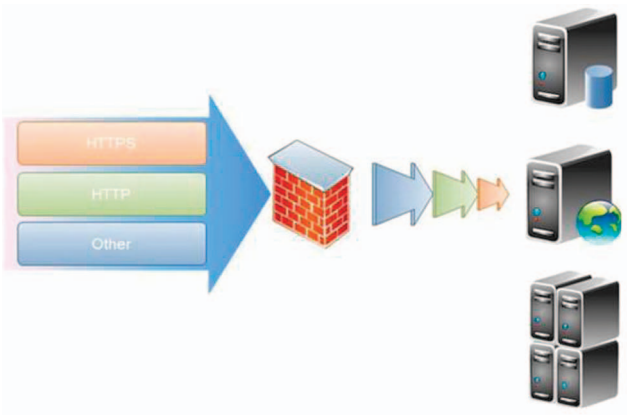
WAF HTTP协议校验浅析

武汉研发中心 彭元

HTTP 协议的合规性一直是 WAF（Web 应用防护系统）的基本功能，这是要求用户使用的 HTTP 协议必须符合 RFC 标准，但是由于用户的网站开发千秋各异，或者过于陈旧，很多网站都无法满足 WAF HTTP 协议合规的要求，本文简要分析了 WAF HTTP 协议校验功能。

大家知道 WAF 通常是串联部署在 Web 服务器前端，对于服务器和客户端都是透明的，采用的是反向代理模式的架构，且仅处理与 Web 应用相关的协议，其他的则进行转发，如下图：



WAF 系统内嵌的应用层协议栈是经过修改和优化的，能完全支持 Http(s) 应用协议的处理，这意味着必须遵循 RFC 标准（Internet Requests For Comments）来处理 Http(s) 报文，包括如下主要 RFC：

- RFC 2616 HTTP 协议语法的定义
- RFC 2396 URL 语法的定义
- RFC 2109 Cookie 是怎样工作的
- RFC 1867 HTTP 如何 POST，以及 POST 的格式

WAF 产品能完全遵循 RFC HTTP/1.1 HTTP/1.0 HTTP/0.9 协议来处理 Web 应用的报文。RFC 中对 Http 的 request 行长度、URL 长度、协议名称长度、头部值长度等都是有严格要求的，以及传输顺序和应用格式，比如 Html 参数的要求（长度，个数等）、Cookie 的版本和格式、文件上传的编码 multipart/form-data encoding 等，这些应用层内容只能在具有完整应用层协议栈的前提下才可正确识别和控制，对于不完整的丢包、重传包以及伪造的畸形包都会通过协议校验机制来处理。

那么什么是协议校验呢?

众所周知，Web 应用都是严格按照 HTTP 协议进行数据交换的，

所以在客户端传输到服务器端的 HTTP 报文必须是符合 HTTP 协议的。否则，Web 服务器可能不能解析 HTTP 请求，造成服务器的资源开销，甚至攻击者可能利用 Web 服务器某些版本的漏洞，构造一些恶意的报文，对 Web 服务器实施攻击。所以，WAF 需要对 HTTP 报文进行 HTTP 协议校验，检测报文是否符合 HTTP 协议标准，若不符合可以执行相应的动作。

WAF 产品应具有完备的 HTTP 协议校验功能，分为解码和策略校验。其中解码是 WAF 按照 HTTP 协议去理解接收到的报文数据。一些明显违背 HTTP 协议的流量，比如请求方法错误或 HTTP 协议版本号不对等，这些流量是不能被 Web 服务器解析的，WAF 在处理这些流量的过程中就会出现解码失败，WAF 引擎在 HTTP 解码过程中会将这些报文直接阻断。这部分阻断的原因包括以下类别：

而对于没有明显违背 HTTP 协议，但是请求头部超长或包含异常字符等，可能造成 Web 服务器缓冲区溢出、拒绝服务等攻击，也可能造成绕过 WAF 的一些安全检测机制，使得 WAF 形同虚设，所以 WAF 还提供了 HTTP 协议校验策略，来防止这些没有明显违背 HTTP 协议格式但危险的流量。因为这部分检测与 Web 服务器本身或站点本身相关性比较大，所以需要用户为防护的站点配置 HTTP 协议校验策略才能生效。这部分可

WAF 内置错误码	阻断原因
HTTP_BAD_REQUEST	request could not be understood
HTTP_METHOD_NOT_ALLOWED	request method is not allowed
HTTP_LENGTH_REQUIRED	request method requires a valid Content-length
HTTP_REQUEST_ENTITY_TOO_LARGE	request exceeds system's limit
HTTP_PROTOCOL_VERSION_FORBIDDEN	request use forbidden http protocol version
HTTP_REQUEST_BAD_METHOD	the requested method is unknown
HTTP_REQUEST_BAD_URI	request with invalid uri
HTTP_REQUEST_BAD_SCHEMA	request with bad schema
HTTP_REQUEST_BAD_PROTOCOL	request with bad protocol
HTTP_REQUEST_BAD_PROTOCOL_VERSION	request with bad protocol version
HTTP_REQUEST_BAD_CRLF	request end failed, bad CRLF
HTTP_RESPONSE_ENTITY_TOO_LARGE	response exceed system's limit

表 1 WAF 解码协议校验错误码

配置的策略校验类别包括如 65 页表 2。

名称	说明	默认值	告警信息 / 备注	检测点
URI 最大长度	指统一资源标识符，即在浏览器地址栏中显示的地址。	4096	URI_LENGTH_TOO_LARGE	解码中
GET 请求参数最大个数	指在 Get 请求时，URI 的查询字符串中所包含的参数个数。	20	URI_ARG_COUNT_TOO_LARGE	插件中
User-Agent 最大长度	指 User-Agent 头部值的长度。	1024	USER_AGENT_LENGTH_TOO_LARGE	解码中
Cookie 最大长度	指整个 Cookie 头部值的长度。	1024	COOKIE_LENGTH_TOO_LARGE	解码中
Cookie 最大个数	指 Cookie 头部中包含的 Cookie 个数。	64	COOKIE_COUNT_TOO_LARGE	插件中
Referer 最大长度	指 Referer 头部值的长度。	4096	REFERER_LENGTH_TOO_LARGE	解码中
Accept 最大长度	指 Accept 头部值的长度。	1024	ACCEPT_LENGTH_TOO_LARGE	解码中
Accept-Charset 最大长度	指 Accept-Charset 头部值的长度。	128	ACCEPT_CHARSET_LENGTH_TOO_LARGE	解码中
Content-Length 最大值	指 Content-Lengt 头部值的最大数字，实质上是 POST 请求的最大 body 长度。	10485760	REQUEST_CONTENT_LENGTH_TOO_LARGE	解码中
最大 Range 区间个数	指 Range 头部值中允许出现的分段区间个数。	5	RANGE_COUNT_TOO_LARGE	插件中
Range 最大跨度	指 Range 头部值中每个分段区间最大范围	5242880	URI_LENGTH_TOO_LARGE	插件中
HTTP 头部最大个数	指 HTTP 头部个数	32	HTTP_HEADER_COUNT_TOO_LARGE	解码中
HTTP 头部字段名最大长度	指 HTTP 头部名的最大长度	64	HTTP_HEADER_NAME_LENGTH_TOO_LARGE	解码中
HTTP 头部值最大长度	指 HTTP 头部值的最大长度	128	这里的头部，不包括前面单独设置的头部，比如 Referer、Accept。日志中的告警信息为： HTTP_HEADER_VALUE_LENGTH_TOO_LARGE	解码中
POST 请求参数最大个数	指 POST 请求时 URI 的查询字符串中所包含的参数个数加上 HTTP 请求体中的参数个数	256	POST_ARG_COUNT_TOO_LARGE	插件中
禁止重复参数	出现了重复的参数名称，有可能导致 WAF 被绕过。	否	REPEAT_ARG_EXIST 因为检测重复参数的时间复杂度为 $O(n^2)$，参数个数过多时性能消耗较明显，所以默认策略关闭。	插件中
禁止重复 HTTP 头部	出现了重复的 HTTP 头部，有可能导致 WAF 被绕过。	是	REPEAT_HADER_EXIST 因为一般头部个数都较少，所以默认策略开启。	插件中
禁止二次 URL 编码	在 HTTP 会话中出现了 %25 且后面不是紧跟着两个十六进制值时，可能会导致绕过 WAF。	否	DOUBLE_URL_ENCODE_EXIST 该条规则容易误报，尤其是 web 应用中将 Referer 当作参数时。所以默认策略关闭。	解码中
禁止异常主机头端口	主机头端口应与 TCP 连接端口一致，否则可能有安全隐患。	是	ABNORMAL_PORT_EXIST	解码中
禁止非法域名	域名中只能由字母（A ~ Z，a ~ z）、数字（0 ~ 9）和连接符（ ）组成，各级域名之间用实点（.）连接。	是	ABNORMAL_HOST_EXIST	插件中
禁止异常锚点	URI 中的锚点 (#) 不应该发送到服务器端，否则可能有安全隐患。	是	ANCHOR_EXIST	解码中
是否清除异常 %	当 URI 或参数值中包含‘%’，且后面不是紧跟着两个十六进制值时（如 %xy），可能会导致绕过 WAF。	否	异常控制不属于检测项，没有告警	解码中
是否清除空字符	当 URI 或参数值中包含 NULL 字符时（如 \0, %00 等），可能会导致绕过 WAF。	否	异常控制不属于检测项，没有告警	解码中

表 2 WAF 策略校验类别说明

对应的在 WAF 站点防护策略配置中 HTTP 协议校验规则有如下配置界面：

default_low	宽松策略	是
URI最大长度	4096	
禁止异常锚点	是	
禁止异常主机头端口	是	
禁止非法域名	是	
User-Agent最大长度	4096	
Cookie最大长度	4096	
Cookie最大个数	128	
Referer最大长度	4096	
Accept最大长度	4096	
Accept-Charset最大长度	4096	
Content-Length最大值	10485760	
最大Range区间个数	5	
Range最大跨度	5242880	
HTTP头部最大个数	128	
HTTP头部字段名最大长度	128	
HTTP头部值最大长度	128	
禁止重复HTTP头部	是	
GET请求参数最大个数	128	
POST请求参数最大个数	256	
禁止重复参数	是	
禁止二次URL编码	是	
是否清除异常%	否	
是否清除空字符	否	

图 2 WAF HTTP 协议校验防护规则默认宽松策略配置

这么多的协议校验策略到底能起到什么样的防护作用呢？下面我们以实际的攻击案例来一一解读。

(1)WebServer Buffer Overflow

例如 CVE-2013-2028 ngx_http_parse.c 的远程栈缓冲区溢出，当以 Chuncked Encoding 发送的 Chunk Size 为一个特殊值就能造成对 Nginx 的 DoS 或者远程 shellcode 执行，WAF 在解码过程中对包含不合法的 ChunkSize 的 HTTP 请求进行阻断，可以在无需任何升级更新就能防护此类攻击。

(2)HTTP Parameter Pollution

HTTP Parameter Pollution（HPP）使用如下形式的请求能够在不同 Web 应用程序中造成不同的攻击效果，比如：/somePage.jsp?param1=value1&param2=value2。

不同的后端 Web 应用程序对于这里请求的处理不相同，如下表所示：

后端编程语言	后端处理结果	样例
ASP.NET/IIS	取值为列表数组	par1=val1,val2
ASP/IIS	取值为列表数组	par1=val1,val2
PHP/Apache	取值为最后一个值	par1=val2
PHP/Zeus	取值为最后一个值	par1=val2

JSP, Servlet/ Apache Tomcat	取值为第一个值	par1=val1

这样我们就能利用这些特性构造一些绕过 WAF 防护规则的攻击。例如这样一个正常的攻击会被 WAF 拦截：

http://test.com/list.asp?prodID=9 UNION SELECT 1,2,3 FROM Users WHERE id=3 –

而这样的攻击则可能绕过 WAF 的防护:

http://test.com/list.asp?prodID=9 /*&prodID=*/UNION /*&prodID=*/ SELECT 1 &prodID=2 &prodID=3 FROM /*&prodID=*/Users /*&prodID=*/ WHERE id=3 --

但是在开启 WAF 的 HTTP 协议校验策略后，WAF 会对重复参数以及重复参数的个数进行检测，从而使得这样的攻击失效。

(3)Apache httpOnly Cookie Disclosure

Apache 的 Cookie 泄漏漏洞是在 Apache 部分版本中如果 Cookie 的内容大于 4K，那么页面就会返回 400 错误。返回的报错内容却包含 Cookie 信息。攻击者通过设置大于 4K 的 Cookie 让 Apache 报错绕过了 HttpOnly 的保护从而得到 Cookie。在 WAF 的 HTTP 协议校验中可以配置 Cookie 长度限制，告警信息为：COOKIE_LENGTH_TOO_LARGE 从而能阻挡此类攻击。

HTTP 协议校验的安全防护作用还不仅仅是这些，通过对 HTTP 请求的 URI, 参数及参数值，请求头部，请求 body 的各个字段的长度、个数等做规范校验，来阻止一些通用的未知的攻击方法，防止 WAF 防护策略的绕过。针对某些 0day 攻击，还具有无需规则升级就能防护的效果。

协议校验功能既然这么有用，但在实际部署 WAF 产品的时候，我们建议用户根据自身业务选择部分校验策略的放过，这是为了兼容性考虑，不影响用户当前的业务。

WAF 应用层协议处理虽然严格按照 RFC 标准来进行，但在实际环境中，由于部分 Web 应用程序并未完全按照 RFC 标准进行开发或通讯，因此如果 WAF 严格开启协议校验，或导致当前 Web 应用业务的中断。

对于普通的例如 IE 等浏览器而言，它会忽略这种错误，但是对于 WAF 这种企业级的防火墙，安全是最重要的，因此它会默认阻止这种非法的数据包。但是 WAF 阻断的这种安全行为也给用户带来不便。

因此为了兼容性考虑，用户需要根据自身业务需要，配置 WAF 协议校验策略的放过，但是这样也就降低了安全性，因此在做这种配置之前，你需要正确的进行评估。同时，WAF 的协议校验也体现在只处理合法的 HTTP 协议流量，阻断非法的 HTTP 协议流量的解码过程之中。

本文简要分析了 WAF 产品的 HTTP 协议校验功能，它分为解码和策略校验。其中解码是对访问服务器的报文符合 HTTP 协议格式的最基本要求，策略校验是用户对不同站点根据自己的需求自定义一些与自身业务、环境相匹配的协议层检查策略，主要集中在对 Web 服务器缓冲区溢出攻击的防护和绕过 WAF 的防护。HTTP 协议校验对防护 Web 服务器有着重要作用，在实际部署 WAF 产品时，应结合用户的业务需求，建议用户开启 HTTP 协议校验默认策略。