# 代理

## 作用

* **访问控制**：出口代理
* **内容路由**：出口代理
* **负载均衡**：反向代理
* **安全**：路由代理
* **缓存**

## 如何将流量导向代理

* **浏览器设置**：手动设置、pac、WPAD
* **路由器设置**：透明拦截
* **反向代理设置**：反向代理直接使用web服务器的ip和port
* **http重定向**：web server发送305响应，将客户端重定向到代理

## http代理请求

* 显示代理请求中url必须是原始web server的绝对URL
* 使用显示代理时，用户的URL会被直接发送给代理，浏览器不会对URL执行自动扩展。

## 代理跟踪

* **via首部**：当报文在通过代理和网关时，代理和网关应该将自己的相关信息添加到via首部的末尾。
* **TRACE方法**：跟踪代理或网关对报文内容做了哪些处理，可以跟踪整条收发的环回链路。

# 网关

* 协议网关
* 安全网关：http/https
* 客户端安全加速器网关：https/http
* 应用程序网关（资源网关）：CGI、ASP

# Web隧道

Web隧道是通过CONNECT方法创建的。CONNECT方法请求**隧道网关**创建一条到达任意目的服务器和端口的**TCP连接**，并对**客户端和服务器之间**的后续数据进行**盲转发**。

## 为什么使用Web隧道

https中，客户端发送的https流量是ssl加密的，这样非https代理就无法解析http报文，从而将报文直接丢弃。为了解决这个问题，在HTTP 1.1中加入了扩展方法CONNECT。当客户端给隧道代理发送CONNECT方法请求建立一条到https服务器（**目的端口443**）的TCP连接时，隧道网关会和https服务器建立TCP连接（即SSL连接），然后发送200响应给客户端。客户端收到后直接发送SSL流量给隧道网关，隧道网关收到后不做任何解析（即不会尝试去解析http报文头）立即将数据转发出去，这样客户端产生的原始SSL加密数据就完整地发送到了隧道网关和https服务器之间的ssl连接上了。

相比于http/https安全网关，web隧道是有

一定优势的。但是web隧道也存在一些问题，比如说，隧道建立好后，隧道网关是无法验证目前使用的协议是否就是它原本打算经过隧道传输的协议。