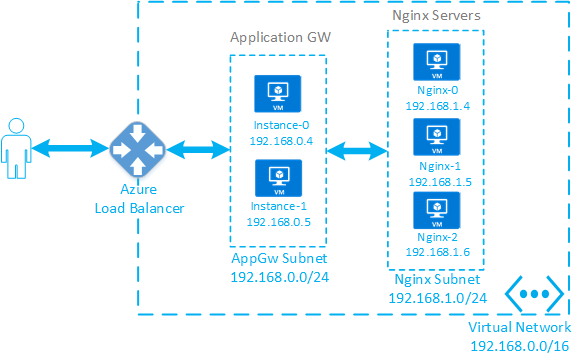
如何在使用应用程序网关和Nginx的环境下实现强制HTTPS跳转

**背景介绍**

大家在使用Nginx部署网站时，实现HTTP到HTTPS的强制跳转是非常容易的事情，一般可以使用rewrite命令或者使用返回自定义301页面的方法对HTTP请求进行HTTPS重定向。如果大家把Nginx服务器部署在Azure应用程序网关后端时，会发现如果原封不动的采取原先的方法进行重定向就无法正常工作。本文通过讲述Azure应用程序网关的工作原理，向大家介绍一种在使用应用程序网关和Nginx的环境下实现强制HTTPS跳转的方法。

**应用程序网关工作原理**

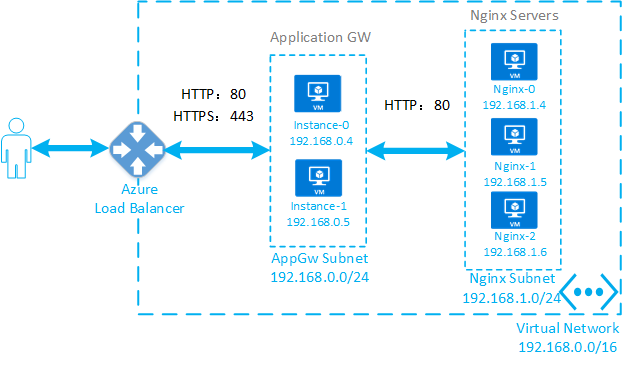
Azure应用程序网关采用的是类似于反向代理服务器的工作方式，客户端直接访问应用程序网关的公网地址而无法感知后端实际的服务器。当应用程序网关接收到客户端的请求之后，它会以自身实例在虚拟网络内部的地址作为源地址对后端池内部的服务器发起新的请求来获取数据，再将获取的数据通过原先跟客户端建立的连接返回给客户端。

如上图所示，我们部署了三台Nginx 服务器在应用程序网关后端。应用程序网关部署在AppGw 子网之内，Nginx服务器部署在Nginx 子网之内。应用程序网关本质上是由多个虚拟机实例组成的群集，默认情况下建立的应用程序网关包含两个实例，每个实例都会占用子网内的一个地址，如上图所示实例-0占用地址192.168.0.4，实例-1占用地址192.168.0.5。当客户端对应用程序网关发起请求时，Azure前端的负载均衡器会将请求发送到对应的应用程序网关实例，应用程序网关上面的服务会以实例本身的内网地址向后端服务器发起新的请求。比如客户端的请求被发送到实例-0，该实例会以192.168.0.4作为源地址对后端服务器发起请求。

**应用程序网关转发机制**

由于应用程序网关向后端服务器发起的请求改变了客户端请求的源地址，所以从TCP层面来讲，客户端的信息会丢失，但是从HTTP层面来讲这个信息并未丢失。这主要是因为应用程序网关会在HTTP包头内部添加X-Forwarded字段来记录客户端访问的源IP、源端口、访问协议和请求的目的地址端口。

假设我们成功部署下面的环境，应用程序网关前端开启80端口走HTTP协议， 443端口走HTTPS协议（同时完成SSL卸载），后端只开启80端口走HTTP协议（目前中国区Azure 应用程序网关在后端只支持HTTP协议）。



当客户端通过80端口使用HTTP协议访问时，如果我们从后端服务器上面抓包，我们可以看到HTTP包头的X-FORWARDED-PROTO为HTTP, X-FORWARDED-PORT 为80，同时也会在X-Forwarded-For部分记录访问者的源IP和源端口（实际IP以x.x.x.x代替）。

Frame: Number = 32, Captured Frame Length = 649, MediaType = ETHERNET

+ Ethernet: Etype = Internet IP (IPv4),DestinationAddress:[00-17-FA-00-73-C4],SourceAddress:[44-03-A7-A4-3F-41]

+ Ipv4: Src = 192.168.0.4, Dest = 192.168.1.4, Next Protocol = TCP, Packet ID = 32620, Total IP Length = 635

+ Tcp: Flags=...AP..., SrcPort=58266, DstPort=HTTP(80), PayloadLen=595, Seq=518190070 - 518190665, Ack=447095254, Win=515 (scale factor 0x8) = 131840

- Http: Request, GET /

Command: GET

+ URI: /

UserAgent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/55.0.2883.87 Safari/537.36

Upgrade-Insecure-Requests: 1

**X-FORWARDED-PROTO: http**

**X-FORWARDED-PORT: 80**

X-Original-URL: /

**X-Forwarded-For: x.x.x.x:64397**

X-ARR-LOG-ID: 3f7b9b21-a862-4b62-a850-dc378bc3c843

HeaderEnd: CRLF

当客户端通过443端口使用HTTPS协议访问时，从下面的抓包可以看到应用程序网关完成了SSL卸载之后仍然是通过HTTP 80端口来访问后端服务器，但是会在HTTP包头的X-FORWARDED-PROTO记录原始访问是HTTPS请求以及原始访问是请求服务器的443端口。

Frame: Number = 36, Captured Frame Length = 700, MediaType = ETHERNET

+ Ethernet: Etype = Internet IP (IPv4),DestinationAddress:[00-17-FA-00-73-C4],SourceAddress:[44-03-A7-A4-3F-41]

+ Ipv4: Src = 192.168.0.4, Dest = 192.168.1.4, Next Protocol = TCP, Packet ID = 15961, Total IP Length = 686

+ Tcp: Flags=...AP..., SrcPort=57271, **DstPort=HTTP(80**), PayloadLen=646, Seq=375761782 - 375762428, Ack=1033546289, Win=511

- Http: Request, GET /

Command: GET

+ URI: /

UserAgent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/55.0.2883.87 Safari/537.36

Upgrade-Insecure-Requests: 1

**X-FORWARDED-PROTO: https**

**X-FORWARDED-PORT: 443**

X-Original-URL: /

**X-Forwarded-For: x.x.x.x:64390**

X-ARR-SSL: 2048|256|CN=FredTest|CN=FredTest

X-ARR-LOG-ID: e93508a5-5242-4f0d-a778-84630f71541b

HeaderEnd: CRLF

通过上面的分析，我们可以看到客户端的HTTP和HTTPS请求到达后端服务器都是HTTP请求，但是应用程序网关会在HTTP包头的X-FORWARDED-PROTO和X-FORWARDED-PORT中对这两种请求加以区分。

**配置Nginx实现强制HTTPS转换**

在未使用应用程序网关的场景下，我们一般可以在Nginx配置里使用类似于下面的rewrite命令来重定向客户端的HTTP请求。

server {

listen 80;

server\_name www.abc.com

**rewrite ^/(.\*) https://$host$1 permanent;**

}

但是在使用应用程序网关的场景下会发生问题，因为Nginx服务器本身虽然将这个请求重写为HTTPS请求，但是当新的HTTPS的请求再次经过应用程序网关到达后端服务器时又会变为HTTP请求从而导致死循环。解决此问题的办法就是需要在后端服务器上来判断客户的原始请求时HTTP还是HTTPS，当客户的原始请求是HTTP时进行重定向，而当客户的原始请求是HTTPS时不进行重定向。通过前面的分析，我们已经知道可以通过HTTP包头的X-FORWARDED-PROTO来判断客户端的实际请求，而Nginx又会将字段X-FORWARDED-PROTO的值保存在变量$http\_x\_forwarded\_proto中，这样我们就可以加入类似下面的判断语句来实现强制HTTPS跳转了。

server {

listen 80;

server\_name www.abc.com

**if ($http\_x\_forwarded\_proto = 'http')**

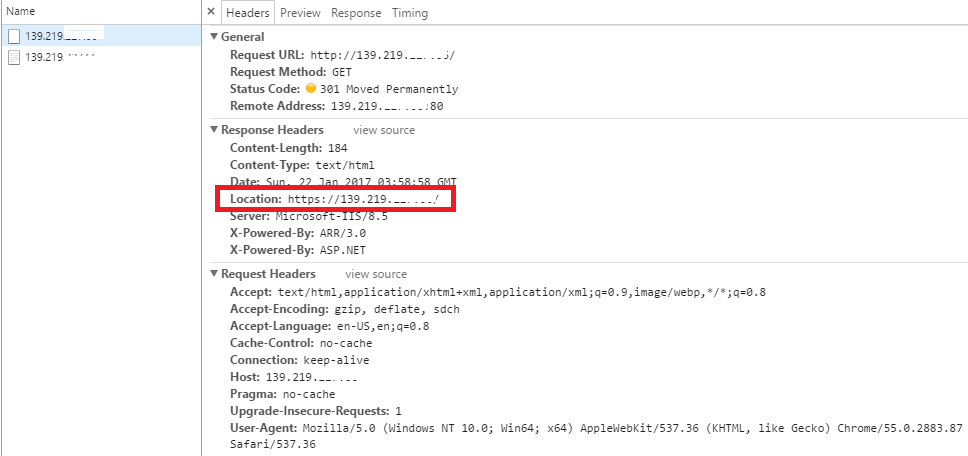
**{**

**rewrite ^/(.\*) https://$host$1 permanent;**

**}**

}

通过这样的设置，当访问网站时我们可以看到服务器端返回了301并请求跳转到HTTPS。



客户端再次通过HTTPS访问时，服务端不再返回301而是返回200表示访问成功。

