

CobaltStrike4.X 之去除 CheckSum8 特征

STATEMENT

声明

由于传播、利用此文所提供的信息而造成的任何直接或者间接的后果及损失，均由使用者本人负责，雷神众测及文章作者不为此承担任何责任。

雷神众测拥有对此文章的修改和解释权。如欲转载或传播此文章，必须保证此文章的完整性，包括版权声明等全部内容。未经雷神众测允许，不得任意修改或者增减此文章内容，不得以任何方式将其用于商业目的。

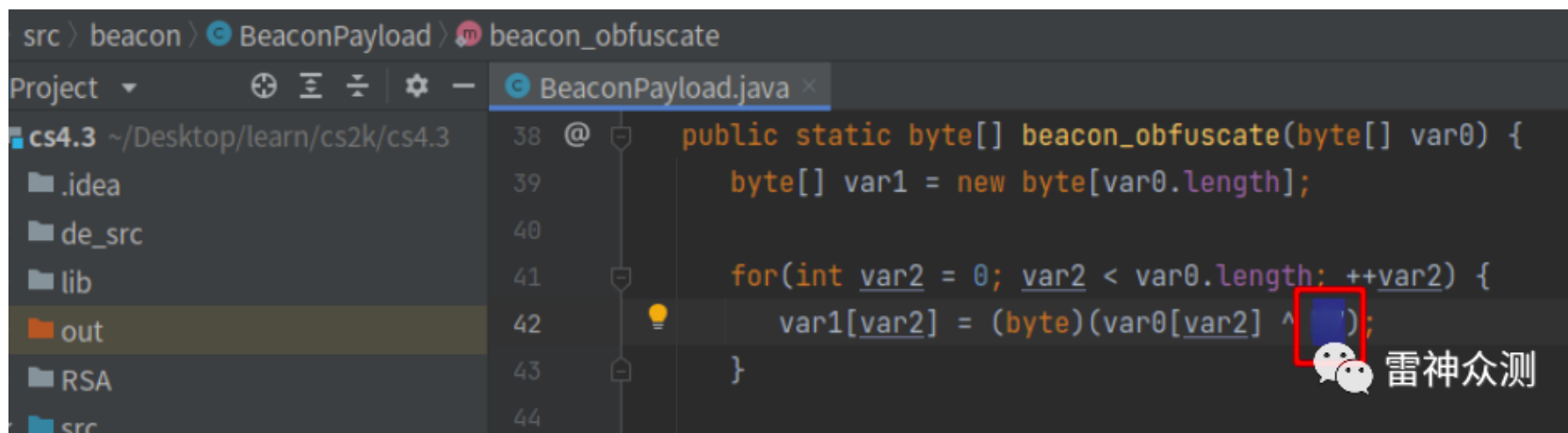
NO.1 前言

之前介绍了 CobaltStrike4.3 的 License 认证分析，今天介绍一下 CobaltStrike4.3 去除 CheckSum8 特征的方法，Checksum8 的特征和具体算法不在此细说，但为了避免被扫描出来还是有必要进行修改的。

NO.2 代码修改

BeaconPayload 中修改异或数值为新：

随便一个 10 进制数字即可，后面 dll 中改成对应的 16 进制数字。



```
src > beacon > BeaconPayload > beacon_obfuscate
Project ▾
cs4.3 ~/Desktop/learn/cs2k/cs4.3
  .idea
  de_src
  lib
  out
  RSA
  src
38 @ public static byte[] beacon_obfuscate(byte[] var0) {
39     byte[] var1 = new byte[var0.length];
40
41     for(int var2 = 0; var2 < var0.length; ++var2) {
42         var1[var2] = (byte)(var0[var2] ^ 0x00000000);
43     }
44 }
```

NO.3 dll 修改

使用 CrackSleeve 将 dll 进行解密:

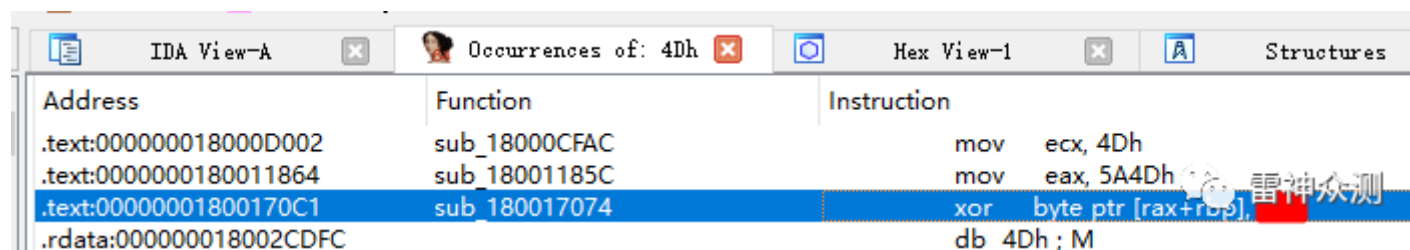
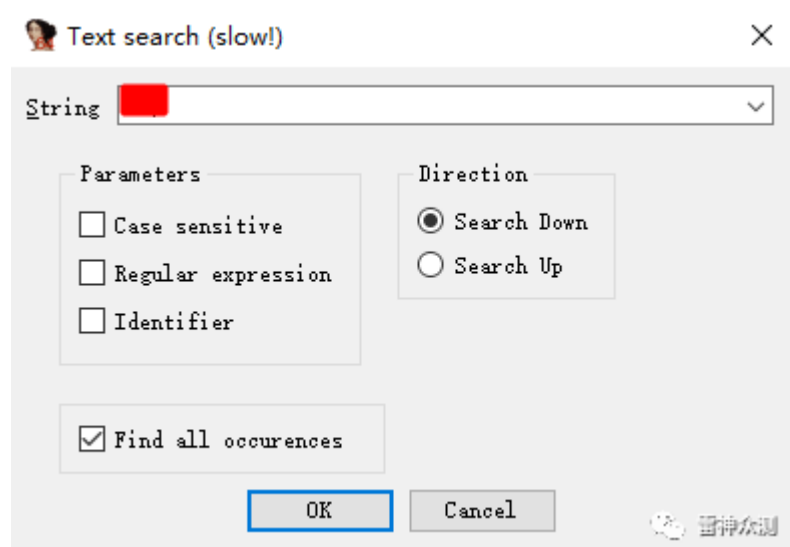
<https://github.com/ca3tie1/CrackSleeve/>

1、将 cobaltstrike.jar 和 CrackSleeve.java 放一起

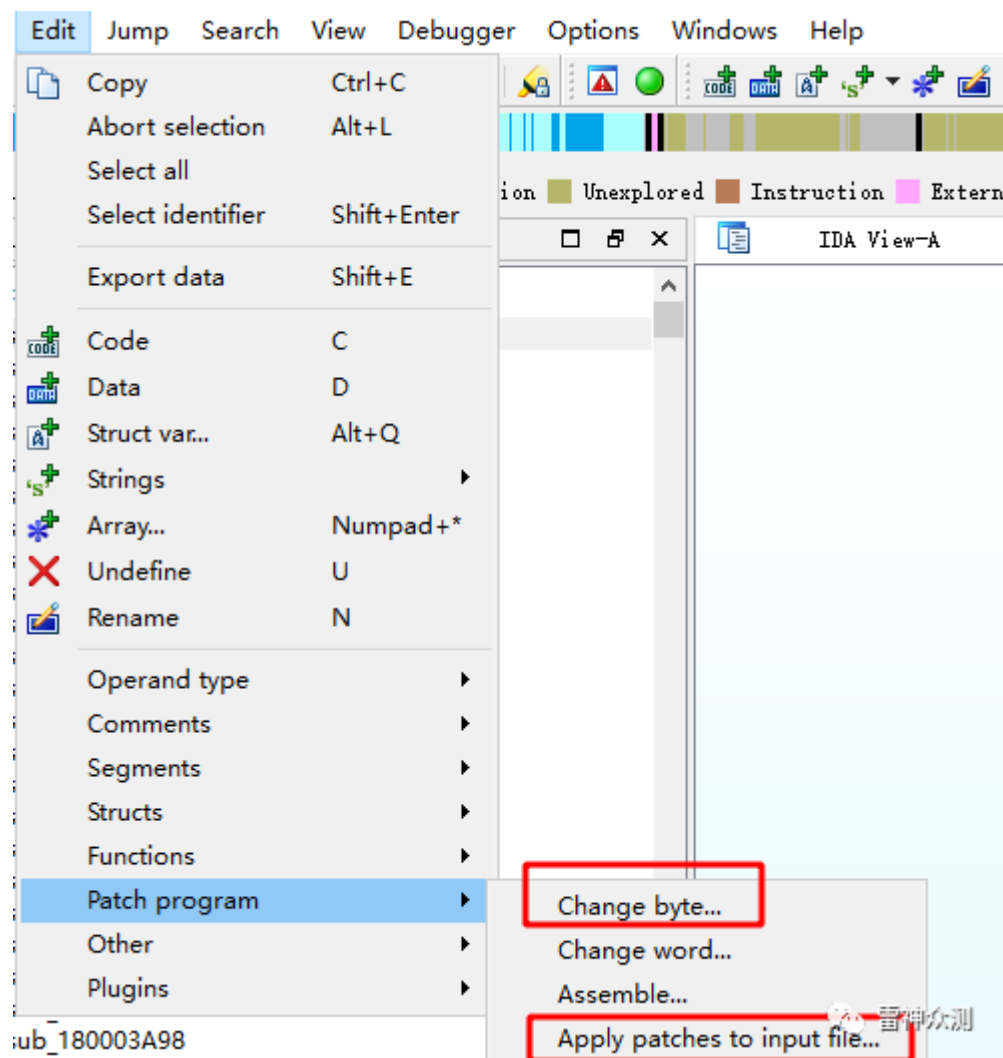
2、编译 (javac -encoding UTF-8 -classpath cobaltstrike.jar CrackSleeve.java)

3、解密文件 (java -classpath cobaltstrike.jar;./ CrackSleeve decode) # windows 命令行执行

Alt+T 进行关键字搜索: 2Eh



先双击再空格。直接修改 xor 的值，先 Change byte 找到 2E 修改，再 Apply pathes to input file 保存。（别忘记保存）



需要修改的 dll: beacon.dll、beacon.x64.dll、dnsb.dll、dnsb.x64.dll、pivot.dll、pivot.x64.dll、extc2.dll、extc2.x64.dll

再加密 dll, 工具给的命令在我这输入 CustomizeString 报错了, 于是给 CustomizeKey 写到代码里了, 改完如下可用:

```
import common.*;
import dns.SleeveSecurity;
import java.io.*;
import java.util.Enumeration;

import java.util.jar.JarEntry;
import java.util.jar.JarFile;

public class CrackSleeve {
    private static byte[] OriginKey = {58, 68, 37, 73, 15, 56, -102, -18, -61, 18, -67, -41, 88, -83, 43, -103};
    private static byte[] CustomizeKey;
    //private static byte[] CustomizeKey = {58, 68, 37, 73, 15, 56, -102, -18, -61, 18, -67, -41, 88, -83, 43, -103};

    private String DecDir = "Resource/Decode/sleeve";
    private String EncDir = "Resource/Encode/sleeve";

    public static void main(String[] args) throws IOException {
        if (args.length == 0 || args[0].equals("-h") || args[0].equals("--help")) {
            System.out.println("Usage: CrackSleeve OPTION [key]");
            System.out.println("Options:");
            System.out.println("\tdecode\t\tDecode sleeve files");
            System.out.println("\tencode\t\tEncode sleeve files");
            System.out.println("\tkey\t\tCustomize key string for encode sleeve files");
            System.exit(0);
        }
        String option = args[0];
        if (option.toLowerCase().equals("encode"))
        {
            CustomizeKey = new byte[]{58, 68, 37, 73, 15, 56, -102, -18, -61, 18, -67, -41, 88, -83, 43, -103};
        }

        CrackSleeve Cracker = new CrackSleeve();
    }
}
```

```

// 使用正版key初始化SleeveSecurity中的key
if (option.equals("decode")){
    CrackSleevedResource.Setup(OriginKey);
    Cracker.DecodeFile();
}else if (option.equals("encode")){

    CrackSleevedResource.Setup(CustomizeKey);
    Cracker.EncodeFile();
}
}

private void DecodeFile() throws IOException {
    // 文件保存目录
    File saveDir = new File(this.DecDir);
    if (!saveDir.isDirectory())
        saveDir.mkdirs();

    // 获取jar文件中sleeve文件夹下的文件列表
    try {
        String path = this.getClass().getClassLoader().getResource("sleeve").getPath();
        String jarPath = path.substring(5,path.indexOf("!/"));
        Enumeration<JarEntry> jarEnum = new JarFile(new File(jarPath)).entries();
        while (jarEnum.hasMoreElements())
        {
            JarEntry Element = jarEnum.nextElement();
            String FileName = Element.getName();
            if (FileName.indexOf("sleeve")>=0 && !FileName.equals("sleeve/")) {
                System.out.print("[+] Decoding "+FileName+".....");
                byte[] decBytes = CrackSleevedResource.DecodeResource(FileName);
                if (decBytes.length > 0) {
                    System.out.println("Done.");
                    CommonUtils.writeToFile(new File(saveDir,"../"+FileName),decBytes);
                }
            }
            else
                System.out.println("Fail.");
        }
    }
}

```

```

    }
} catch (IOException e) {
    e.printStackTrace();
}

}

private void EncodeFile(){
    // 文件保存目录
    File saveDir = new File(this.EncDir);
    if (!saveDir.isDirectory())
        saveDir.mkdirs();

    // 获取解密文件列表
    File decDir = new File(this.DecDir);
    File[] decFiles = decDir.listFiles();
    if (decFiles.length == 0) {
        System.out.println("[-] There's no file to encode, please decode first.");
        System.exit(0);
    }

    for (File file : decFiles){
        String filename = decDir.getPath()+"/"+file.getName();
        System.out.print("[+] Encoding " + file.getName() + ".....");
        byte[] encBytes = CrackSleevedResource.EncodeResource(filename);
        if (encBytes.length > 0) {
            System.out.println("Done.");
            CommonUtils.writeFile(new File(saveDir,file.getName()),encBytes);
        }
        else
            System.out.println("Fail.");
    }
}
}

class CrackSleevedResource{

```

```
private static CrackSleevedResource singleton;

private SleeveSecurity data = new SleeveSecurity();

public static void Setup(byte[] paramArrayOfbyte) {
    // singleton = new CrackSleevedResource(paramArrayOfbyte);
    singleton = new CrackSleevedResource(paramArrayOfbyte);
}

public static byte[] DecodeResource(String paramString) {
    return singleton._DecodeResource(paramString);
}

public static byte[] EncodeResource(String paramString) {
    return singleton._EncodeResource(paramString);
}

private CrackSleevedResource(byte[] paramArrayOfbyte) {
    this.data.registerKey(paramArrayOfbyte);
}

private byte[] _DecodeResource(String paramString) {
    byte[] arrayOfByte1 = CommonUtils.readResource(paramString);
    if (arrayOfByte1.length > 0) {
        long l = System.currentTimeMillis();
        return this.data.decrypt(arrayOfByte1);
    }
    byte[] arrayOfByte2 = CommonUtils.readResource(paramString);
    if (arrayOfByte2.length == 0) {
        CommonUtils.print_error("Could not find sleeved resource: " + paramString + " [ERROR]");
    } else {
        CommonUtils.print_stat("Used internal resource: " + paramString);
    }
    return arrayOfByte2;
}
```



```

private byte[] _EncodeResource(String paramString){
    try {
        File fileResource = new File(paramString);
        InputStream fileStream = new FileInputStream(fileResource);

        if (fileStream != null)
        {
            byte[] fileBytes = CommonUtils.readAll(fileStream);
            if (fileBytes.length > 0)
            {
                byte[] fileEncBytes = this.data.encrypt(fileBytes);
                return fileEncBytes;
            }
        }
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        return null;
    }
}

```

命令重新加密 dll：（解密加密其他版本 cs 修改为对应的 OriginKey、CustomizeKey 即可）

```
java -classpath cobaltstrike.jar;./ CrackSleeve encode
```

key 的值从 16 进制到 byte 型转换可用代码：

```

import java.util.Arrays;

public class authTest {
    public byte[] intToByteArray(int num){
        return new byte[] {
            (byte) ((num >> 24) & 0xFF),
            (byte) ((num >> 16) & 0xFF),
            (byte) ((num >> 8) & 0xFF),
            (byte) (num & 0xFF)
        };
    }
}

```

```

        (byte) ((num >> 8) & 0xFF),
        (byte) (num & 0xFF)
    };
}

public static byte[] hex2bytes(String s) {

    int len = s.length();
    byte[] data = new byte[len / 2];
    for (int i = 0; i < len; i += 2) {
        data[i / 2] = (byte) ((Character.digit(s.charAt(i), 16) << 4) + Character.digit(s.charAt(i+1), 16));
    }
    return data;}

public static void main(String[] args){
    authTest authTest = new authTest();
    int header = -889274157;
    int num = 29999999;
    int watermark = 1;

    byte[] bheader = authTest.intToByteArray(header);
    byte[] bnum = authTest.intToByteArray(num);
    byte[] bwatermark = authTest.intToByteArray(watermark);
    //      System.out.print(Arrays.toString(bheader)+'\n');
    //      System.out.print(Arrays.toString(bnum)+'\n');
    //      System.out.print(Arrays.toString(bwatermark)+'\n');
    System.out.println(Arrays.toString(hex2bytes("3a4425490f389aee312bdd758ad2b99"))));

}
}

```

最后，把 encode 目录下的 dll，放到 idea 项目目录中重新编译打包。

进行测试 uri 地址虽说仍旧可以请求到，但内容已经无法用 nmap 脚本解密出来，同理也可躲避扫描：

```

Response: 404 Not Found
null

11/07 23:33:11 visit (port 80) from: 10.0.1.1
Request: GET /Si8h/
beacon beacon stager x86
Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)

11/07 23:33:13 visit (port 80) from: 10.0.1.1
Request: GET /hp3R/
beacon beacon stager x64
Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)

11/07 23:34:08 visit (port 80) from: 10.0.1.1
Request: [i]eU[]$random1random2random3random4 /
Response: 404 Not Found
null

11/07 23:34:00 visit (port 80) from: 10.0.1.1
Request: GET /5xfI/
beacon beacon stager x86
Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)

11/07 23:34:10 visit (port 80) from: 10.0.1.1
Request: GET /VQDr/
beacon beacon stager x64
Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)

```

```

-> nmap -n -v --open -Pn -p 80 10.0.1.1 --script=grab_beacon_config.nse
Starting Nmap 7.80 ( https://nmap.org ) at 2021-11-07 23:34 CST
NSE: Loaded 1 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 23:34
Completed NSE at 23:34, 0.00s elapsed
Initiating SYN Stealth Scan at 23:34
Scanning 10.0.1.1 [1 port]
Discovered open port 80/tcp on 10.0.1.1
Completed SYN Stealth Scan at 23:34, 0.05s elapsed (1 total ports)
NSE: Script scanning 10.0.1.1.
Initiating NSE at 23:34
Completed NSE at 23:34, 5.07s elapsed
Nmap scan report for 10.0.1.1
Host is up (0.000049s latency).

PORT      STATE SERVICE
80/tcp    open  http

NSE: Script Post-scanning.
Initiating NSE at 23:34
Completed NSE at 23:34, 0.00s elapsed
Read data files from: /usr/bin/./share/nmap
Nmap done: 1 IP address (1 host up) scanned in 5.47 seconds
Raw packets sent: 1 (44B) | Rcvd: 2 (88B)

```

除了修改异或值的方式，也可以用 Beacon Stager listener 去特征 的方式改掉 checksum8 算法，但是只能固定 url 访问了，需要配合 profile 才能使用。

END