## **限制请求方法**

**【1】禁用所有不需要的 HTTP 方法**

禁用所有不需要的 HTTP 方法，下面设置意思是只允许 GET、HEAD、POST 方法，过滤掉 DELETE 和 TRACE 等方法。

|  |
| --- |
| location / {  limit\_except GET HEAD POST { deny all; }  } |

### 隐藏不必要的信息

由于某些 Nginx 漏洞只存在于特定的版本，隐藏版本号可以提高安全性

|  |
| --- |
| server\_tokens off; |

## **Header头设置**

另一种方法是在 server 块 设置，不过这样是全局设置的，要注意评估影响

|  |
| --- |
| if ($request\_method !~ ^(GET|HEAD|POST)$ ) {  return 444; }  if ($http\_range ~ "\d{9,}") {      return 444;  } |

<https://zhuanlan.zhihu.com/p/336489487>

**【1】X-Frame-Options**

可以使用 X-Frame-Options HTTP 响应头指示是否应允许浏览器在 <frame> 或 <iframe> 中呈现页面。 这样可以防止点击劫持攻击。

配置文件中添加：

|  |
| --- |
| add\_header X-Frame-Options "SAMEORIGIN"; |

X-Frame-Options三个参数:

**X-Frame-Options：** 响应头表示是否允许浏览器加载frame等属性，有三个配置DENY禁止任何网页被嵌入,SAMEORIGIN只允许本网站的嵌套,ALLOW-FROM允许指定地址的嵌套

换一句话说，如果设置为DENY，不光在别人的网站frame嵌入时会无法加载，在同域名页面中同样会无法加载。另一方面，如果设置为SAMEORIGIN，那么页面就可以在同域名页面的frame中嵌套。正常情况下我们通常使用SAMEORIGIN参数

测试网站是否设置了X-Frame-Options

将如下的代码中iframe中的链接换成待测网站的，保存为.html文件，本地打开。如果打开的页面中显示了待测的网站，则说明没有设置，反之设置了。

|  |
| --- |
| <!DOCTYPE html>  <html>  <head lang="en">  <meta charset="UTF-8" >  <title>点击劫持测试</title>  </head>  <body>  <iframe src="http://www.xxx.com/" width="500" height="500" frameborder="10"> </iframe>  </body>  </html> |

【2】CSP

Content Security Policy (CSP) 保护你的网站避免被使用如 XSS，SQL注入等手段进行攻击

http://www.ruanyifeng.com/blog/2016/09/csp.html

http://www.javashuo.com/article/p-mwgtvopf-p.html

https://blog.csdn.net/m0\_46266503/article/details/108500504

|  |
| --- |
| add\_header Content-Security-Policy "default-src 'self' \*.wetm.top 'unsafe-inline' 'unsafe-eval' blob: data: ;";  add\_header X-Xss-Protection "1;mode=block";  add\_header X-Content-Type-Options nosniff; |

unsafe-inline：允许执行页面内嵌的&lt;script>标签和事件监听函数

unsafe-eval：允许将字符串当作代码执行，比如使用eval、setTimeout、setInterval和Function等函数

**X-XSS-Protection：** 表示启用XSS过滤（禁用过滤为X-XSS-Protection: 0），mode=block表示若检查到XSS攻击则停止渲染页面

**X-Content-Type-Options：** 响应头用来指定浏览器对未指定或错误指定Content-Type资源真正类型的猜测行为，nosniff 表示不允许任何猜测

|  |
| --- |
| 一个网站管理者想要全部内容均来自站点的同一个源 (不包括其子域名)  Content-Security-Policy: default-src 'self' |

|  |
| --- |
| 一个网站管理者容许内容来自信任的域名及其子域名 (域名没必要须与CSP设置所在的域名相同)  Content-Security-Policy: default-src 'self' \*.trusted.com |

|  |
| --- |
| 一个网站管理者容许网页应用的用户在他们本身的内容中包含来自任何源的图片, 可是限制音频或视频需从信任的资源提供者(得到)，全部脚本必须从特定主机服务器获取可信的代码.  Content-Security-Policy: default-src 'self'; img-src \*; media-src media1.com media2.com; script-src userscripts.example.com |

【3】Strict-Transport-Security

HTTP Strict Transport Security，简称为 HSTS。它允许一个 HTTPS 网站，要求浏览器总是通过 HTTPS 来访问它，同时会拒绝来自 HTTP 的请求，操作如下：

add\_header Strict-Transport-Security "max-age=31536000; includeSubdomains; preload";

## **图片防盗链**

|  |
| --- |
| location /images/ {  valid\_referers none blocked www.ops-coffee.cn ops-coffee.cn;  if ($invalid\_referer) {  return 403;  }  } |

**valid\_referers：** 验证referer，其中none允许referer为空，blocked允许不带协议的请求，除了以上两类外仅允许referer为www.ops-coffee.cn或ops-coffee.cn时访问images下的图片资源，否则返回403

## **过滤非法的UA**

编辑配置文件nginx.conf，在添加如下内容：

|  |
| --- |
| if ($http\_user\_agent ~\* "java|python|perl|ruby|curl|bash|echo|uname|base64|decode|md5sum|select|concat|httprequest|httpclient|nmap|scan" ) {  return 403;  } |

## **过滤不支持的URL**

编辑配置文件nginx.conf，在添加如下内容：

|  |
| --- |
| location ~\* \.(bak|save|sh|sql|mdb|svn|git|old|txt|doc|gz)$ {  rewrite ^/(.\*)$ $host permanent;} |

知名程序,比如phpmyadmin

|  |
| --- |
| location /(admin|phpadmin|status) {  deny all; } |

server\_name  \_;#\_代表任意地址

### 配置/etc/sysctl.conf强化Linux安全

可以通过编辑/etc/sysctl.conf来控制和配置Linux内核、网络设置。

# Avoid a smurf attack

net.ipv4.icmp\_echo\_ignore\_broadcasts = 1# Turn on protection for bad icmp error messages

net.ipv4.icmp\_ignore\_bogus\_error\_responses = 1# Turn on syncookies for SYN flood attack protection

net.ipv4.tcp\_syncookies = 1# Turn on and log spoofed, source routed, and redirect packets

net.ipv4.conf.all.log\_martians = 1

net.ipv4.conf.default.log\_martians = 1# No source routed packets here

### url 参数过滤敏感字

if ($query\_string ~\* "union.\*select.\*\(") {

rewrite ^/(.\*)$ $host permanent; }

if ($query\_string ~\* "concat.\*\(") {

rewrite ^/(.\*)$ $host permanent; }

### 根据用户的真实 IP 做连接限制

## 这里取得原始用户的IP地址

map $http\_x\_forwarded\_for $clientRealIp {

"" $remote\_addr;

~^(?P<firstAddr>[0-9\.]+),?.\*$ $firstAddr;}

## 针对原始用户 IP 地址做限制

limit\_conn\_zone $clientRealIp zone=TotalConnLimitZone:20m ;

limit\_conn TotalConnLimitZone 50;

limit\_conn\_log\_level notice;

### 隐藏版本信息

server\_tokens off;

proxy\_hide\_header X-Powered-By;

### 目录只读

如果没有上传需求，完全可以把网站根目录弄成只读的，加固安全。

做了一点小动作，给网站根目录搞了一个只读的挂载点。这里假设网站根目录为/var/www/html

mkdir -p /data

mkdir -p /var/www/html

mount --bind /data /var/www/html

mount -o remount,ro --bind /data /var/www/html

网站内容实际位于/data，网站内容更新就往/data里更新，目录/var/www/html无法执行任何写操作，否则会报错“Read-only file system”，极大程度上可以防止提权篡改。

**三、限定非法文件访问**

**名称：**限定非法文件访问

**实施目的：**禁止nginx访问某类型的文件。

**问题影响：**非法访问，恶意攻击。

**实施步骤：**编辑nginx.conf配置文件

如，Nginx下禁止访问\*.txt文件

|  |
| --- |
| location ~\* \.(txt|doc)${  root /var/InfiNET/web /test;  deny all;  } |

禁止下载以点开头的文件

|  |
| --- |
| location ~ /\..+ {  deny all;  } |

### url 参数过滤敏感字

if ($query\_string ~\* "union.\*select.\*\(") {

rewrite ^/(.\*)$ $host permanent; }

if ($query\_string ~\* "concat.\*\(") {

rewrite ^/(.\*)$ $host permanent; }