针对防火墙HTTP严格传输安全保障、Http响应头缺失 、CSRF漏洞说明

漏洞相关 吴昊A 2022-04-24 发表

漏洞相关信息

漏洞编号： 不涉及

漏洞名称： HTTP严格传输安全保障、Http响应头缺失 、CSRF漏洞

产品型号及版本： V7安全设备

漏洞描述

漏洞一：检测到缺少Strict-Transport-Security响应头

【原理介绍】

Strict-Transport-Security响应头是要求客户端（即PC浏览器）禁用HTTP协议。尝试后续都通过HTTPS协议进行通信。

由于HTTPS协议存在加密层，本身更安全，故该响应头也是为了安全考虑，希望服务端能返回。

漏洞二：检测到缺少Content-Security-Policy响应头

【扫漏原理】

扫漏软件，应该只检测HTTP响应报文中是否存在Content-Security-Policy字段。并不关心其他防范XSS注入的操作。

漏洞三：CSRF漏洞

【CSRF漏洞简述】

用户登录了我司设备网址，COOKIE中记录了我司的session信息（第一步）

------>我司设备网站没有退出时，用户登陆了钓鱼网站B。（第二步）

------->钓鱼网站B伪造了一个类似我司设备网站的请求，并发送给了我司设备网站，请求的URL是我司域名，请求会携带我司设备网站的COOKIE----这是浏览器客户端的行为。（第三步）

------>我司网站如果没有对CSRF漏洞进行规避，这样就会认为，第三步发送的请求是个正常合法的请求，并响应第三步的请求。（第四步）

至此，从上面四步，就进行了一次CSRF的攻击。

漏洞解决方案

以上漏洞我司均不涉及。详细解释如下：

漏洞一：检测到缺少Strict-Transport-Security响应头

【原理介绍】

Strict-Transport-Security响应头是要求客户端（即PC浏览器）禁用HTTP协议。尝试后续都通过HTTPS协议进行通信。

由于HTTPS协议存在加密层，本身更安全，故该响应头也是为了安全考虑，希望服务端能返回。

【扫漏原理】

扫漏软件应该只检测HTTP响应报文中是否存在Strict-Transport-Security该响应报文，并不关心设备侧具体配置。所以报了这个漏洞

【规避措施】

从该报文实现原理上可以有如下解释：

由于现场设备本身不开启HTTP，只开启HTTPS。本身已经可以确保所有请求通过HTTPS协议进行传输。所以该报文头即使不返回，也不会有安全隐患。

漏洞二：检测到缺少Content-Security-Policy响应头

【原理介绍】

Content-Security-Policy又称内容安全策略，实际上就是告诉客户端（即PC浏览器）什么行为是被授权的，什么行为是被禁止的。主要用来解决XSS注入漏洞。

他的响应内容主要包括如下表格

指令名 Demo 说明

default-src 'self' cdn.example.com 默认策略,可以应用于js文件/图片/css/ajax请求等所有访问

script-src 'self' js.example.com 定义js文件的过滤策略

style-src 'self' css.example.com 定义css文件的过滤策略

img-src 'self' img.example.com 定义图片文件的过滤策略

connect-src 'self' 定义请求连接文件的过滤策略

font-src font.example.com 定义字体文件的过滤策略

object-src 'self' 定义页面插件的过滤策略,如 <object>, <embed> 或者<applet>等元素

media-src media.example.com 定义媒体的过滤策略,如 HTML6的 <audio>, <video>等元素

frame-src 'self' 定义加载子frmae的策略

sandbox allow-forms allow-scripts 沙盒模式,会阻止页面弹窗/js执行等,你可以通过添加allow-forms allow-same-origin allow-scripts allow-popups, allow-modals, allow-orientation-lock, allow-pointer-lock, allow-presentation, allow-popups-to-escape-sandbox, and allow-top-navigation策略来放开相应的操作

report-uri /some-report-uri 指定报告发送的地址

从上面可以看到，这个响应内容包括了基本上所有页面上可以使用的资源。

响应报文的值：大多数都是两个选择（1）站点内部加载，（2）某个域名下加载。

目的：为了解决XSS注入攻击

【XSS注入攻击】

简单介绍一下XSS注入攻击。

XSS注入攻击可以解释为：攻击者在本网站站点，通过篡改客户端代码，或者引用外部代码，向服务端或数据库中注入一段攻击代码。

当攻击代码再次被请求或者，在服务端运行时，达到攻击效果。

【扫漏原理】

扫漏软件，应该只检测HTTP响应报文中是否存在Content-Security-Policy字段。并不关心其他防范XSS注入的操作。

【解释】

1. 首先我司设备之前已经分析过，XSS注入攻击时不生效的。注入代码不会执行。达不到攻击动作。
2. 其次，为了防止XSS注入攻击，我司设备也返回了响应头X-XSS-Protection要求客户端开启XSS防护器。

漏洞三：CSRF漏洞

【CSRF漏洞简述】

用户登录了我司设备网址，COOKIE中记录了我司的session信息（第一步）

------>我司设备网站没有退出时，用户登陆了钓鱼网站B。（第二步）

------->钓鱼网站B伪造了一个类似我司设备网站的请求，并发送给了我司设备网站，请求的URL是我司域名，请求会携带我司设备网站的COOKIE----这是浏览器客户端的行为。（第三步）

------>我司网站如果没有对CSRF漏洞进行规避，这样就会认为，第三步发送的请求是个正常合法的请求，并响应第三步的请求。（第四步）

至此，从上面四步，就进行了一次CSRF的攻击。

【CSRF漏洞业界方案】

方案一：采用HTTP报文中的Referer字段，处理CSRF漏洞攻击。 -------- 我司采用

1. HTTP中的referer字段，标明了请求的来源地址。HTTP协议规范中说明，该字段是由客户端自动生成，不允许人为修改
2. 例如：上述攻击的第三步，当钓鱼网站B，伪造了一个攻击请求，此时该攻击请求的HTTP报文中，携带的referer字段为钓鱼网站B的域名地址。而该攻击请求的URL为我司域名地址
3. 我司设备HTTP服务器，对于请求会校验请求的URL和referer是否相同。当referer和URL相同时，我认为当前请求来源是我司设备网站。当referer和URL不同时，我认为请求来源为第三方网站，并拒绝该请求的回复

方案二：采用携带随机token值的方案，对请求进行检测。 ------方案二有很多种不同的思路，下述大概思路流程

1. 我司设备的HTTP 服务器在生成session信息的时候，同时对session绑定一个随机生成的tok