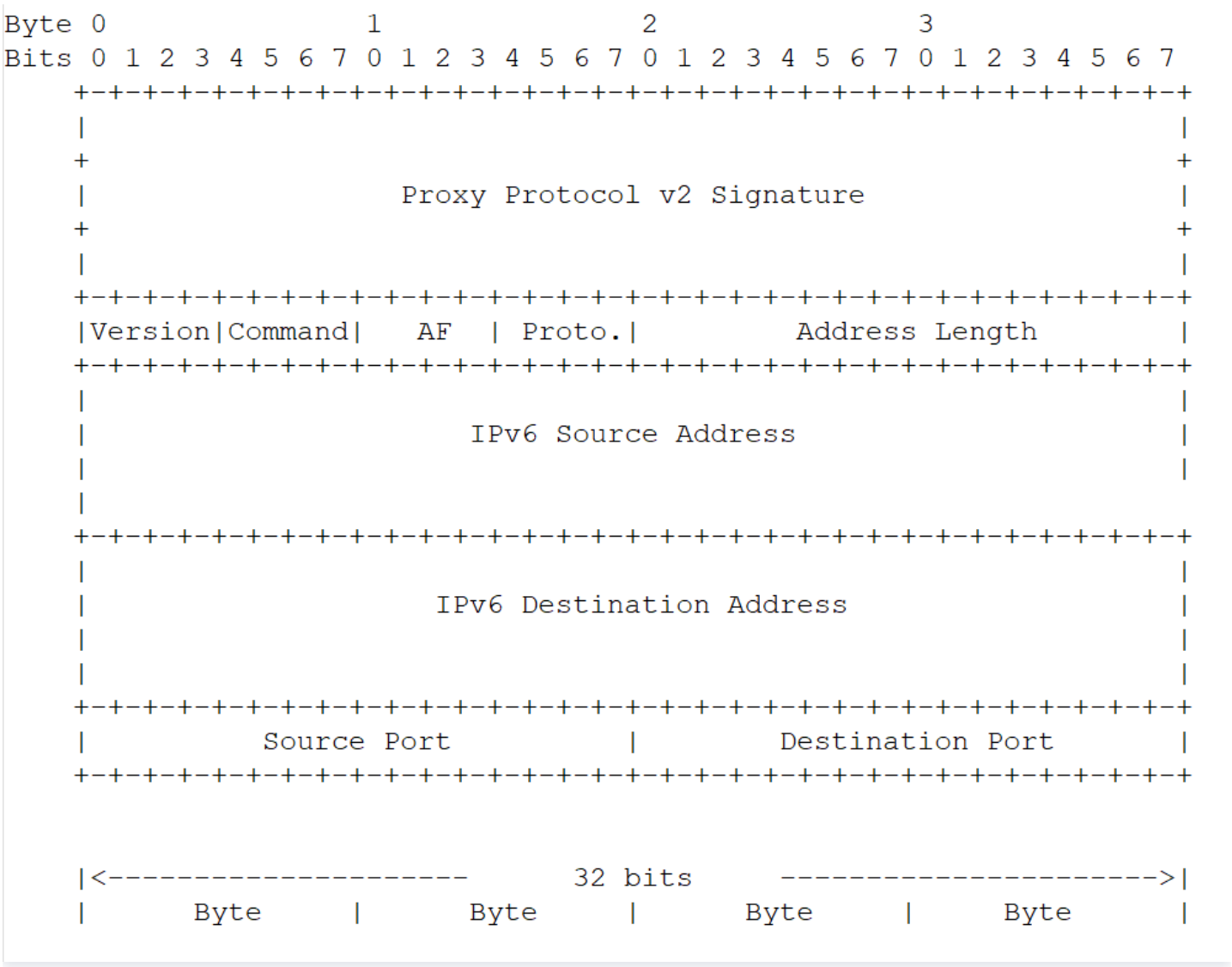
Proxy Protocol V2

Proxy Protocol V2 协议采用二进制格式，支持 TCPv4、TCPv6、UDPv4、UDPv6 协议，其格式如下：

IPv4 格式


Byte 0 1 2 3
Bits 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
Proxy Protocol v2 Signature
Version
Command
AF
Proto.
Address Length
IPv4 Source Address
IPv4 Destination Address
Source Port
Destination Port
32 bits
Byte
Byte
Byte
Byte

IPv6 格式



前提条件

- 启用 Proxy Protocol 之前，请确保您的后端服务器支持 Proxy Protocol v2 版本，否则会导致新建连接失败。
- 本文以 IPv4 CLB 的 TCP 监听器为例进行介绍。

操作步骤

步骤1：为 TCP 监听器打开 Proxy Protocol 配置

1. 登录 负载均衡控制台，在左侧导航栏单击**实例管理**。
2. 在 CLB 实例列表页面左上角选择地域，在实例列表右侧的操作列中单击**配置监听器**。
3. 在 **TCP/UDP/TCP SSL/QUIC 监听器**下，单击目标监听器的详情，查看 ProxyProtocol 配置为**已开启**。若未开启时，请编辑监听器，并在高级选项中，勾选 ProxyProtocol 配置，并提交保存。

步骤2：为后端服务器开启 Proxy Protocol

此处以 CentOS 7.9操作系统、Nginx 1.20.1版本配置为例介绍。具体请以您实际使用的环境为准。

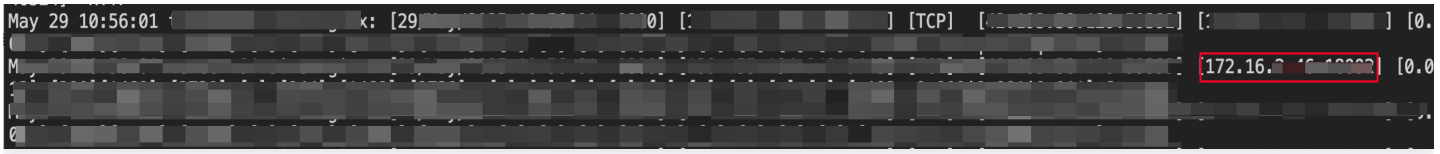
1. 登录后端服务器，执行 nginx -t 命令查看配置文件所在路径。默认通常为 `/etc/nginx/nginx.conf` ，具体请以实际环境为准。
2. 修改配置文件中的 Proxy Protocol 内容并保存，修改点可参考下方说明。

```
http {
# 确保设置$proxy_protocol_addr，该变量用于记录客户端真实IP
log_format  main  '$proxy_protocol_addr - $remote_addr- $remote_user [$time_local]
"$request" '
                  '$status $body_bytes_sent "$http_referer" '
                  '"$http_user_agent" "$http_x_forwarded_for"';
 # 以80监听端口为例，增加proxy_protocol字段
 server {    listen 80   proxy_protocol;
 #...
   }
 }
```

3. 执行 sudo nginx -s reload 命令，重新加载 Nginx 配置文件。

步骤3：验证后端服务器可获取客户端真实 IP

当 Nginx 作为后端服务器时，您可以通过检查 Nginx 日志来判断是否成功获取到了客户端的真实 IP 地址。
Nginx 日志文件默认路径为： `/var/log/nginx/access.log`
每行日志中，$proxy_protocol_addr 变量对应的 IP 地址即为客户端真实IP地址。



IPv4 CLB 场景下获取客户端真实 IP

最近更新时间：2025-05-28 17:38:32

负载均衡获取客户端真实 IP 的说明

CLB 的四层（TCP/UDP/TCP SSL）和七层（HTTP/HTTPS）服务均支持直接在后端 CVM 上获取客户端真实 IP，无需进行额外配置。

- 四层负载均衡，在后端 CVM 上获取的源 IP 即为客户端 IP。
- 七层负载均衡，在 CLB 与后端服务之间使用短连接时，在后端 CVM 上获取的源 IP 即为客户端 IP；在 CLB 与后端服务之间使用长连接时，CLB 不再透传源 IP，您可以通过 X-Forwarded-For 或 remote_addr 字段来直接获取客户端 IP。七层负载均衡的访问日志请参见 配置访问日志到 CLS 。

> **说明：**
> 对于四层负载均衡来说，无需在后端 CVM 上做额外配置即可获取客户端 IP。
> 对于其他做了 SNAT 的七层负载均衡服务，您需要在后端 CVM 上配置，然后使用 X-Forwarded-For 的方式获取客户端的真实 IP。

下文将对常见的应用服务器配置方案进行介绍。

IIS 6 配置方案

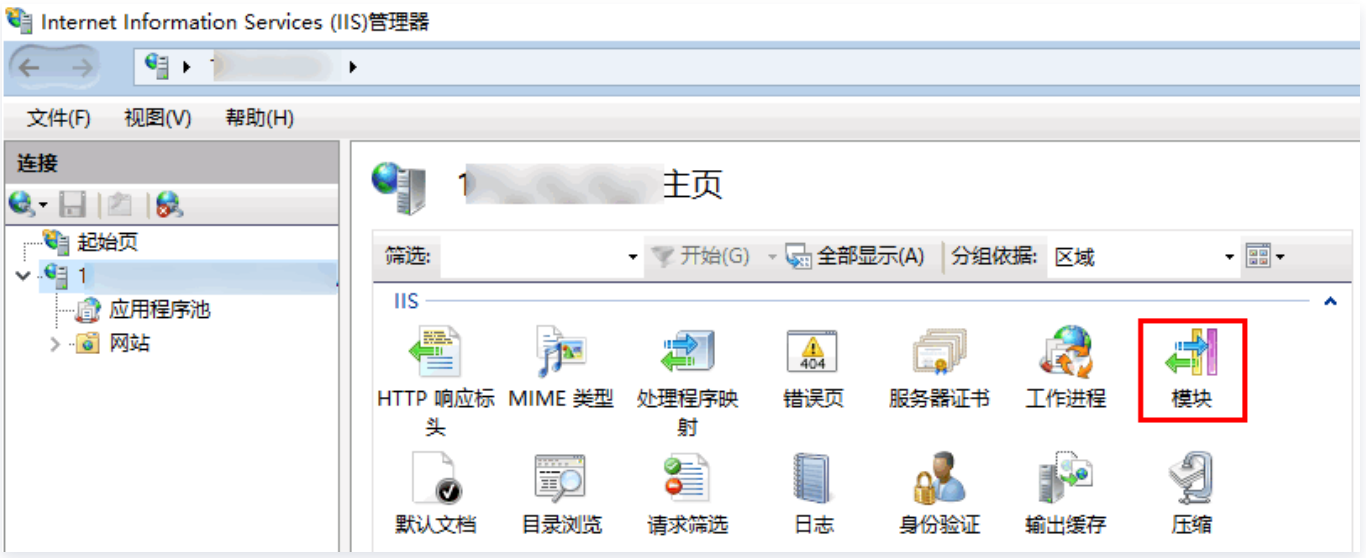
1. 下载与安装插件 **F5XForwardedFor** 模块，根据自己的服务器操作系统版本将 `x86\Release` 或者 `x64\Release` 目录下的 `F5XForwardedFor.dll` 拷贝到某个目录，这里假设为 `C:\ISAPIFilters`，同时确保对 IIS 进程对该目录有读取权限。
2. 打开控制面板，选择**程序** > **程序和功能** > **启用或关闭 Windows 功能** > 勾选 **Internet Information Services**，并确认该目录下**万维网服务** > **应用程序开发功能** > **ISAPI 相关项** 已勾选，单击确认。
3. 打开 **IIS 管理器** > **ISAPI筛选器**，右键选择**添加**，弹出添加窗口。
4. 在添加窗口“筛选器名称”中填写“F5XForwardedFor”，“可执行文件”填写 `F5XForwardedFor.dll` 的完整路径，单击**确定**。
5. 重启 IIS 服务器，等待配置生效。

IIS 7 配置方案

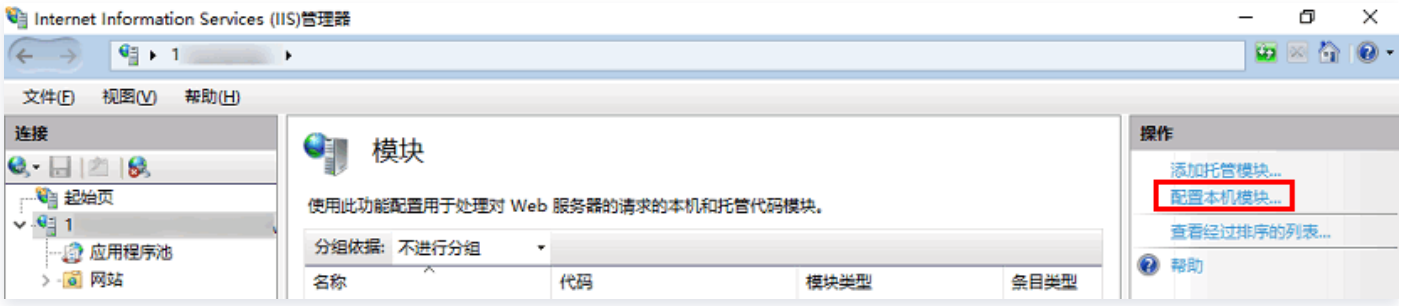
1. 下载与安装插件 **F5XForwardedFor** 模块，根据自己的服务器操作系统版本将 `x86\Release` 或者 `x64\Release` 目录下的 `F5XFFHttpModule.dll` 和 `F5XFFHttpModule.ini` 拷贝到某个目录，这里假设为 `C:\x_forwarded_for`，确保对 IIS 进程对该目录有读取权限。

版权所有：腾讯云计算（北京）有限责任公司

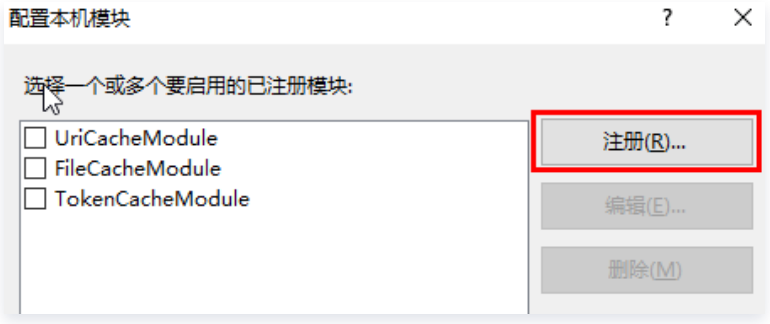
2. 选择 **IIS 服务器**，双击**模块**功能。



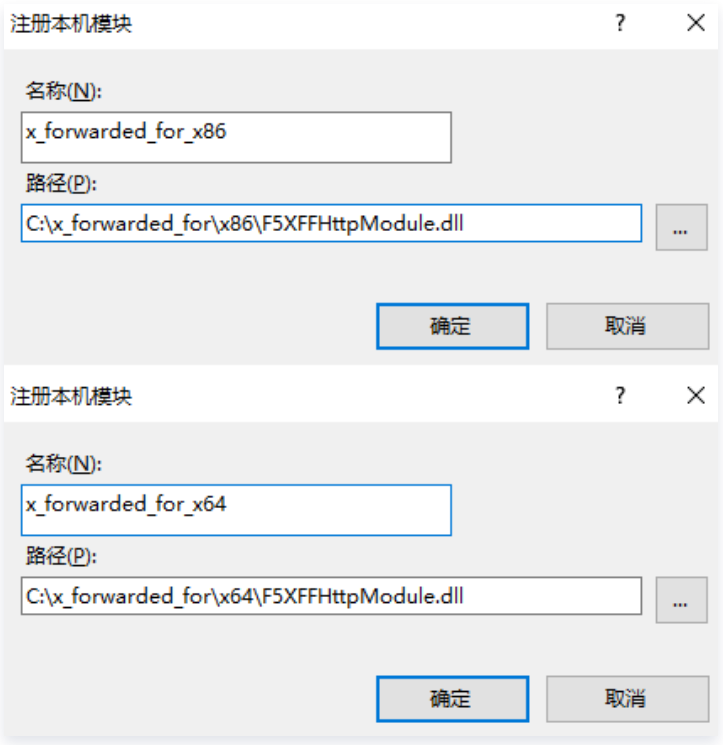
3. 单击**配置本机模块**。



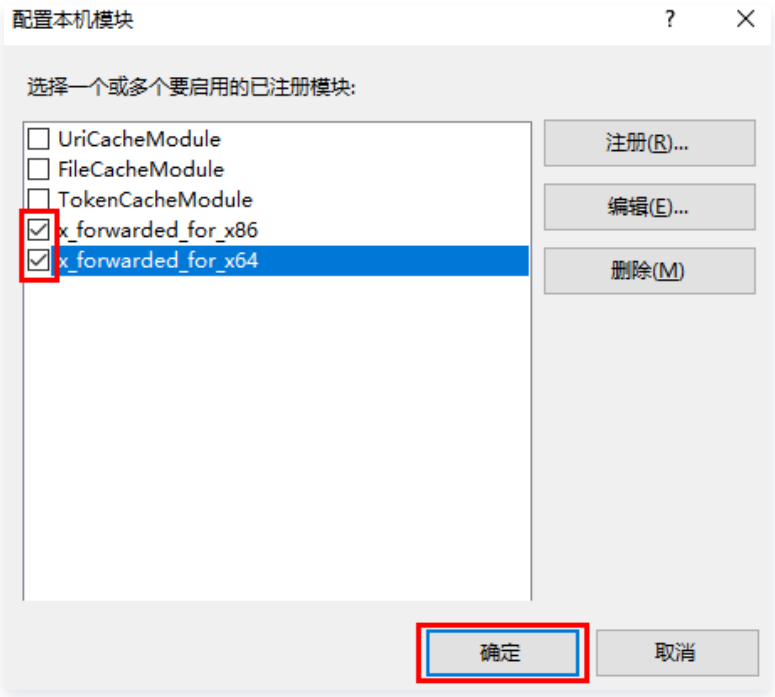
4. 在弹出框中单击**注册**。



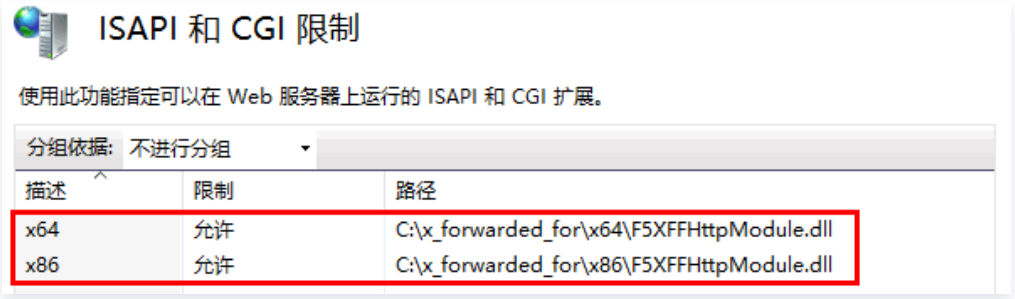
5. 添加下载的 DLL 文件，如下图所示：



6. 添加完成后，勾选并单击**确定**。



7. 在 **ISAPI 和 CGI 限制**添加如下两个 DLL ，并将限制设置为允许。



8. 重启 IIS 服务器，等待配置生效。

Apache 配置方案

1. 检查 Apache 服务器是否安装了 remoteip_module 模块。Apache 使用 remoteip_module 模块解析 X-Forwarded-For 记录。

   如果返回信息中包括 remoteip_module (shared)，表示已安装该模块，可进行下一步。

   > **说明：**
   > - 如果您使用的 Apache 版本过老，建议您备份配置数据并升级 Apache 版本。
   > - 如果您未安装 remoteip_module 模块，请您重新编译安装 Apache 并安装 remoteip_module 模块。

2. 修改 Apache 配置 `/etc/httpd/conf/httpd.conf`，在最末尾添加（具体以实际环境为准）：

```
LoadModule rpaf_module modules/mod_rpaf-2.0.so
RPAFenable On
RPAFsethostname On
RPAFproxy_ips IP地址（这个 IP 地址首先不是负载均衡提供的公网 IP，具体 IP 多少可以查看 Apache 日志，通常会有2个 都要写上）
RPAFheader X-Forwarded-For
```

3. 添加完成后，重启 Apache。

```
/usr/sbin/apachectl restart
```

Nginx 配置方案

1. Nginx 作为服务器时，获取客户端真实 IP 使用 `http_realip_module`，可使用 `nginx -V` 命令查看是否已安装 `http_realip_module` 模块。若 Nginx 未安装此模块，需要重新编译 Nginx 增加 `--with-http_realip_module`。

```
yum -y install gcc pcre pcre-devel zlib zlib-devel openssl openssl-devel
wget http://nginx.org/download/nginx-1.17.0.tar.gz
tar zxvf nginx-1.17.0.tar.gz
cd nginx-1.17.0
./configure --prefix=/path/server/nginx --with-http_stub_status_module --without-http-cache --with-http_ssl_module --with-http_realip_module
make
make install
```

2. 修改 nginx.conf 文件。

```
vi /etc/nginx/nginx.conf
```

修改如下部分的配置字段和信息：

> **说明：**
> 其中 `xx.xx.xx.xx` 需要修改为上一级代理服务器 IP 或者 IP 段。

```
fastcgi connect_timeout 300;
fastcgi send_timeout 300;
fastcgi read_timeout 300;
fastcgi buffer_size 64k;
fastcgi buffers 4 64k;
fastcgi busy_buffers_size 128k;
fastcgi temp_file_write_size 128k;

# 修改如下部分的配置字段和信息
# 若代理的回源IP比较分散，或不清楚代理回源IP时，可以写成0.0.0.0/0，代表所有请求都从XFF中获取源IP
set_real_ip_from xx.xx.xx.xx;
real_ip_header X-Forwarded-For;
real_ip_recursive on;
```

3. 重启 Nginx。

```
service nginx restart
```

4. 查看 Nginx 的访问日志，您可以获取客户端的真实 IP。

```
cat /path/server/nginx/logs/access.log
```

混合云部署场景下通过 TOA 获取客户端真实 IP

最近更新时间：2025-04-30 15:44:12

本文介绍混合云部署场景和 NAT64 CLB 场景下的 CLB 的四层（仅 TCP）服务如何通过 TOA 获取客户端真实源 IP。

- 控制台开启 TOA
- 加载 TOA 模块
- 适配后端服务
- （可选）监控 TOA 模块状态

说明：

- 仅上海、北京、广州地域的 NAT64 CLB 支持通过 TOA 获取客户端真实源 IP。
- 仅四层 TCP 支持通过 TOA 获取客户端真实源 IP，UDP 和七层（HTTP/HTTPS）不支持获取。
- 该功能目前处于内测中，如需使用，请提交 工单申请 。

应用场景

混合云部署场景

在 混合云部署 中，IDC 的 IP 和云上 VPC 的 IP 可能会有地址重叠，因此需要配置 SNAT IP，进行 SNAT 转换源 IP。对于服务端而言，无法获得真实源 IP，因此需要通过 TOA 进行获取。

NAT64 CLB 场景

在 NAT64 CLB 场景中，客户端真实的 IPv6 源 IP 会被转换成 IPv4 的公网 IP，因此对于真实的服务端的服务而言，无法获得真实的客户端 IPv6 IP。
腾讯云 NAT64 CLB 提供获取客户端真实 IP 的功能，即将客户端真实的源 IP 放入 TCP 协议的自定义 option 中，当被嵌入真实源 IP 的 TCP 数据包发往服务端时，服务端插入的 TOA 内核模块可提取 TCP 数据包中的真实客户端源 IP，此时客户端应用只需要调用 TOA 内核模块提供的接口即可获取真实客户端源 IP。

限制说明

资源限制

- 编译 TOA 内核模块环境的内核版本需要与服务所在环境的内核版本一致。
- 容器环境下需要在宿主机中加载 TOA 内核模块。
- 加载 TOA 内核模块的环境需拥有 root 权限。

兼容性限制

- UDP 监听器不支持通过 TOA 获取源 IP。
- 若客户端和真实服务端中间的设备有其他已经进行过 TOA 相关操作的设备，则可能存在冲突，无法保证服务端获取真实 IP 的有效性。
- 插入 TOA 后，只对插入后的新建连接生效，对存量已有连接无效。
- 由于 TOA 模块需要对 TCP option 中的地址进行提取等额外处理，因此 TOA 模块会引起服务端部分的性能下降。
- 腾讯云的 TOA 模块无法保证和其他用户自定义的内核模块兼容，也无法保证与其他厂商或开源的 TOA 模块兼容。
- 腾讯自研的 TencentOS 内嵌的 TOA 模块支持混合云部署场景下获取真实源 IP，因此若服务端的系统为 TencentOS 且为混合云部署时，可尝试直接执行 modprobe toa 命令进行加载使用。需要注意的是，TencentOS 与其他发行版 Linux 系统是两套 TOA，不支持混用。

控制台开启 TOA

1. 已创建 NAT64 版本的 CLB 实例，详情请参见 创建 IPv6 NAT64 负载均衡实例 。
2. 登录 负载均衡控制台 ，创建 TCP 监听器，详情请参见 配置 TCP 监听器 。
3. 在**创建监听器**对话框中，开启 TOA 开关。

加载 TOA 模块

1. 根据腾讯云上 Linux 的版本，下载对应的 TOA 包解压。

centos CentOS 8.0 64 CentOS 7.6 64 CentOS 7.2 64
debian Debian 9.0 64
suse linux SUSE 12 64 SUSE 11 64
ubuntu Ubuntu 18.04.4 LTS 64 Ubuntu 16.04.7 LTS 64

2. 解压完成后，执行 cd 命令进入到刚解压的文件夹里，执行以下命令加载模块：

```
insmod toa.ko
```

3. 执行以下命令确认 TOA 模块是否加载成功。若提示“toa load success”，则说明已加载成功。

```
dmesg -T | grep TOA
```

4. 加载成功以后，在启动脚本中加载 `toa.ko` 文件（重启机器 ko 文件需要重新加载）。
5. （可选）若不再需要使用 TOA 模块，执行以下命令进行卸载。

```
rmmod toa
```

6. （可选）执行以下命令确认 TOA 模块是否卸载成功。若提示“TOA unloaded”，则说明卸载成功。

```
dmesg -T
```

若上述下载文件中没有您的操作系统版本对应的安装包，则可以下载 Linux 通用版的源码包，编译后获取对应的 ko，该版本支持 Centos8、Centos7、Ubuntu18.04、Ubuntu16.04 等绝大多数具有代表性的 Linux 发行版。

> **说明**
> 由于 Linux 内核版本众多，且 Linux 发行版操作系统市场庞大，版本繁多，因此考虑到内核模块的兼容性问题，建议在使用的系统上对 TOA 源码包进行编译后使用。

1. 下载源码包

   Linux：

```
wget "https://clb-toa-1255852779.file.myqcloud.com/tgw_toa_linux.tar.gz"
```

2. 编译 TOA 内核模块的 Linux 环境需先安装 GCC 编译器、Make 工具和内核模块开发包。

CentOS 环境下的安装操作

```
yum install gcc
yum install make
//安装内核模块开发包，开发包头文件与库的版本需要与内核版本一致
yum install kernel-devel-`uname -r`
yum install devtoolset-8
```

Ubuntu、Debian 环境下的安装操作

```
apt-get install gcc
apt-get install make
//安装内核模块开发包，开发包头文件与库的版本需要与内核版本一致
apt-get install linux-headers-`uname -r`
```

```
apt-get install devtoolset-8
```

SUSE 环境下的安装操作

```
zypper install gcc
zypper install make
//安装内核模块开发包，开发包头文件与库的版本需要与内核版本一致
zypper install kernel-default-devel
zypper install devtoolset-8
```

3. 修改 PATH 环境变量为 `PATH=/opt/rh/devtoolset-8/root/bin:$PATH` 。编译前请确认内核 gcc 编译版本，gcc 版本需与编译版本保持一致，可使用 `dmesg | grep 'Linux version'` 命令查看内核 gcc 编译版本信息。
4. 编译源码，生成 toa.ko 文件。编译过程中未提示 `warning` 和 `error` ，则说明编译成功。以 Linux 系统对应的源码包为例：

```
tar zxvf tgw_toa_linux_ver.tar.gz
cd tgw_toa_linux_ver//进入解压后的tgw_toa目录
make
```

5. 编译 toa.ko 成功后，执行上文 步骤2 中的加载 TOA 模块的操作。

适配后端服务

混合云部署场景

在混合云部署场景下适配后端服务时，无需进行代码改造，只需调用 Linux 网络编程中标准的接口即可获取访问用户的真实源 IP。例如以下的 C 代码样例。

```
struct sockaddr v4addr;
len = sizeof(struct sockaddr);
//get_peer_name 为 Linux 网络编程中标准接口。
if (get_peer_name(client_fd, &v4addr, &len) == 0) {
    inet_ntop(AF_INET, &(((struct sockaddr_in *)&v4addr)->sin_addr), from,
sizeof(from));
    printf("real client v4 [%s]:%d\n", from, ntohs(((struct sockaddr_in
*)&v4addr)->sin_port));
}
```

NAT64 CLB 场景

在 NAT64 CLB 场景中，使用 TOA 源地址透传功能，后端服务器在插入 toa.ko 内核模块后，还需对应用程序的源码进行改造以适配获取真实源 IP 的功能。

1. 首先定义一个用来保存地址的数据结构。

```
struct toa_nat64_peer {
    struct in6_addr saddr;
    uint16_t sport;
};
....
```

```
struct toa_nat64_peer client_addr;
....
```

2. 其次定义消息并调用函数获取真实的 IPv6 源地址。

```
enum {
    TOA_BASE_CTL            = 4096,
    TOA_SO_SET_MAX          = TOA_BASE_CTL,
    TOA_SO_GET_LOOKUP       = TOA_BASE_CTL,
    TOA_SO_GET_MAX          = TOA_SO_GET_LOOKUP,
};
getsockopt(client_fd, IPPROTO_IP, TOA_SO_GET_LOOKUP, &client_addr, &len);
```

3. 最后获取地址。

```
real_ipv6_saddr = client_addr.saddr;
real_ipv6_sport = client_addr.sport;
```

完整示例如下所示：

```
//需要定义一个调用获取真实 IP 的函数的消息，值为4096即可。
enum {
    TOA_BASE_CTL            = 4096,
    TOA_SO_SET_MAX          = TOA_BASE_CTL,
    TOA_SO_GET_LOOKUP       = TOA_BASE_CTL,
    TOA_SO_GET_MAX          = TOA_SO_GET_LOOKUP,
};
//需要定义一个用来保存地址的数据结构。
struct toa_nat64_peer {
    struct in6_addr saddr;
    uint16_t sport;
};
//声明用来保存地址的变量，类型为自定义用来保存地址的数据结构。
struct toa_nat64_peer client_addr;
.……
//获取客户端的文件描述符，其中 listenfd 为服务端的监听文件描述符。
client_fd = accept(listenfd, (struct sockaddr*)&caddr, &length);
//调用函数获取对应 NAT64 场景下的用户真实源 IP。
char from[40];
int len = sizeof(struct toa_nat64_peer);
if (getsockopt(client_fd, IPPROTO_IP, TOA_SO_GET_LOOKUP, &client_addr, &len) == 0)
 {
    inet_ntop(AF_INET6, &client_addr.saddr, from, sizeof(from));
    //获取源IP和源port的信息
    printf("real client [%s]:%d\n", from, ntohs(client_addr.sport));
}
```

混合云部署与 NAT64 CLB 混合场景

在混合云部署与 NAT64 CLB 混用场景中，使用 TOA 源地址透传功能，后端服务器在插入 toa.ko 内核模块后，还需对应用程序的源码进行改造以适配获取真实源 IP 的功能。
完整示例如下所示：

```
//需要定义一个调用获取真实 IP 的函数的消息，值为4096即可。
enum {
    TOA_BASE_CTL = 4096,
    TOA_SO_SET_MAX = TOA_BASE_CTL,
    TOA_SO_GET_LOOKUP = TOA_BASE_CTL,
    TOA_SO_GET_MAX = TOA_SO_GET_LOOKUP,
};

//需要定义一个用来保存地址的数据结构。
struct toa_nat64_peer {
    struct in6_addr saddr;
    uint16_t sport;
};

//声明用来保存地址的变量，类型为自定义用来保存地址的数据结构。
struct toa_nat64_peer client_addr_nat64;
.......
//获取客户端的文件描述符，其中 listenfd 为服务端的监听文件描述符。
//调用函数获取对应 NAT64 场景下真实的用户源 IP。
char from[40];
int len = sizeof(struct toa_nat64_peer);
int ret;
ret = getsockopt(client_fd, IPPROTO_IP, TOA_SO_GET_LOOKUP, &client_addr_nat64,
&len);
if (ret == 0) {
    inet_ntop(AF_INET6, &(client_addr_nat64.saddr), from, sizeof(from));
    //获取源 IP 和源 Port 的信息。
    printf("real client v6 [%s]:%d\n", from, ntohs(client_addr_nat64.sport));
} else if (ret != 0) {
    struct sockaddr v4addr;
    len = sizeof(struct sockaddr);
    //获取源 IP 和源 Port 的信息，注意此函数获取的源地址对于：
    //经过混合云部署场景的 SNAT IP 的链接而言为真正的源地址；
    //不经过混合云部署场景的 SNAT IP 也不经过 NAT64 的链接而言是客户端地址，同样是真正的源地
址。
    //因此此函数的语义便为获取真正的客户端地址、端口。
    if (get_peer_name(client_fd, &v4addr, &len) == 0) {
        inet_ntop(AF_INET, &(((struct sockaddr_in *)&v4addr)->sin_addr), from,
sizeof(from));
        printf("real client v4 [%s]:%d\n", from, ntohs(((struct sockaddr_in
*)&v4addr)->sin_port));
    }
}
```

（可选）监控 TOA 模块状态

为保障 TOA 内核模块运行的稳定性，TOA 内核模块还提供了监控功能。在插入 toa.ko 内核模块后，可以在容器所在的宿主机通过以下两种方式监控 TOA 模块的工作状态。

方式一：查看 TOA 保存的连接的 IPv6 地址

执行以下命令查看 TOA 保存的连接的 IPv6 地址。

> ⚠ **注意：**
> 此命令有可能会引起性能下降，请勿频繁调用此命令查看。

```
cat /proc/net/toa_table
```

方式二：查看 TOA 相关的计数状态

执行以下命令查看 TOA 相关的计数状态。

```
cat /proc/net/toa_stats
```

其中主要的监控指标对应的含义如下所示：

指标名称	说明
syn_recv_sock_toa	接收带有 TOA 信息的连接个数。
syn_recv_sock_no_toa	接收并不带有 TOA 信息的连接个数。
getname_toa_ok	调用 getsockopt 获取源 IP 成功即会增加此计数，另外调用 accept 函数接收客户端请求时也会增加此计数。
getname_toa_mismatch	调用 getsockopt 获取源 IP 时，当类型不匹配时，此计数增加。例如某条客户端连接内存放的是 IPv4 源 IP，并非为 IPv6 地址时，此计数便会增加。
getname_toa_empty	对某一个不含有 TOA 的客户端文件描述符调用 getsockopt 函数时，此计数便会增加。
ip6_address_alloc	当 TOA 内核模块获取 TCP 数据包中保存的源 IP、源 Port 时，会申请空间保存信息。
ip6_address_free	当连接释放时，toa 内核模块会释放先前用于保存源 IP、源 port 的内存，在所有连接都关闭的情况下，所有 CPU 的此计数相加应等于 ip6_address_alloc 的计数。

FAQ

为什么在 NAT64 CLB 场景下插入了 TOA 模块后仍需要改造服务端程序?

这是由于 IP 类型发生了变化导致的。在混合云部署场景下，做了 IPv4 的 Fullnat 转换，在此场景下，客户端的真实源 IP 仍然是从 IPv4 的 IP 转换成另外一个 IPv4 的 IP，因此 IP 的类型没有发生变化。但是在 NAT64 CLB 场景下，客户端真实源 IP

是从 IPv6 转换成了 IPv4，IP 类型发生了变化，因此服务端为了理解此 IPv6 的 IP，必须要对服务端程序进行改造才可以理解此 IPv6 地址的含义。

如何确定所用的系统是基于 Linux 的发行版还是腾讯 TLinux 的内核?

- 执行以下命令查看内核版本。若执行结果的版本中包含 `tlinux`，则为 TLinux 系统。反之则为 Linux 发行版。

```
uname -a
```

- 还可以执行以下命令，若执行结果中包含 `tlinux` 或者是 `tl2`，则为 TLinux 系统。

```
rpm -qa | grep kernel
```

无法获取源地址，如何进行初步的排查?

1. 执行以下命令确认 TOA 模块是否已经加载。

```
lsmod | grep toa
```

2. 确认服务端程序是否已经正确调用接口获取源地址，请参见以上 适配后端服务 内容。
3. 在服务端抓包排查，确认是否已经有携带真实源地址的 TCP 包抵达。
   - 若 tcp option 中存在 `unknown-200` 的提示，则说明经过 SNAT 后，真实的源 IP 已经插入到 TCP option 中。
   - 若存在 `unknown-253`，则说明在 NAT64 场景下的真实 IPv6 的源 IP 已经插入。

```
[root@VM-0-133-centos ]# tcpdump -i any "ip[40:1]==200" -c 100
dropped privs to tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked v1), capture size 262144 bytes
18:04:24.864649 IP 192.168.0.177.23638 > VM-0-133-centos.webcache: Flags [.], ack 3309146461, win 229, options [unknown-200 0xa
2662ac13bca,nop,nop,TS val 2243901958 ecr 3395797654], length 0
18:04:24.864679 IP 192.168.0.177.23638 > VM-0-133-centos.webcache: Flags [P.], seq 0:154, ack 1, win 229, options [unknown-200
0xa2662ac13bca,nop,nop,TS val 2243901958 ecr 3395797654], length 154: HTTP: GET /data/1K HTTP/1.1
```

4. 在上一步的操作中，若确定携带 TOA 地址的包进入了服务端，则将 toa.ko 编译出 DEBUG 版本，通过内核日志便可进一步定位。在下载出的 TOA 源码目录中，将 Makefile 中添加 DEBUG 编译选项。

```
endif
PWD       := $(shell pwd)

ccflags-y += -DTOA_NAT64_ENABLE -DTOA_DEBUG_ENABLE

ifeq ($(DEBUG), 1)
ccflags-y += -g -O0
endif
```

5. 执行以下命令重新编译。

```
make clean
make
```

6. 执行以下命令卸载原有 ko，并重新插入编译出的最新 ko。

```
rmmod toa
insmod ./toa.ko
```

7. 执行以下命令观察内核日志。

```
dmesg -Tw
```

若提示以下内容，则说明 TOA 模块正常工作，请进一步排查服务端程序是否有调用接口获取真实源 IP，或是是否接口使用错误。

```
[Wed Dec 29 18:07:11 2021] [DEBUG] TOA: inet_getname_toa called, sk->sk_user_data is 000000003088927f
[Wed Dec 29 18:07:11 2021] [DEBUG] TOA: inet_getname_toa: set new sockaddr, ip 192.168.0.177 -> 42.193.59.202, port 49682 -> 41
618
```

8. 若以上步骤皆没有排查出具体原因，请 联系我们 。

负载均衡配置监控告警最佳实践

最近更新时间：2025-05-23 19:48:02

为完善负载均衡 CLB 业务监控体系，结合腾讯云可观测平台的数据收集与告警能力，打造一体化预警机制。您可以通过使用腾讯云可观测平台全面了解负载均衡 CLB 的资源使用、性能和运行状况，您可以为您关注的实例配置监控告警，设置监控指标和事件的告警触发规则。当该实例的监控指标异常时，您可以第一时间接收到异常告警通知，及时响应处理故障。更多内容请参见 告警管理简介 。

使用场景

您可以为您关注的实例指标创建告警，使负载均衡 CLB 实例在运行状态达到某一条件时，及时发送告警信息至关心的用户群体。更方便、快捷的掌控可能出现的突发情况，提升运维效率，减少运维成本。
本文将介绍如何为已升级为性能容量型的公网负载均衡 CLB 实例配置告警，以标准型为例。更多性能容量型规格介绍，请参见 性能容量型规格介绍 。

前提条件

- 您已创建负载均衡实例并配置监听器，详情请参见 负载均衡快速入门 。
- 您已成功绑定后端服务器，详情请参见 绑定后端服务器 。
- 根据本例，目标实例需已升级为性能容量型，详情请参见 升级为性能容量型实例 。

基本概念

术语	定义
告警策略	由策略名称、策略类型、告警对象、触发条件和通知模板组成。
策略类型	告警策略类型用于标识策略分类，类型与云产品对应。例如：当您选择云服务器策略，即可自定义 CPU 使用率、磁盘使用率等指标告警。
触发条件	触发条件是指标、比较关系、阈值、统计粒度和持续 N 个监控数据点组成的一个有语义的条件。
监控类型	支持云产品监控、应用性能监控、前端性能监控、云拨测和终端性能监控。
通知模板	多个策略一键复用模板，适用于多种场景接收告警通知，详情请参考 新建通知模板 。

指标介绍

判断性能容量型实例是否超限的核心指标有：客户端到 LB 的并发连接数、客户端到 LB 的新建连接数、每秒请求数、客户端到 LB 的出带宽、客户端到 LB 的入带宽，故需要关注上述核心指标的利用率告警指标，如下表所示。其中丢弃/利用率监控指标处于内测阶段，如需使用，请提交 工单申请 。更多告警指标的说明请参见 告警指标说明 。

维度	告警策略类型	告警策略	告警指标	指标说明
实例	公网负载均衡实例	丢弃/利用率监控	入带宽利用率	在统计粒度内，客户端通过外网访问负载均衡所用的带宽利用率。
			出带宽利用率	在统计粒度内，负载均衡访问外网所用的带宽使用率。
			并发连接数利用率	在统计粒度内的某一时刻，从客户端到负载均衡的并发连接数相比性能容量型规格的并发连接数性能上限的利用率。

			新建连接数利用率	在统计粒度内的某一时刻，从客户端到负载均衡的新建连接数相比性能容量型规格的新建连接数性能上限的利用率。
		QPS 相关监控	QPS 利用率	在统计粒度内的某一时刻，负载均衡的 QPS 相比性能容量型规格的 QPS 性能上限的利用率。

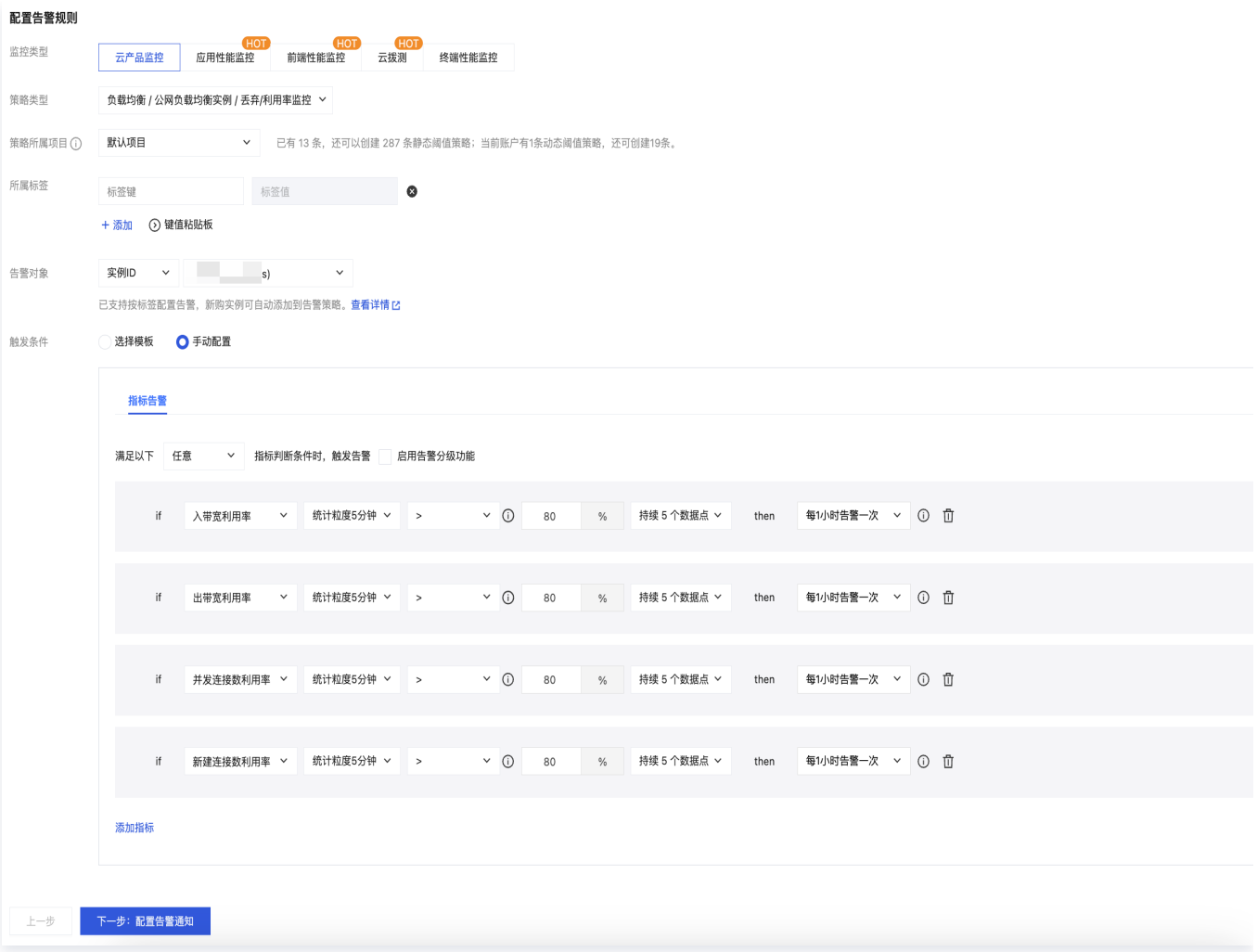
操作步骤

1. 登录 腾讯云可观测平台 。
2. 在左侧导航栏中，单击**告警管理 > 告警配置**，进入告警策略页面。
3. 单击**新建策略**，配置以下选项。
   3.1 基本信息
   - 策略名称：输入策略名称，最多60个字符。
   - 备注：输入备注，最多100个字符。

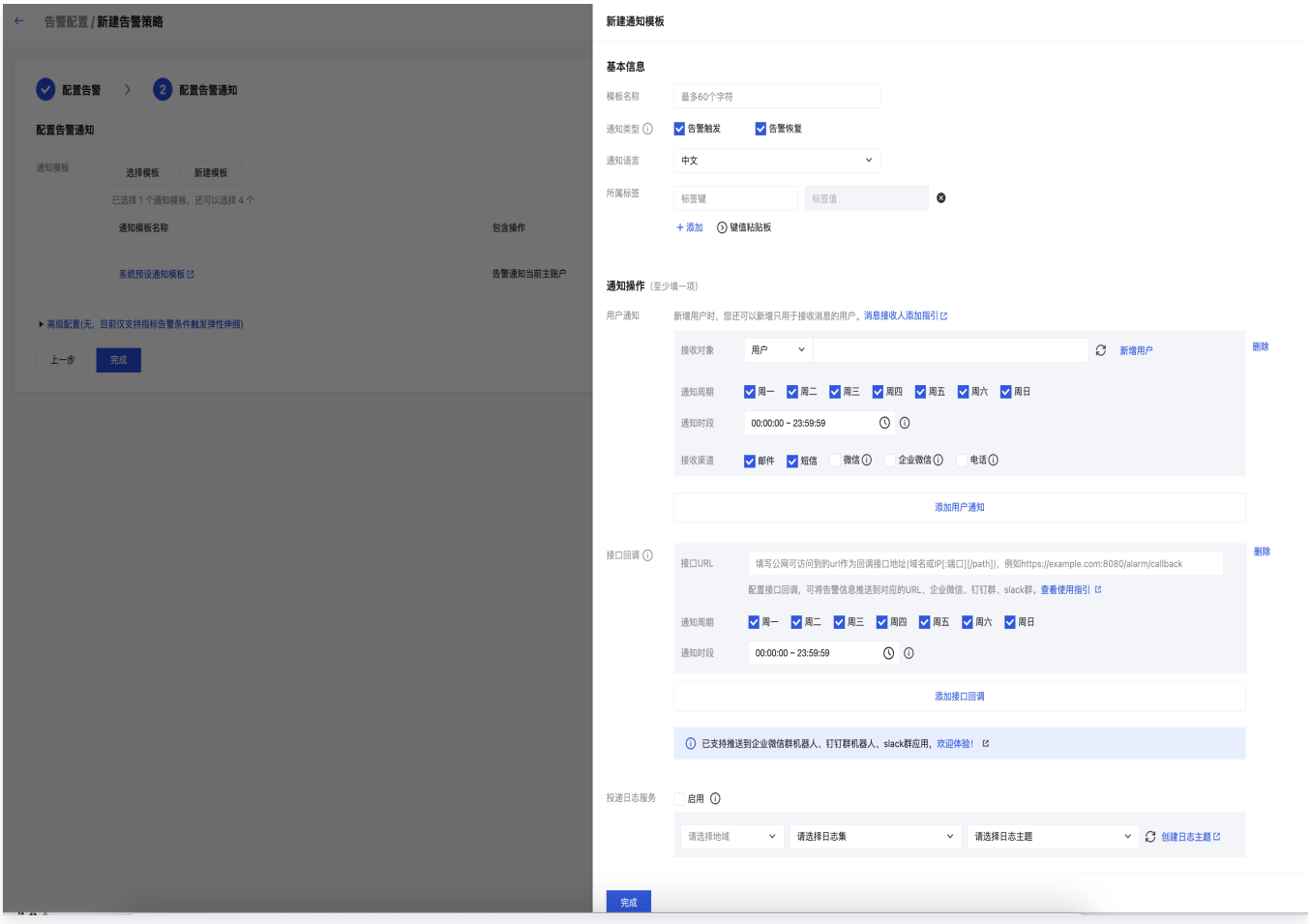
   3.2 配置告警规则
   - 监控类型：选择**云产品监控**。
   - 策略类型：选择**负载均衡 > 公网负载均衡实例 > 丢弃/利用率监控**。
   - 策略所属项目：选择策略所属项目。所属项目用于告警策略的分类和权限管理，与云产品实例的项目没有强绑定关系。
   - 所属标签：选择策略所属标签。
   - 告警对象：选择目标实例作为告警对象。
   - 触发条件：告警指标、统计粒度、比较关系、阈值、持续 N 个监控数据点和告警频率组成的一个有语义的条件。

     例如，告警指标为**入带宽利用率**、**统计粒度为5分钟** 、比较关系为 **>** 、阈值为 **80%** 、**持续监控数据点为 5 个数据点**、告警频率为**每 1 个小时告警一次**。表示：每 5 分钟收集一次入带宽利用率数据，若某负载均衡实例的入带宽利用率连续 5 次大于 80% 则触发告警，告警频率为每 1 小时告警一次。

     选择配置入带宽利用率、出带宽利用率、并发连接数利用率、新建连接数利用率，示例如下图所示。



3.3 配置告警通知：选择通知模板，选择告警接受对象、通知周期与接受渠道。若未创建通知模板，请单击**新建模板**进行创建，详情请参见 新建通知模板 。



4. 单击**完成**，即可完成配置入带宽利用率、出带宽利用率、并发连接数利用率、新建连接数利用率的监控告警。QPS 利用率监控告警请参考上一步骤新建告警策略，修改策略类型为**负载均衡** > **公网负载均衡实例** > **QPS 相关监控**，触发条件配置以下内容即可。

解决方案

当接收到上述告警后，表明您业务量上涨，当前标准型的性能容量型实例规格即将达到性能上限，无法满足业务需求。请前往 调整性能容量型实例规格，以确保业务不受影响。

产品高可用说明

最近更新时间：2024-12-02 16:45:23

负载均衡 CLB 的高可用是从系统架构、产品配置等多维度来保障的。您可以根据业务场景和需求，选择跨地域容灾、同地域跨可用区容灾等多种功能方案。

CLB 集群高可用

负载均衡CLB实例采用集群部署，支持会话同步，消除服务器单点，提升系统冗余，保证服务稳定。所有CLB实例均具备集群高可用。

- 四层主要基于腾讯自研的统一接入网关（Tencent Gateway，TGW）来实现负载均衡，TGW 具有可靠性高、扩展性强、性能高、抗攻击能力强等特点，支持 Data Plane Development Kit（DPDK）高性能转发，单集群可支持亿级并发、千万级 PPS。腾讯内部诸多业务均通过 TGW 接入服务，包括腾讯游戏、腾讯视频、微信、QQ 等。
- 七层主要基于 Secure Tencent Gateway（STGW）实现负载均衡，STGW 是腾讯基于 Nginx 自研的支持大规模并发的七层负载均衡服务，承载了腾讯内大量的七层业务流量。

单 CLB 实例高可用

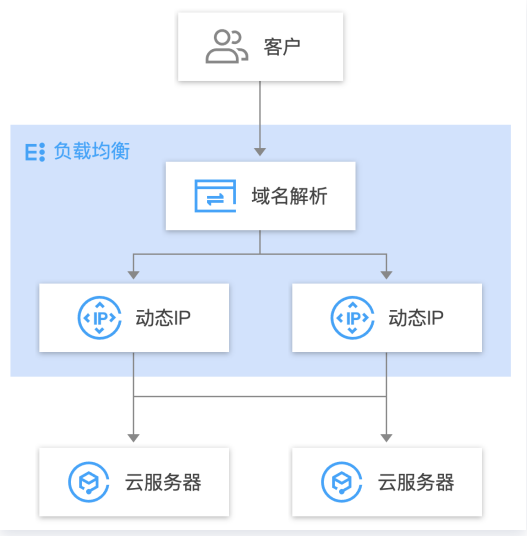
非域名化公网 CLB

非域名化的公网 CLB 以 VIP 形式提供服务，SLA 为99.95%，VIP 所属集群有2种部署方案：

部署模式	集群容灾	跨可用区容灾
单可用区	支持	不支持
多可用区	支持	支持，主备可用区模式，当主可用区故障时，负载均衡可在非常短的时间内（约30s）自动切换到备可用区并恢复服务

域名化公网 CLB

“域名化公网 CLB”在上述“非域名化公网 CLB”的基础上，增加一层 DNS 服务，SLA 从99.95%提升至99.99%，可自动替换故障 VIP，提高可用性。详情请参见 域名化公网负载均衡上线公告 。

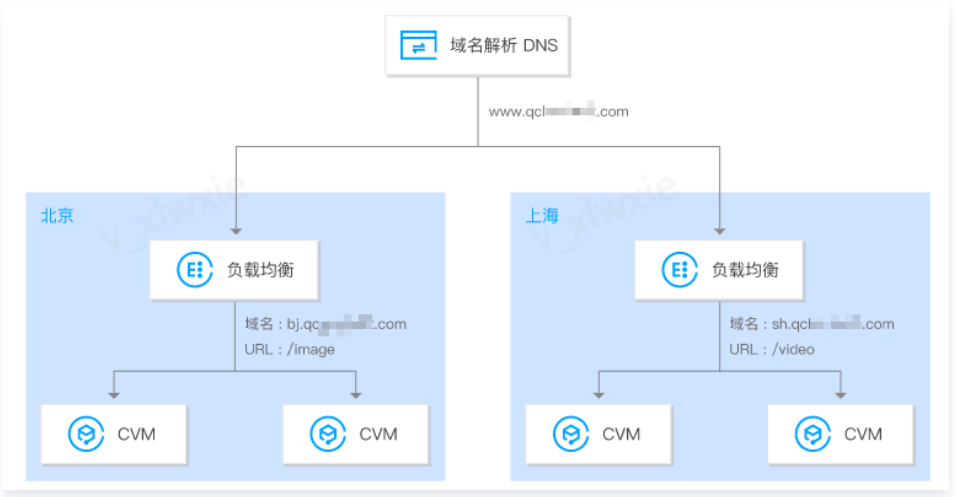


内网 CLB

内网 CLB 采用就近接入架构部署，同一个 CLB 实例会下发到一个或多个可用区，客户端访问该 CLB 时，访问流量会自动选择延时最低的可用区集群，然后转发到后端服务器。
内网 CLB 暂不具备跨可用区容灾切换的能力，如果某个可用区的 CLB 集群不可用，会影响来自该可用区访问源的访问，若访问源为其他可用区且对应可用区有就近接入，则流量不受影响；若访问源为其他可用区，对应可用区无就近接入，且默认指向故障 CLB，则流量受影响。

多 CLB 实例高可用

如果您对可用性要求非常高，CLB 实例自身的可用性保障机制可能无法满足您的需求，如网络攻击、跨地域切换、配置有误等场景。您可以创建多个 CLB 实例，通过云解析 DNS 对访问流量进行调度。



多 CLB 实例高可用与域名化公网 CLB 对比：

对比项	域名化公网 CLB	多 CLB 实例高可用
SLA	99.99%	99.95%
容灾切换	提供链路检测和容灾切换的能力，无需担心单 IP 入口中断问题。当单 IP 发生故障时，可自动切换故障 IP，降低业务影响。	依赖您配置的 DNS 解析及切换策略，需业务及时发现及切换。
运维管理	仅需配置单实例。	需配置多个 CLB 实例和对应的 DNS 解析策略。
成本	成本较低，仅收取CLB相关费用。	需部署多个 CLB 实例和 DNS 解析等组件，成本更高。
地域	CLB 实例所属集群部署在单地域。	可选择多地域的 CLB 实例。

最佳实践：

- 若您的业务是单地域部署，建议优先选择域名化公网 CLB 方案，自动切换故障 IP；
- 若您的业务是多地域部署，且对容灾需求非常高，建议选择多 CLB 实例高可用方案。
- 同一个客户端在同一时刻，通过不同的中间节点访问同一个后端服务器的同一个端口可能会出现串流现象，详情请参见 串流问题说明 。

后端服务高可用

负载均衡 CLB 通过健康检查来判断后端服务的可用性，避免后端服务异常影响前端业务，从而提高业务整体可用性。
开启健康检查后，无论后端服务器权重是多少（包括权重为0），负载均衡实例都会进行健康检查。您可在实例列表页面的**健康状态**列查看健康检查状态，或者在监听器的绑定后端服务详情页面查看健康检查状态。关于健康检查的详细机制，请参见 健康检查概述 。

均衡算法选择与权重配置示例

最近更新时间：2023-11-24 17:53:21

负载均衡算法比较分析

加权轮询算法 Weighted Round-Robin Scheduling

加权轮询算法是以轮叫的方式、依次请求调度不同的服务器。加权轮询调度算法可以解决服务器间性能不一的情况，它用相应的权值表示服务器的处理性能，按权值的高低和轮询方式分配请求到各服务器。加权轮询算法根据新建连接数来调度，权值高的服务器先收到连接，权重值越高被轮询到的次数（概率）也越高，相同权值的服务器处理相同数目的连接数。

- **优势**：简洁实用，无需记录当前所有连接的状态，是一种无状态调度。
- **劣势**：相对简单，在请求服务时间变化较大或每个请求消耗时间不一致的情况下，容易导致服务器间的负载不平衡。
- **适用场景**：当每个请求所占用的后端时间基本相同时，负载情况最好。常用于短连接服务，例如 HTTP 等。
- **用户推荐**：已知每个请求所占用后端时间基本相同、后端服务器处理的请求类型相同或者相似时，推荐您选择加权轮询的方式。请求时间相差较小时，也推荐您使用加权轮询的方式，因为该实现方式消耗小，无需遍历，效率较高。

加权最小连接数 Weighted Least-Connection Scheduling

- **原理**
  在实际情况中，客户端的每一次请求服务在服务器停留的时间可能会有较大的差异，随着工作时间的延伸，如果采用简单的轮询或随机均衡算法，每一台服务器上的连接进程数目可能会产生极大的不同，这样实际上并没有达到真正的负载均衡。最小连接调度是一种动态调度算法，它通过服务器当前所活跃的连接数来评估服务器的负载情况。与轮询调度算法相反。调度器需要记录各个服务器已建立连接的数目，当一个请求被调度到某台服务器，其连接数加1；当连接中止或超时，其连接数减一。权重最少连接数调度算法是在最少连接数调度算法的基础上，根据服务器的不同处理能力，给每个服务器分配不同的权值，使其能够接受相应权值数的服务请求，是在最少连接数调度算法的基础上的改进。
  1.1 假设各台 RS 的权值依次为 Wi（i=1…n），当前连接数依次为 Ci（i=1…n），依次选取 Ci/Wi 值为最小的 RS 作为下一个分配的 RS。
  1.2 若存在 Ci/Wi 相同的 RS，则这些 RS 再使用加权轮询的方式调度。
- **优势**
  此种均衡算法适合长时处理的请求服务，如 FTP 等应用。
- **劣势**
  由于接口限制，目前最小连接数和会话保持功能不能同时开启。
- **适用场景**
  每个请求所占用的后端时间相差较大的场景。常用于长连接服务。
- **用户推荐**
  如果用户需要处理不同的请求，且请求所占用后端时间相差较大，如3ms和3s这种数量级的差距时，推荐使用加权最小连接数算法实现负载均衡。

源地址散列调度算法 ip_hash

- **原理**
  根据请求的源IP地址，作为散列键（Hash Key）从静态分配的散列表找出对应的服务器，若该服务器是可用的且未超载，将请求发送到该服务器，否则返回空。
- **优势**
  ip_hash 可以实现部分会话保持的效果，能够记住源 IP，使某一 client 请求通过 hash 表一直映射在同一台 rs 上。因此在不支持会话保持的场景可以使用 ip_hash 进行调度。

- **用户推荐**
  将请求的源地址进行hash运算，并结合后端的服务器的权重派发请求至某匹配的服务器，这可以使得同一个客户端 IP的请求始终被派发至某特定的服务器。该方式适合负载均衡无 cookie 功能的 TCP 协议。

均衡算法选取及权重配置示例

在负载均衡即将发布的新功能中，**七层转发将支持最小连接数的均衡方式**，为了让用户在不同场景下，能够让 RS 集群稳定的承接业务，因此我们给出几个负载均衡选择与权重配置的实例供用户进行参考。

- 场景1
  设有3台配置相同（CPU / 内存）的 RS，由于性能一致，用户可以将 RS 权重都设置为10。设现在每台 RS 与 client 端建立了100个 TCP 连接，此时新增1台 RS。在此场景下，推荐用户使用最小连接数的均衡方式，这种配置能快速的让第四台 RS 的负载提升，降低另外3台 RS 的压力。
- 场景2
  设用户首次接触云服务，且建站时间不长，网站负载较低，则建议购买相同配置的 RS，因此 RS 都是无差别的接入层服务器。在此场景下，用户可以将 RS 权重都设为10，采用加权轮询的均衡方式进行流量分发。
- 场景3
  用户有5台服务器，用于承载简单的静态网站访问，且5台服务器的计算能力的比例为 9：3：3：3：1（按 CPU、内存换算）。在此场景下，用户可以依次将 RS 权重比例设置为90，30，30，30，10，由于静态网站访问大多数是短连接请求，因此可以采用加权轮询的均衡方式，让 CLB 按 RS 的性能比例分配请求。
- 场景4
  某用户有10台 RS 用于承担海量的 Web 访问请求，且不希望多购置 RS 增加支出。某台 RS 经常会因为负载过高，导致服务器重启。在此场景下，建议用户根据 RS 的性能进行相应的权重设置，给负载过高的 RS 设置较小的权值。除此之外，可以采用最小连接数的负载均衡方式，将请求分配到活跃连接数较少的 RS 上，从而解决某台 RS 负载过高的问题。
- 场景5
  某用户有3台 RS 用于处理若干长连接请求，且这3台服务器的计算能力比例为3：1：1（按 CPU、内存换算）。此时性能最好的服务器处理请求较多，用户不希望过载此服务器，希望能够将新的请求分配到空闲服务器上。在此场景下，可以采用最小连接数的均衡方式，并适当降低繁忙服务器的权重，便于 CLB 将请求分配到活跃数较少的 RS 上，实现负载均衡。
- 场景6
  某用户希望后续客户端的请求可以分配到同一服务器上。而采用加权轮询或加权最小连接数的方式，不能保证相同客户端的请求被分到固定某台服务器上去。为了配合客户特定应用程序服务器的需求，保证客户端的会话具有“粘性”或是“持续性”，在此场景下，我们可以采用 ip_hash 的均衡方式进行流量分发。此方法可以确保来自同一客户端的请求总被定向分发到同一 RS 上去。（服务器数量变化或是该服务器不可用时除外）

权重置为0与解绑 RS 的区别

- 权重置为0：TCP 监听器存量连接继续转发，UDP 监听器相同五元组的继续转发，HTTP/HTTPS 监听器存量连接继续转发。TCP、UDP、HTTP/HTTPS 监听器新增连接不会再转发到权重为 0 的 RS 上。
- 解绑 RS：TCP/UDP 监听器存量连接立即停止转发，HTTP/HTTPS 监听器存量连接继续转发，存量连接转发完毕之后断开与 RS 的连接。

相关文档

修改后端服务权重

配置 WAF 对负载均衡的监听域名进行 Web 安全防护

最近更新时间：2025-05-23 17:06:12

负载均衡型 Web 应用防火墙（WAF）通过域名和负载均衡监听器进行绑定，实现对经过负载均衡监听器的 HTTP 或 HTTPS 流量进行检测和拦截。本文档将介绍如何通过负载均衡型 WAF 为已经添加到负载均衡的域名进行 Web 安全防护。

前提条件

- 您已成功创建 HTTP 监听器或 HTTPS 监听器，并且域名可以正常访问。操作详情请参考 负载均衡快速入门。
- 您已成功购买负载均衡型 WAF。购买方式请参考 购买方式。若未购买，可选择 7天免费试用 负载均衡型 WAF。

操作步骤

步骤1：确认负载均衡域名配置

本文以防护 `www.example.com` 域名为例。

1. 登录 负载均衡控制台，在左侧导航栏中，单击**实例管理**。
2. 在**实例管理**页面，选择所在地域，在实例列表中单击目标实例右侧“操作”列的**配置监听器**。
3. 在**监听器管理**页签的 **HTTP/HTTPS 监听器**区域，单击目标监听器左侧的 + 查看域名详情。

4. 确认负载均衡域名配置信息：负载均衡实例的 ID 为“lb-f8lm****”，监听器的名称为“http-test”，监听器转发规则所监听的域名为 `www.example.com`，域名防护状态为“未启用”（所有 ID、名称和域名以实际为准）。

步骤2：在 WAF 中添加域名绑定负载均衡

为了使负载均衡型 WAF 能够识别出需要防护的域名，需要在 WAF 中添加负载均衡监听的域名并绑定负载均衡监听器。

1. 登录 Web 应用防火墙控制台，在左侧导航栏中，选择**资产中心** > **接入管理** > **域名接入**。
2. 在域名接入页面，单击**添加域名**，配置相关参数，单击**确定**即可。

字段说明：

- **所属实例：**选择负载均衡型和实例名称。
- **域名：**在域名输入框中添加需要防护的域名 `www.example.com` 。
- **流量来源：**选择 CLB 即可。
- **代理情况：**根据实际情况选择是否已使用了高防、CDN、云加速等代理。

> **说明：**
> 选择“是”，WAF 将通过 XFF 字段获取客户真实 IP 地址作为源地址，可能存在源 IP 被伪造的风险。

- **国内地域：**根据实际需求选择。
- **选择域名对应的负载均衡监听器：**根据实际需求选择和配置接入域名的监听器信息。

> **说明：**
> 当前 WAF 已经支持公网型和内网型的负载均衡型实例的监听器流量接入防护，可以通过网络类型字段进行查看和筛选。

3. 单击**确定**，即可返回域名接入。在域名接入可以查看到防护域名 `www.example.com` 和负载均衡的负载均衡 ID、名称、VIP 和监听器信息等。

步骤3：结果验证

1. WAF 通过域名和 CLB 对应监听器进行绑定，对经过 CLB 监听器的域名流量进行防护。验证负载均衡型 WAF 是否生效，请先确保本地电脑可以正常访问在负载均衡不同实例下添加的域名。

> **说明：**
> 验证添加在负载均衡中域名型访问是否正常，IPv4 域名请求，请参见负载均衡快速入门的 验证负载均衡服务，IPv6 域名请求，请参见 IPv6 负载均衡快速入门的 步骤4：测试 IPv6 负载均衡 。

2. 在浏览器中输入网址 `http://www.example.com/?test=alert(123)` 并访问，浏览器返回阻断页面，说明 Web 应用防火墙防护功能正常。

> **注意：**
> `www.example.com` 为本案例中域名，此处需要将域名替换为实际添加的域名。