

原创

[3\\_n1ac](#) 于 2019-09-09 23:32:08 发布 84 收藏

分类专栏: [ft](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: [https://blog.csdn.net/weixin\\_44864225/article/details/100675789](https://blog.csdn.net/weixin_44864225/article/details/100675789)

版权



[ft](#) 专栏收录该内容

1 篇文章 0 订阅

订阅专栏

## 题目: Anderson Application Auditing

描述: You are in the role of a secret hacker. As always.

Your next job is the following:

VSA (Very Secret Agency) has followed very strict security policies for years, it is almost impossible to break into their network.

Unfortunately, that's what your boss wants from you.

After some social engineering you gathered, that VSA wants to order some simple programs from SoftMicro software development corporation.

SoftMicro is the old partner for VSA, and he has implemented lots of backdoors for a commercial operating system named "Doors" for VSA.

SoftMicro's software is usually crappy, but their network is very well defended - thanks to the very often attacks against SoftMicro's network.

But VSA doesn't accept any code from SoftMicro directly, because they hired a well known company named Anderson to audit every piece of code that are used at VSA.

Your plan is to hijack the communication between Anderson and SoftMicro, so you can analyse the program, and after Anderson audited the program, you will hijack the traffic between Anderson and VSA, exchange the program with your evil one, and the job is done.

The plan is great, but maybe not everything goes as planned...

Your first task is to hijack the communication between Anderson's and SoftMicro's network.

Here is the information you have already gathered:

The SoftMicro's network is 207.46.197.0

Your public IP is 17.149.160.49

Anderson's main page is Anderson

As you make progress on the challenge, you will get six pieces of a secret code, which is the proof that you have solved the challenge.

So, don't forget to write down those secret code pieces.

解析你的角色是一个黑客, 目的是入侵 **vsa**, 而 **vsa** 想要从 **softMrcro** 外包程序, **ms** 对一个叫做 **door** 的程序留下了许多后门。但是 **vsa** 并不是直接接受 **ms** 的代码, 他们雇佣了一个 **anderson** 的审计机构来审计所有的代码。计划劫持 **ms** 和 **anderson** 之间的通信, 这样可以分析程序, 当 **Anderson** 审计结束之后, 你将劫持 **Anderson** 和 **vsa** 之间的通信, 留下恶意代码, 我们劫持双方的通信, 做一个 **man-in-middle**, 是当 **anderson** 审查结束之后, 将正确的代码发过去的时候, 先经过我们, 然后我们留下好的代码, 发送恶意代码给 **vsa**

题目的第一部分目的是劫持双方的通信, 给了两个 **ip** 地址和 **anderson** 的主页

注意题目总共有六部分, 每解决一个部分会给出一个 **keyword**, 最后的 **keyword** 合起来就是本题的答案。

我们首先在 **anderson** 主页寻找隐藏信息, 各个链接都点一下, 发现 **parnter** 链接里又一个文档, 打开如下

AJP/1.3 8009  
http-alt 8080

With the new built-in Core 32768 CPU-s, the webserver can service twice the Earth population, whe they make connection simultaneously.

#### Firewall:

Version: 6.6  
Firewall Throughput: 2 GBps  
10/100 Interfaces: 8  
10/100/1000 Interfaces: 1

#### Router:

Version: 23.6  
CPU Speed: 2 GHz  
RAM: 2 GB  
Flash Memory: 512 MB  
Notes: 20 front panel LEDs (including link/activity, collision detection and speed rating indicat

There is a default username/password for all modules (which is a very strong one), but we strongly recommend to choose a new strong password for every module.

Default username:  
admin

Default password:  
jtqkcrInZpT

The web server config page is here:  
[www.example.com/webserver\\_config.html](http://www.example.com/webserver_config.html)

The firewall config page is here:  
[www.example.com/firewall\\_config.html](http://www.example.com/firewall_config.html)

The router config page is here:  
[www.example.com/router\\_config.html](http://www.example.com/router_config.html)

[https://blog.csdn.net/weixin\\_44864225](https://blog.csdn.net/weixin_44864225)

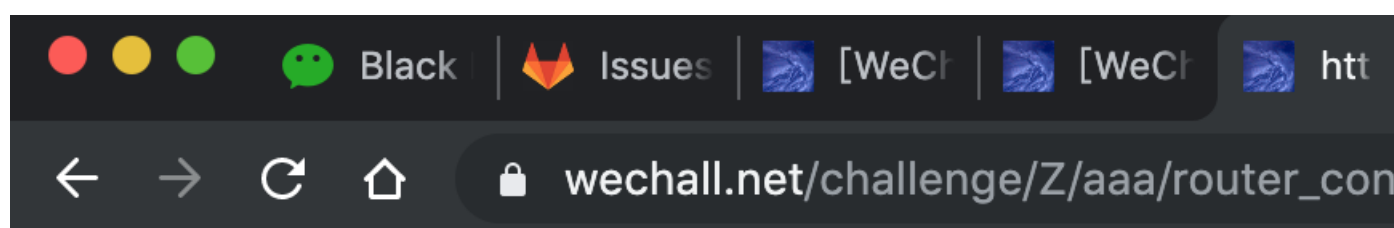
文档里不仅给出了用户名和密码，还给出了三个网站，分别是

The web server config page is here:  
[www.example.com/webserver\\_config.html](http://www.example.com/webserver_config.html)

The firewall config page is here:  
[www.example.com/firewall\\_config.html](http://www.example.com/firewall_config.html)

The router config page is here:  
[www.example.com/router\\_config.html](http://www.example.com/router_config.html)

三个网站都试一下，其实分析一下就可以知道是router\_config.html，因为需要窃听通信，所以需要添加路由规则。题目的网址是<https://www.wechall.net/challenge/Z/aaa/index.php>，所以路由网址是[https://www.wechall.net/challenge/Z/aaa/router\\_config.html](https://www.wechall.net/challenge/Z/aaa/router_config.html)



# Router configuration

# Router Configuration

Username:

Password:

Login

[https://blog.csdn.net/weixin\\_44864225](https://blog.csdn.net/weixin_44864225)

输入用户名密码，这个用户名和密码是在文档里边被泄漏的。输入之后添加路由。

wechall.net/challenge/Z/aaa/router.php

Login successful.

You can configure your router here, the syntax is the same as on all \*NIX boxes.

Example: `route add -net x.x.x.x netmask 255.255.255.0 gw x.x.x.x`

Config command:

Login

`route add -net 207.46.197.0 netmask 255.255.255.0 gw 17.149.160.49`，进入下一部分

> New route added successfully.  
1st part of the secret string: route

Actually, you found some very bad news on the net which may thwart your plans.

Anderson implemented PKI technology to digitally sign their documents, programs, etc. It is very likely, that they will digitally sign the program you want to change when it is transferred to VSA. Only one thing you have left as your final rescue. Anderson signs the MD5 hashes of the programs/documents, so maybe you can create two different programs for Anderson – which have identical MD5 hashes. The first good one will do the things as SoftMicro implemented, but the second evil one will do what you want. Anderson analyses the good one, you change it to the bad one, same hashes, same signature, game over.

Your journey continues here:

[Generating hashes](#)

第二部分分析题干中提示andersion通过pki加密，对文件进行数字签名，通过md5来验证文件，保证文件的完整性和信任性。所以如果我们需要用恶意程序替换正确程序，就需要将恶意程序的md5边成跟正确程序一样。点击链接，进入下一部分

Your next task is to create two different binaries, whose md5sum is the same. The first binary must be sent to Anderson for software analysis, and the second has to be sent to VSA in the next part of your mission.

The first good program has to print "Hello VSA employee" and your second, evil one has to print "I am a super VIRUS, game over."

This script verifies if the two different binaries are doing as specified, and if the md5sum is equal for them.

As SoftMicro's developers are in late to finish their job, maybe you can find the collision before they want it to send to Anderson.

Note: The script only checks if the good file contains the good string, the evil contains the evil string, and md5sum differs from the sha1 sum. On my average PC it took 8 hours to find a collision, but there is a quicker and smarter way.

Good  未选择任何文件

Evil  未选择任何文件

Hidden hint: search for md5 collision.

https://blog.csdn.net/weixin\_44864225

这个全选之后会有提示，这个提示非常有用，一开始做的时候没有发现走了很多弯路但不发现也能做出来。其实界面有变化本身就是一个提示，这个也是ctf的魅力之一。

谷歌搜索md5 collision，打开网址[网址](#)

按照指令，使用evilizi来解决。

#### quick usage instructions:

note for Windows users: the below instructions are for Unix/Linux. On Windows, you may have to append ".exe" to the names of executable files. Also, to use "make" you must have the GNU tools installed and working.

1. Unpack the archive and build the library and tools:

```
tar xzf evilize-0.2.tar.gz
cd evilize-0.2
make
```

This creates the programs "evilize", "md5coll", and the object file "goodevil.o".

2. Create a C program with multiple behaviors. Instead of the usual top-level function main(), write two separate top-level functions main\_good() and main\_evil(). See the file hello-erase.c for a simple example.
3. Compile your program and link against goodevil.o. For example:

```
gcc hello-erase.c goodevil.o -o hello-erase
```

4. Run the following command to create an initialization vector:

```
./evilize hello-erase -i
```

5. Create an MD5 collision by running the following command (but replace the vector on the command line with the one you found in step 4):

```
./md5coll 0x23d3e487 0x3e3ea619 0xc7bdd6fa 0x2d0271e7 > init.txt
```

Note: this step can take several hours.

6. Create a pair of good and evil programs by running:

```
./evilize hello-erase -c init.txt -g good -e evil
```

Here "good" and "evil" are the names of the two programs generated, and "hello-erase" is the name of the program you created in step 3.

NOTE: steps 4-6 can also be done in a single step, as follows:

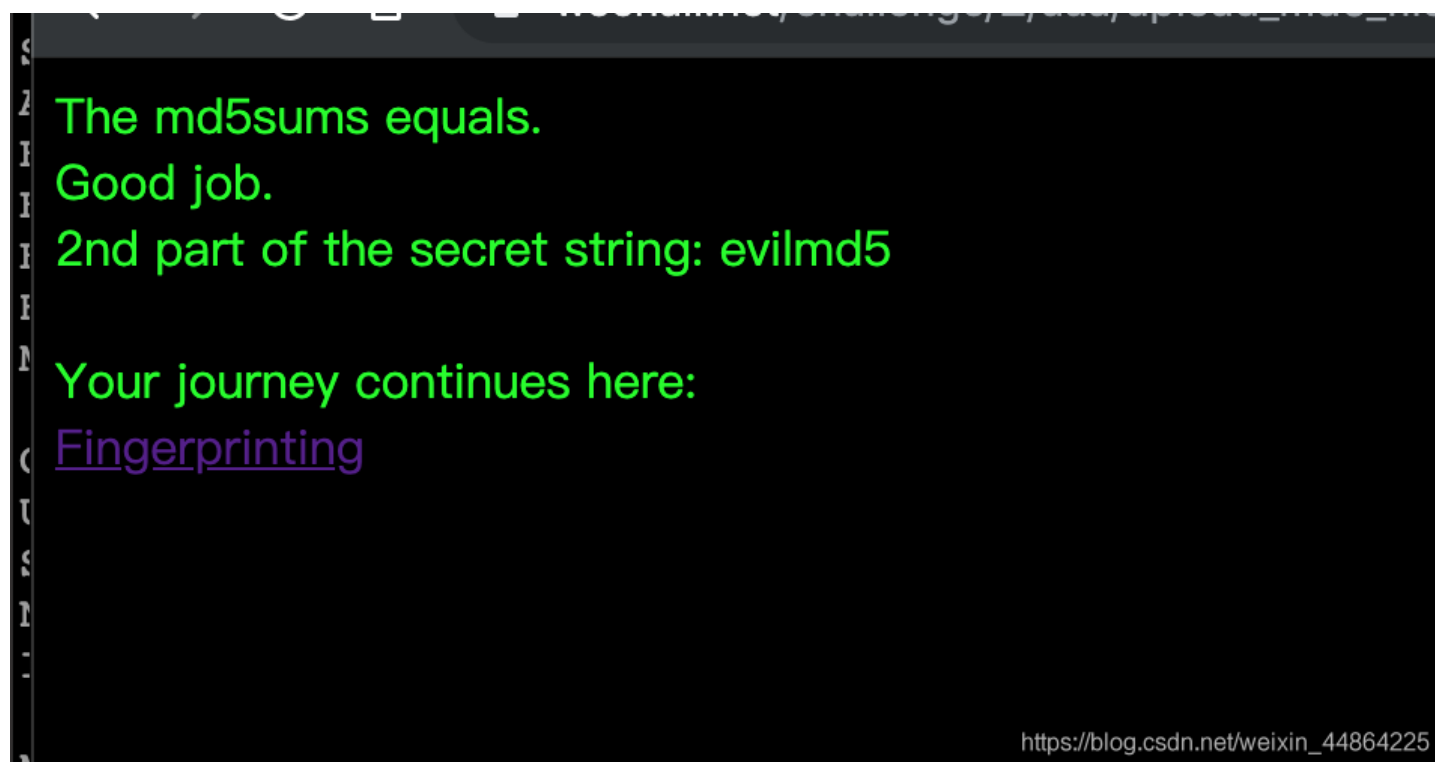
这个耗费的时间比较长，但是最终会生成两个程序 good和evil，正确程序和恶意程序按照如下写

```
#include <stdio.h>
#include <unistd.h>

int main_good(int ac,char *av[]){
    fprintf(stdout,"Hello VSA employee");
    return 0;
}

int main_evil(int ac,char *av[]){
    fprintf(stdout,"I am a super VIRUS,game over.");
    return 0;
}
```

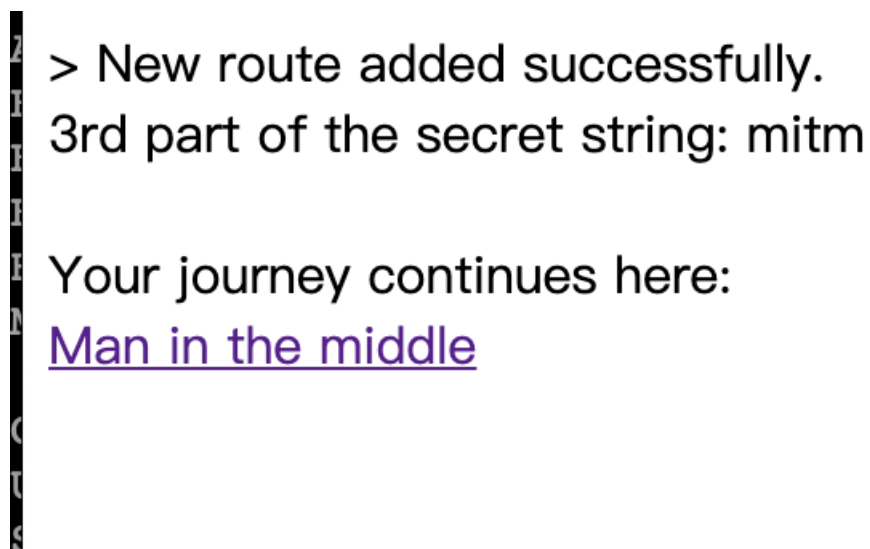
上传之后得到第二部分的密钥: evilmd5



得到提示: **Your next task is to hijack the traffic between Anderson and VSA.**

**VSA's IP network is 12.110.110.0 You know what to do...**

我们再次重复, 添加路由规则, 得到



第三部分密钥get，进入下一部分。

```
Great job, now you are a 'Man In The Middle' between Anderson and VSA. Waiting for some traffic you run nmap against VSA's servers, and find that only one SSH port is open to the public. This means you don't have a chance to change the traffic between VSA and Anderson (because it is encrypted and integrity checked), so you can't change your good program to the evil one. After knocking your head into the wall you read some posts on public forums from Anderson, and finally find something. Here is a small suggestion from Anderson when using ssh:  
"We strongly recommend our clients to check the fingerprint for first time communication, but we know how long these fingerprints are, so we recommend only to check the first 2 and the last bytes in hexa values in the fingerprint. Believe us, we really know what security is about..."  
Here you can download the public key for the VSA SSH server.  
VSA public key  
In this context fingerprint means a hexa digest for a public key. Your job is to create a 1024 bit ssh rsa2 private key, which corresponding fingerprint can trick the Anderson fingerprint matching protocol. This means that the first 2 and the last bytes in hexa values of the corresponding fingerprint must be the same as VSA's public key fingerprint  
For example if the current SSH key looks like this:  
02:34:54:78:AA:BB:CC:DD:EE:FF:11:12:13:14:15:16  
you have to create a private key which corresponding public key is  
02:34:54:78  
if you have this private key file, paste it in an OpenSSH format, without password protection, so your SSHarp tool can use it.  
Hint: playing with the parameters helped me to find 3 "good enough" private keys in 1 minute  
hidden hint: Search for fuzzy fingerprinting. You can find the tool with other search words, but thats the tool you really need.
```

题目解析题目的意思就是想要提交恶意程序，先要发送密钥到ssh端口。但是ssh端口的人工会进行md5审查，使用fuzzy fingerprinting可以一定程度上迷惑端口的审查人，同时提出一些要求，当然，最后一部分仍用黑色字体写了小彩蛋。

<https://bl4ckh47.wordpress.com/2009/10/25/fuzzy-fingerprinting/>

关于fuzzy fingerprinting的背景，原理以及安装和使用都在上边的链接，但是关于安装，其链接和资源都有问题，安装耽误了我大概一天的时间，详细步骤在我另一篇博客中给出。[fip安装过程链接](#)

上传私钥之后，提示要回去配置路由

```
Private key accepted.  
You are so close to finishing the challenge...  
The 4th secret piece is: fingerprint  
You fire up SSHarp with your newly generated private keys, and wait for the communication. After some days Anderson finishes the analyzing of the good code, digitally signs it, and sends the files to VSA. They open a new ssh/scp flow, but don't accept your private key. You are waiting, and waiting, when you hear a police sirene near to you. You almost destroy your computer when finally Anderson accepts your key file. They just needed some time to verify the digits by phone...  
After changing the good program to the evil one, you imagine what your virus will do at VSA. Your boss will be very proud of you...  
You have only one thing left to do: Cover your tracks at the router configuration.
```

主要就是删除路由规则，防止泄漏身份。  
将路由的规则配置时候的命令中的add改成del就行  
具体：

```
route del -net 207.46.197.0 netmask 255.255.255.0 gw 17.149.160.49  
route del -net 12.110.110.0 netmask 255.255.255.0 gw 17.149.160.49
```

得到最后两个密钥，将所有密钥连起来为：

```
routevilmd5mitmfingerprintgameover
```

就是题目最后的答案。



[创作打卡挑战赛](#) >

[赢取流量/现金/CSDN周边激励大奖](#)