Bash Shell脚本得到宽带公网动态IP地址

Published on Feb 20, 2022 UTC by Wang Ye <https://wangye.org>

如何确定公网IP地址是一个让很多家用宽带朋友们经常遇到的问题，毕竟固定IP地址总是价格不菲因而不适用于家庭宽带，而我们的电信宽带运营商（ISP）往往给我们分配的是动态IP地址，之所以称作为动态IP地址，是因为每次拨号得到的IP地址可能会不一样，还有更骚的操作就是有的运营商每天或不定期在某个时间刷新IP地址池，这样已经获得的IP地址将会被强制释放并分配新的IP地址，这对于我们一些运行在家用带宽下的服务带来了影响——需要重新设置服务端IP地址，比如监控、网络存储等等，这时候DDNS就发挥作用的，现在大部分路由器内置了DDNS客户端，比如花生壳等服务客户端，但是这些客户端存在收费、不稳定以及刷新间隔小等问题，对于自己有域名的朋友可以使用DNS服务商的API自己实现IP更新操作，比如阿里云（Aliyun）、Cloudflare均可以实现。

对于DDNS如何使用API更新不在本文叙述范围内，但调用API有个关键参数那就是公网IP地址，如何获取当前运营商分配的公网IP地址呢？除了问路由器外（打开路由器管理界面找到WAN口信息）我们还可以通过一些网络服务进行检测。

1 使用Shell命令获取公网IP地址

1.1 浏览器方式

如果我们使用浏览器打开这个地址checkip.amazonaws.com你就会发现你的公网IP地址赫然在目，互联网上的服务器总是能够知道是什么IP地址的客户端发起了连接，当然就可以通过这种方式获取公网IP，同样功能的网站有很多，这里列举部分我收集到的：

```
https://checkip.amazonaws.com
https://api.ipify.org
https://ifconfig.me/ip
https://icanhazip.com
```

```
https://ipinfo.io/ip
https://ipecho.net/plain
https://checkipv4.dedyn.io
```

1.2 cURL方式

在命令行下我们当然不能为这点小事随便启动浏览器，我们可以请出另外一个功能强大的工具，那就是curl，cURL是一个利用URL语法在命令行下工作的文件传输工具，关于cURL的快速用法可以参考阮一峰的网络日志《curl 的用法指南》，这里不再详述，最简单的用法就是`curl` 你要请求的网址，比如`curl checkip.amazonaws.com`。

1.3 使用dig命令

首次在别人脚本中看到这个方式获取公网IP觉得比较新奇，命令如下：

```
dig +short myip.opendns.com @resolver1.opendns.com
```

这个是由OpenDNS提供的服务，如果你的主机上没有安装dig命令，对于Debian系列系统可以通过`apt-get install dnsutils`安装，这个命令的原理是指定`resolver1.opendns.com`为域名`myip.opendns.com`的解析服务器，为什么要指定，主要是避免DNS下游服务器缓存，另外这个指定的解析服务器被OpenDNS进行了特殊配置，其始终将域名`myip.opendns.com`解析为发起DNS请求的客户端IP地址，这样也就实现了查找公网IP的功能。

这个和cURL方式比有什么优势呢？当然是数据量更小传输更快，毕竟使用cURL发起HTTP请求必然会导致协议头等无关信息的交换，如果启用了HTTPS/TLS那么还要进行加密协商，效率会比较低，当然这种方式的缺点就是DNS查询容易被运营商审计和过滤，稳定性略差，不过我测试下来除了偶尔查询失败外其余情况是完全OK的。

2 使用Shell脚本获取公网IP地址

至此我有个新的主意，那就是首先采用dig命令快速检索公网IP地址，如果失败则切换到cURL的方式，为了避免cURL单一服务器失败，采用轮询或者随机的方式，尽可能提高成功率。

使用Bash Shell脚本do it，比如获取公网IPv4的脚本如下：

```
#!/bin/bash

# This script try to ensure gets the current IP address (as assigned
by the ISP) from
# OpenDNS and other online services as fallbacks

hosts=("checkip.amazonaws.com" "api.ipify.org" "ifconfig.me/ip"
"icanhazip.com" "ipinfo.io/ip" "ipecho.net/plain"
"checkipv4.dedyn.io")

CURL=`which curl`
DIG=`which dig`

check=$($DIG +short myip.opendns.com @resolver1.opendns.com A)

if [ ! $? -eq 0 ] || [ -z "$check" ] || [[ ! $check =~ ^[0-9]+\.[0-
9]+\.[0-9]+\.[0-9]+$ ]]; then
    echo "Unable to get your public IP address by OpenDNS service, try
to another way."
    count=${#hosts[@]}

    while [ -z "$check" ] && [[ $count -ne 0 ]]; do
        selectedhost=${hosts[ $RANDOM % ${#hosts[@]} ]}
        check=$($CURL -4s https://$selectedhost | grep '[^[:blank:]]')
&& {
            if [ -n "$check" ] && [[ $check =~ ^[0-9]+\.[0-9]+\.[0-
9]+\.[0-9]+$ ]]; then
                break
            else
                check=""
                count=$(expr $count - 1)
```

```
                echo "The host $selectedhost returned an invalid IP
address."
            fi
        } || {
            check=""
            count=$(expr $count - 1)
            echo "The host $selectedhost did not respond."
        }
    done
fi

if [ -z "$check" ]; then
    echo "Unable to get your public IP address. Please check your
internet connection."
    exit 1
fi

echo "Your public IP address is $check"

exit 0
```

上述脚本可以看出首先我使用dig方式查询OpenDNS，如果查询失败或者返回为空或者不是IP地址，那么进入cURL模式，这里将可用服务器添加到hosts列表，并且随机抽取一个进行cURL，这里使用了`curl -4s`参数`-4s`分别表示仅使用IPv4方式连接（确保获取IPv4地址）和抑制进度条和错误信息，如果抽取的服务发生错误，那么进入循环再抽取一个，直到循环满最大hosts数停止。

3 总结

大部分脚本甚至一些程序仅使用了一种方式获取公网IP，但是互联网上这些服务往往是不可靠的，如果你所选用的服务出现故障，那么将会影响到你后续业务的开展，所以本文的核心思想还是多个备份，另外对于cURL方式采用随机确保所谓的“负载平衡”，避免fallback时过度请求某个服务导致IP被Ban。

External References (Links)

checkip.amazonaws.com
https://checkip.amazonaws.com
curl
https://curl.se
《curl 的用法指南》
https://www.ruanyifeng.com/blog/2019/09/curl-reference.html