simple

出题角度

这个程序就是简单用java写的一个tea加密判断，然后使用soot将.class转化为.jimple中间代码。没怎么混淆，就是把函数名字改了一下。可能问题在于很长的代码和tea的数据类型是int，拿普遍unsigned int的脚本要变一变?

jimple的代码查看很清晰，基本和高级语言差不了多少(虽然看的心累)，慢慢梳理逻辑也很明确，依次就是BytetoInt，InttoByte，tea_encrypt，transform(有符号char变成无符号的)4个方法，然后main方法输入字符串，然后加密比对一下。其实本意作为签到题，直接给gpt就行，虽然分析的可能有点偏差，但是八九不离十，再拷打拷打，你就差不多知道大概逻辑了，然后套脚本 → 改脚本 → 得到flag。

题目源码

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.Arrays;

public class simple {
    private static int[] byteToInt(byte[] content) {

        int[] result = new int[content.length >> 2];
        for (int i = 0, j = 0; j < content.length; i++, j += 4) {
            result[i] = transform(content[j + 3]) | transform(content[j +
2]) << 8 |
                    transform(content[j + 1]) << 16 | (int) content[j] <<
24;
        }
        return result;
    }

    private static byte[] intToByte(int[] content) {
        byte[] result = new byte[content.length << 2];
        for (int i = 0, j = 0; j < result.length; i++, j += 4) {
            result[j + 3] = (byte) (content[i] & 0xff);
            result[j + 2] = (byte) ((content[i] >> 8) & 0xff);
            result[j + 1] = (byte) ((content[i] >> 16) & 0xff);
            result[j] = (byte) ((content[i] >> 24) & 0xff);
        }
        return result;
    }

    public static byte[] tea_encrypt(byte[] content, int[] MD5) {
        int[] tempInt = byteToInt(content);
        for (int k = 0; k < tempInt.length; k += 2) {
            int v0 = tempInt[k], v1 = tempInt[k + 1];
            int sum = 0;
            int delta = 0x9e3779b9;
            int rounds = 32;
```

```
            for (int i = 0; i < rounds; i++) {
                sum += delta;
                v0 += ((v1 << 4) + MD5[0]) ^ (v1 + sum) ^ ((v1 >> 5) +
MD5[1]);
                v1 += ((v0 << 4) + MD5[2]) ^ (v0 + sum) ^ ((v0 >> 5) +
MD5[3]);
            }
            tempInt[k] = v0;
            tempInt[k + 1] = v1;
        }
        return intToByte(tempInt);
    }

    private static int transform(byte temp) {
        int tempInt = temp;
        if (tempInt < 0) {
            tempInt += 256;
        }
        return tempInt;
    }

    public static void main(String[] args) {
        // SBCTF{have_fun_with_tea}
        int[] MD5 = new int[]{
                0x67452301, 0xefcdab89, 0x98badcfe, 0x10325476
        };
        System.out.println("please input your flag:");
        BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));
        try {
            byte[] message = reader.readLine().getBytes();
            byte[] enc = {73, -65, 27, -19, -77, 28, 108, 82, 43, 60, -14,
58, 28, 44, -21, 77, 31, 114, 43, 98, 88, 17, 23, -9};
            byte[] dec = tea_encrypt(message, MD5);
            if (Arrays.equals(dec, enc))
                System.out.println("right flag!");
            else
                System.out.println("wrong flag!");
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
}
```

soot编译

> 下载地址
>
> 使用参考

这里我使用的是 jdk1.8 来处理的，soot是第一个 sootclasses-trunk-jar-with-dependencies.jar

```
# .java → .class
javac simple.java

# .class → .jimple
java -cp sootclasses-trunk-jar-with-dependencies.jar soot.Main -f J -pp -cp
. simple

# 这里还可以将代码逻辑转化为控制流图，有兴趣可以试一试
# .class → .dot
java -cp sootclasses-trunk-jar-with-dependencies.jar soot.tools.CFGViewer -
pp -cp . Triangle
# .dot → .png
dot -Tpng -o Triangle.png Triangle.dot
```

这里我就是把方法的名称改为method[0……3]了，不然gpt一给就真的全部都出来了

解题角度

建议还是拷打一下gpt。自己分析主要逻辑就是method3的tea加密，里面的逻辑没变动，对着加密脚本可以很清楚的认出来，然后就是改脚本解密。

exp

java版本

```
public class simple2 {
    private static int[] byteToInt(byte[] content) {

        int[] result = new int[content.length >> 2];
        for (int i = 0, j = 0; j < content.length; i++, j += 4) {
            result[i] = transform(content[j + 3]) | transform(content[j +
2]) << 8 |
                    transform(content[j + 1]) << 16 | (int) content[j] <<
24;
        }
        return result;
    }

    private static byte[] intToByte(int[] content) {
        byte[] result = new byte[content.length << 2];
        for (int i = 0, j = 0; j < result.length; i++, j += 4) {
            result[j + 3] = (byte) (content[i] & 0xff);
            result[j + 2] = (byte) ((content[i] >> 8) & 0xff);
            result[j + 1] = (byte) ((content[i] >> 16) & 0xff);
            result[j] = (byte) ((content[i] >> 24) & 0xff);
        }
        return result;
    }
```

```
    public static byte[] decrypt(byte[] encryptContent, int[] MD5) {
        int[] tempInt = byteToInt(encryptContent);
        for (int k = 0; k < tempInt.length; k += 2) {
            int v0 = tempInt[k], v1 = tempInt[k + 1];
            int delta = 0x9e3779b9;
            int rounds = 32;
            int sum = delta * rounds;
            for (int i = 0; i < rounds; i++) {
                v1 -= ((v0 << 4) + MD5[2]) ^ (v0 + sum) ^ ((v0 >> 5) +
MD5[3]);
                v0 -= ((v1 << 4) + MD5[0]) ^ (v1 + sum) ^ ((v1 >> 5) +
MD5[1]);
                sum -= delta;
            }
            tempInt[k] = v0;
            tempInt[k + 1] = v1;
        }
        return intToByte(tempInt);
    }

    private static int transform(byte temp) {
        int tempInt = temp;
        if (tempInt < 0) {
            tempInt += 256;
        }
        return tempInt;
    }

    public static void main(String[] args) {
        int[] MD5 = new int[]{
                0x67452301, 0xefcdab89, 0x98badcfe, 0x10325476
        };
        byte[] enc = {73, -65, 27, -19, -77, 28, 108, 82, 43, 60, -14, 58,
28, 44, -21, 77, 31, 114, 43, 98, 88, 17, 23, -9};
        byte[] decryptInfo = decrypt(enc, MD5);
        String flag = new String(decryptInfo);
        System.out.println(flag);
    }
}
```

python版本

```
import struct
from ctypes import *


def decrypt(v, k):
    for m in range(0, len(v), 2):
        v0, v1 = c_int32(v[m]), c_int32(v[m + 1])
        delta = 0x9e3779b9
        k0, k1, k2, k3 = k[0], k[1], k[2], k[3]

        total = c_int32(delta * 32)
```

```
        for _ in range(32):
            v1.value -= ((v0.value << 4) + k2) ^ (v0.value + total.value) ^
((v0.value >> 5) + k3)
            v0.value -= ((v1.value << 4) + k0) ^ (v1.value + total.value) ^
((v1.value >> 5) + k1)
            total.value -= delta
        v[m], v[m + 1] = v0.value, v1.value
    return v


def byte_to_int(content):
    result = []
    for i in range(0, len(content), 4):
        num = 0
        for j in range(4):
            num = (num << 8) | (content[i + j] & 0xFF)
        result.append(num)
    return result


if __name__ == "__main__":
    enc = [73, -65, 27, -19, -77, 28, 108, 82, 43, 60, -14, 58, 28, 44,
-21, 77, 31, 114, 43, 98, 88, 17, 23, -9]
    value = byte_to_int(enc)
    key = [1732584193, -271733879, -1732584194, 271733878]
    res = decrypt(value, key)
    result = ''
    for i in range(len(res)):
        result += struct.pack('>I', res[i]).decode('ISO-8859-1')
    print(result)
```