**[Top](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/SECURITY/DAY04/CASE/01/index.html" \l "page_top_case)**

# NSD SECURITY DAY04

1. **[案例1：加密与解密应用](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/SECURITY/DAY04/CASE/01/index.html" \l "case1)**

**加密目的方式**

**确保数据的机密性**

**对称加密:加密／解密用同一个密钥(相当口令)**

**非对称加密:加密／解密用不同的密钥(公钥加密，私钥解密)**

**保护信息的完整性**

**信息摘要:基于输入的信息生成长度较短，位数固定的散列值**

**常见的加密算法**

**对称加密**

**DES，Data Encryption Standard**

**AES，Advanced Encryption Standard**

**非对称加密**

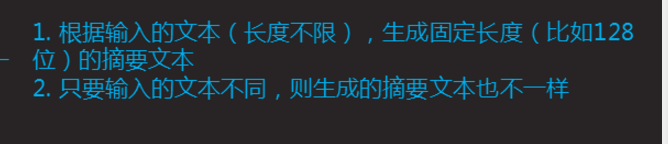
**RSA，Rivest Shamirh Adleman**

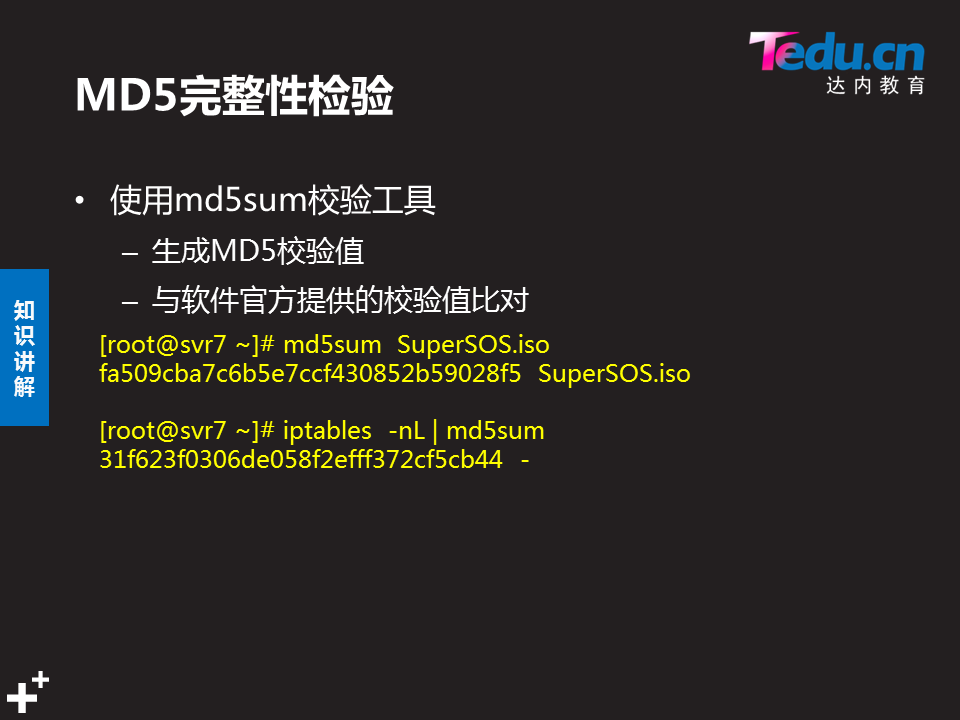
**DSA，Digital Signature Algorithm**

**Hash散列技术，用于信息摘要**

**MD5，Message Digest Algorithm5**

**SHA，Secure Hash Algorithm**

****



**gpg数据对称加解密**

**随机数**

**mv /dev/random /dev/randomold**

**ln -s /dev/urandom /dev/random**

1. **[案例2：使用AIDE做入侵检测](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/SECURITY/DAY04/CASE/01/index.html" \l "case2)**
2. **[案例3：扫描与抓包分析](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/SECURITY/DAY04/CASE/01/index.html" \l "case3)**

**为什么需要扫描**

**以获取一些公开／非公开信息为目的**

**检测潜在的风险**

**查找可攻击的目标**

**收集设备/主机/系统/软件信息**

**发现可利用的安全漏洞**

**扫描方式及工具**

**典型的扫描方式**

**Scan,主动探测**

**Sniff,被动监听/嗅探**

**Capture,数据包捕获(抓包)**

**常见的安全分析工具**

**扫描器:NMAP**

**协议分析:tcpdump,Wireshark**

**网络抓包工具**

**Tcpdump抓包命令**

**命令行抓取数据包的工具**

**用法**

**tcpdump [选项]　[过滤条件]**

**常见的监控选项**

**-i //指定监控的网络接口**

**-A //转换为ACSII码，以方便阅读**

**-w //将数据包信息保存到指定文件**

**-r //从指定文件读取数据包信息**

**-c //定义抓包个数**

**Tcpdump抓包命令(1)**

**Tcpdump的过滤条件**

**类型:host net port portrange**

**方向:src dst**

**协议:tcp udp ip wlan arp icmp ....**

**多个条件组合:and or not**

## **1 案例1：加密与解密应用**

### **1.1 问题**

本案例要求采用gpg工具实现加/解密及软件签名等功能，分别完成以下任务：

1. 检查文件的MD5校验和
2. 使用GPG实现文件机密性保护，加密和解密操作
3. 使用GPG的签名机制，验证数据的来源正确性

### **1.2 方案**

加密算法主要有以下几种分类：

1.为确保数据机密性算法：

a) 对称加密算法(AES,DES)

b) 非对称加密算法（RSA，DSA）

2.为确保数据完整性算法：

a) 信息摘要（MD5，SHA256，SHA512）

### **1.3 步骤**

实现此案例需要按照如下步骤进行。

**步骤一：检查文件的MD5校验和**

1） 查看文件改动前的校验和，复制为新文件其校验和不变

1. **[**root@proxy **~]**# vim file1**.**txt
2. abcdef
3. **123456779**
4. **[**root@proxy **~]**# cp file1**.**txt file2**.**txt
5. **[**root@proxy **~]**# cat file1**.**txt **>** file3**.**txt
6. **[**root@proxy **~]**# md5sum file**?.**txt                 //文件内容一致，则校验和也不变
7. b92aa0f8aa5d5af5a47c6896283f3536 file1**.**txt
8. b92aa0f8aa5d5af5a47c6896283f3536 file2**.**txt
9. b92aa0f8aa5d5af5a47c6896283f3536 file3**.**txt

2） 对文件内容稍作改动，再次检查校验和，会发现校验和已大不相同

1. **[**root@proxy **~]**# echo "x" **>>** file1**.**txt
2. **[**root@proxy **~]**# md5sum file**?.**txt
3. 6be3efe71d8b4b1ed34ac45f4edd2ba7 file1**.**txt
4. b92aa0f8aa5d5af5a47c6896283f3536 file2**.**txt
5. b92aa0f8aa5d5af5a47c6896283f3536 file3**.**txt

**步骤二：使用GPG对称加密方式保护文件**

GnuPG是非常流行的加密软件，支持所有常见加密算法，并且开源免费使用。

1）确保已经安装了相关软件（默认已经安装好了）

1. **[**root@proxy **~]**# yum **-**y install gnupg2            //安装软件
2. **[**root@proxy **~]**# gpg **--**version                    //查看版本
3. gpg **(**GnuPG**)** **2.0.22**

2） gpg使用对称加密算法加密数据的操作

执行下列操作：

1. **[**root@proxy **~]**# gpg **-**c file2**.**txt
2. **..** **..**

根据提示依次输入两次密码即可。如果是在GNOME桌面环境，设置密码的交互界面会是弹出的窗口程序，如图-1所示：



图－1

如果是在tty终端执行的上述加密操作，则提示界面也是文本方式的，如图-2所示。



图-2

根据提示输入两次口令，加密后的文件（自动添加后缀 .gpg）就生成了，传递过程中只要发送加密的文件（比如 file2.txt.gpg）就可以了。

1. **[**root@proxy **~]**# cat file2**.**txt**.**gpg                    //查看加密数据为乱码

3）使用gpg对加密文件进行解密操作

收到加密的文件后，必须进行解密才能查看其内容。

1. **[**root@proxy **~]**# gpg **-**d file2**.**txt**.**gpg **>** file2**.**txt     //解密后保存
2. gpg**:** 3DES 加密过的数据
3. **..** **..**                                             //根据提示输入正确密码
4. **[**root@proxy **~]**# cat file2**.**txt                     //查看解密后的文件
5. abcdef
6. **123456779**

**步骤三：使用GPG非对称加密方式保护文件**

非对称加密/解密文件时，UserA（192.168.4.100）生成私钥与公钥，并把公钥发送给UserB（192.168.4.5），UserB使用公钥加密数据，并把加密后的数据传给UserA，UserA最后使用自己的私钥解密数据。

实现过程如下所述。

1）接收方UserA创建自己的公钥、私钥对(在192.168.4.100操作)

1. **[**root@client **~]**# gpg **--**gen**-**key //创建密钥对
2. … …
3. 请选择您要使用的密钥种类：
4. **(1)** RSA and RSA **(default)**                            //默认算法为RSA
5. **(2)** DSA and Elgamal
6. **(3)** DSA **(**仅用于签名**)**
7. **(4)** RSA **(**仅用于签名**)**
8. 您的选择？                                             //直接回车默认(1)
9. RSA 密钥长度应在 **1024** 位与 **4096** 位之间。
10. 您想要用多大的密钥尺寸？**(2048)**                             //接受默认2048位
11. 您所要求的密钥尺寸是 **2048** 位
12. 请设定这把密钥的有效期限。
13. **0** **=** 密钥永不过期
14. **<**n**>** **=** 密钥在 n 天后过期
15. **<**n**>**w **=** 密钥在 n 周后过期
16. **<**n**>**m **=** 密钥在 n 月后过期
17. **<**n**>**y **=** 密钥在 n 年后过期
18. 密钥的有效期限是？**(0)**                                         //接受默认永不过期
19. 密钥永远不会过期
20. 以上正确吗？**(**y**/**n**)**y                                         //输入y确认
21. You need a user ID to identify your key**;** the software constructs the user ID
22. from the Real Name**,** Comment and Email Address **in** **this** form**:**
23. "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"
24. 真实姓名：UserA
25. 电子邮件地址：UserA@tarena**.**com
26. 注释：UserA
27. 您选定了这个用户标识：
28. “UserA **(**UserA**)** **<**UserA@tarena**.**com**>**”
29. 更改姓名**(**N**)**、注释**(**C**)**、电子邮件地址**(**E**)**或确定**(**O**)/**退出**(**Q**)**？O         //输入大写O确认
30. 您需要一个密码来保护您的私钥。
31. 我们需要生成大量的随机字节。这个时候您可以多做些琐事**(**像是敲打键盘、移动
32. 鼠标、读写硬盘之类的**)**，这会让随机数字发生器有更好的机会获得足够的熵数。
33. gpg**:** 正在检查信任度数据库
34. gpg**:** 需要 **3** 份勉强信任和 **1** 份完全信任，PGP 信任模型
35. gpg**:** 深度：**0** 有效性： **1** 已签名： **0** 信任度：**0-**，0q，0n，0m，0f，**1u**
36. pub 2048R**/**421C9354 **2017-08-16**
37. 密钥指纹 **=** 8A27 6FB5 **1315** CEF8 D8A0 A65B F0C9 7DA6 421C **9354**
38. uid UserA **(**UserA**)** **<**UserA@tarena**.**com**>**
39. sub 2048R**/**9FA3AD25 **2017-08-16**

注意：生产密钥后当前终端可能会变的无法使用，执行reset命令即可，或者关闭后再开一个终端。

2）UserA导出自己的公钥文件(在192.168.4.100操作)

用户的公钥、私钥信息分别保存在pubring.gpg和secring.gpg文件内：

1. **[**root@client **~]**# gpg **--**list**-**keys                         //查看公钥环
2. /root/**.**gnupg**/**pubring**.**gpg
3. **------------------------------**
4. pub 2048R**/**421C9354 **2017-08-16**
5. uid UserA **(**User A**)** **<**UserA@tarena**.**com**>**
6. sub 2048R**/**9FA3AD25 **2017-08-16**

使用gpg命令结合--export选项将其中的公钥文本导出：

1. **[**root@client **~]**# gpg **-**a **--export** UserA **>** UserA**.**pub
2. //--export的作用是导出密钥，-a的作用是导出的密钥存储为ASCII格式
3. **[**root@client **~]**# scp UserA**.**pub **192.168.4.5:**/tmp/
4. //将密钥传给Proxy

3）UserB导入接收的公钥信息（在192.168.4.5操作）

使用gpg命令结合--import选项导入发送方的公钥信息，以便在加密文件时指定对应的公钥。

1. **[**root@proxy **~]**# gpg **--**import **/**tmp**/**UserA**.**pub
2. gpg**:** 密钥 421C9354：公钥“UserA **(**UserA**)** **<**UserA@tarena**.**com**>**”已导入
3. gpg**:** 合计被处理的数量：**1**
4. gpg**:** 已导入：**1** **(**RSA**:** **1)**

4) UserB使用公钥加密数据，并把加密后的数据传给UserA（在192.168.4.5操作）

1. **[**root@proxy **~]**# echo "I love you ." **>** love**.**txt
2. **[**root@proxy **~]**# gpg **-**e **-**r UserA love**.**txt
3. 无论如何还是使用这把密钥吗？**(**y**/**N**)**y                         //确认使用此密钥加密文件
4. //-e选项是使用密钥加密数据
5. //-r选项后面跟的是密钥，说明使用哪个密钥对文件加密
6. **[**root@proxy **~]**# scp love**.**txt**.**gpg **192.168.4.100:**/root    /**/**加密的数据传给UserA

4）UserA以自己的私钥解密文件（在192.168.4.100操作）

1. **[**root@client **~]**# gpg **-**d love**.**txt**.**gpg **>** love**.**txt
2. 您需要输入密码，才能解开这个用户的私钥：“UserA **(**UserA**)** **<**UserA@tarena**.**com**>**”
3. **2048** 位的 RSA 密钥，钥匙号 9FA3AD25，建立于 **2017-08-16** **(**主钥匙号 421C9354**)**
4. //验证私钥口令
5. gpg**:** 由 **2048** 位的 RSA 密钥加密，钥匙号为 9FA3AD25、生成于 **2017-08-16**
6. “UserA **(**UserA**)** **<**UserA@tarena**.**com**>**”
7. **[**root@client **~]**# cat love**.**txt                     //获得解密后的文件内容
8. I love you**.**

**步骤四：使用GPG的签名机制，检查数据来源的正确性**

使用私钥签名的文件，是可以使用对应的公钥验证签名的，只要验证成功，则说明这个文件一定是出自对应的私钥签名，除非私钥被盗，否则一定能证明这个文件来自于某个人！

1）在client(192.168.4.100)上，UserA为软件包创建分离式签名

将软件包、签名文件、公钥文件一起发布给其他用户下载。

1. **[**root@client **~]**# tar zcf log**.**tar **/var**/log             /**/**建立测试软件包
2. **[**root@client **~]**# gpg **-**b log**.**tar                     //创建分离式数字签名
3. **[**root@client **~]**# ls **-**lh log**.**tar**\***
4. **-**rw**-**rw**-**r**--.** **1** root root **170** **8**月 **17** **21:18** log**.**tar
5. **-**rw**-**rw**-**r**--.** **1** root root **287** **8**月 **17** **21:22** log**.**tar**.**sig
6. **[**root@client **~]**# scp log**.**tar**\*** **192.168.4.5:**/root        /**/**将签名文件与签名传给UserB

2）在192.168.4.5上验证签名

1. **[**root@proxy **~]**# gpg **--**verify log**.**tar**.**sig log**.**tar
2. gpg**:**于**2028**年**06**月**07**日 星期六 **23**时**23**分**23**秒 CST 创建的签名，使用 RSA，钥匙号 421C9354
3. gpg**:** 完好的签名，来自于“UserA **(**UserA**)** **<**UserA@tarena**.**com**>**”
4. **..** **..**

## **2 案例2：使用AIDE做入侵检测**

### **2.1 问题**

本案例要求熟悉Linux主机环境下的常用安全工具，完成以下任务操作：

1. 安装aide软件
2. 执行初始化校验操作，生成校验数据库文件
3. 备份数据库文件到安全的地方
4. 使用数据库执行入侵检测操作

### **2.2 方案**

Aide通过检查数据文件的权限、时间、大小、哈希值等，校验数据的完整性。

使用Aide需要在数据没有被破坏前，对数据完成初始化校验，生成校验数据库文件，在被攻击后，可以使用数据库文件，快速定位被人篡改的文件。

### **2.3 步骤**

实现此案例需要按照如下步骤进行。

**步骤一：部署AIDE入侵检测系统**

1）安装软件包

1. **[**root@proxy **~]**# yum **-**y install aide

2) 修改配置文件

确定对哪些数据进行校验，如何校验数据

1. **[**root@proxy **~]**# vim **/**etc**/**aide**.**conf
2. @@define DBDIR **/var**/lib/aide                            //数据库目录
3. @@define LOGDIR **/var**/log/aide                            //日志目录
4. database\_out**=**file**:**@@**{**DBDIR**}**/aide.db.new.gz                /**/**数据库文件名
5. //一下内容为可以检查的项目（权限，用户，组，大小，哈希值等）
6. #p**:** permissions
7. #i**:** inode**:**
8. #n**:** number of links
9. #u**:** user
10. #g**:** group
11. #s**:** size
12. #md5**:** md5 checksum
13. #sha1**:** sha1 checksum
14. #sha256**:** sha256 checksum
15. DATAONLY **=** p**+**n**+**u**+**g**+**s**+**acl**+**selinux**+**xattrs**+**sha256
16. //以下内容设置需要对哪些数据进行入侵校验检查
17. //注意：为了校验的效率，这里将所有默认的校验目录与文件都注释
18. //仅保留/root目录，其他目录都注释掉
19. **/**root DATAONLY
20. #/boot NORMAL                                    /**/**对哪些目录进行什么校验
21. #**/**bin NORMAL
22. #**/**sbin NORMAL
23. #**/**lib NORMAL
24. #**/**lib64 NORMAL
25. #**/**opt NORMAL
26. #**/**usr NORMAL
27. #**!**/usr/src                                        //使用[!]，设置不校验的目录
28. #**!**/usr/tmp

**步骤二：初始化数据库，入侵后检测**

1）入侵前对数据进行校验，生成初始化数据库

1. **[**root@proxy **~]**# aide **--**init
2. AIDE**,** version **0.15.1**
3. AIDE database at **/var**/lib/aide**/**aide**.**db**.new.**gz initialized**.**
4. //生成校验数据库，数据保存在/var/lib/aide/aide.db.new.gz

2）备份数据库，将数据库文件拷贝到U盘（非必须的操作）

1. **[**root@proxy **~]**# cp **/var**/lib/aide**/**aide**.**db**.new.**gz **/**media**/**

3）入侵后检测

1. **[**root@proxy **~]**# cd **/var**/lib/aide**/**
2. **[**root@proxy **~]**# mv aide**.**db**.new.**gz aide**.**db**.**gz
3. **[**root@proxy **~]**# aide **--**check                            //检查哪些数据发生了变化

## **3 案例3：扫描与抓包分析**

### **3.1 问题**

本案例要求熟悉Linux主机环境下的常用安全工具，完成以下任务操作：

1. 使用NMAP扫描来获取指定主机/网段的相关信息
2. 使用tcpdump分析FTP访问中的明文交换信息

### **3.2 步骤**

实现此案例需要按照如下步骤进行。

**步骤一：使用NMAP扫描来获取指定主机/网段的相关信息**

1）安装软件

1. **[**root@proxy **~]**# yum **-**y install nmap
2. //基本用法：
3. # nmap **[**扫描类型**]** **[**选项**]** **<**扫描目标 **...>**
4. //常用的扫描类型
5. // -sS，TCP SYN扫描（半开）
6. // -sT，TCP 连接扫描（全开）
7. // -sU，UDP扫描
8. // -sP，ICMP扫描
9. // -A，目标系统全面分析

2）检查192.168.4.100主机是否可以ping通

1. **[**root@proxy **~]**# nmap **-**sP **192.168.4.100**
2. Starting Nmap **6.40** **(** http**:**//nmap.org ) at 2018-06-06 21:59 CST
3. mass\_dns**:** warning**:** Unable to determine any DNS servers**.** Reverse DNS is disabled**.** Try using **--**system**-**dns or specify valid servers **with** **--**dns**-**servers
4. Nmap scan report **for** host3 **(192.168.4.100)**
5. Host is up **(0.**00036s latency**).**
6. MAC Address**:** **52:54:00:71:07:76** **(**QEMU Virtual NIC**)**
7. Nmap done**:** **1** IP address **(1** host up**)** scanned **in** **0.02** seconds

使用-n选项可以不执行DNS解析

1. **[**root@proxy **~]**# nmap **-**n **-**sP **192.168.4.100**
2. Starting Nmap **6.40** **(** http**:**//nmap.org ) at 2018-06-06 22:00 CST
3. Nmap scan report **for** **192.168.4.100**
4. Host is up **(0.**00046s latency**).**
5. MAC Address**:** **52:54:00:71:07:76** **(**QEMU Virtual NIC**)**
6. Nmap done**:** **1** IP address **(1** host up**)** scanned **in** **0.03** seconds

3）检查192.168.4.0/24网段内哪些主机可以ping通

1. **[**root@proxy **~]**# nmap **-**n **-**sP **192.168.4.0/24**
2. Starting Nmap **5.51** **(** http**:**//nmap.org ) at 2017-05-17 18:01 CST
3. Nmap scan report **for** **192.168.4.1**
4. Host is up**.**
5. Nmap scan report **for** **192.168.4.7**
6. Host is up**.**
7. Nmap scan report **for** **192.168.4.120**
8. Host is up **(0.**00027s latency**).**
9. MAC Address**:** **00:**0C**:29:74:**BE**:21** **(**VMware**)**
10. Nmap scan report **for** **192.168.4.110**
11. Host is up **(0.**00016s latency**).**
12. MAC Address**:** **00:50:56:**C0**:00:01** **(**VMware**)**
13. Nmap scan report **for** **192.168.4.120**
14. Host is up **(0.**00046s latency**).**
15. MAC Address**:** **00:**0C**:29:**DB**:84:46** **(**VMware**)**
16. Nmap done**:** **256** IP addresses **(5** hosts up**)** scanned **in** **3.57** seconds

4）检查目标主机所开启的TCP服务

1. **[**root@proxy **~]**# nmap **-**sT **192.168.4.100**
2. Starting Nmap **5.51** **(** http**:**//nmap.org ) at 2018-05-17 17:55 CST
3. Nmap scan report **for** **192.168.4.100**
4. Host is up **(0.**00028s latency**).**
5. Not shown**:** **990** closed ports
6. PORT STATE SERVICE
7. **21/**tcp open ftp
8. **22/**tcp open ssh
9. **25/**tcp open smtp
10. **80/**tcp open http
11. **110/**tcp open pop3
12. **111/**tcp open rpcbind
13. **143/**tcp open imap
14. **443/**tcp open https
15. **993/**tcp open imaps
16. **995/**tcp open pop3s
17. MAC Address**:** **00:**0C**:29:74:**BE**:21** **(**VMware**)**
18. Nmap done**:** **1** IP address **(1** host up**)** scanned **in** **1.31** seconds

5）检查192.168.4.0/24网段内哪些主机开启了FTP、SSH服务

1. **[**root@proxy **~]**# nmap **-**p **21-22** **192.168.4.0/24**
2. Starting Nmap **5.51** **(** http**:**//nmap.org ) at 2017-05-17 18:00 CST
3. Nmap scan report **for** **192.168.4.1**
4. Host is up **(0.**000025s latency**).**
5. PORT STATE SERVICE
6. **21/**tcp open ftp
7. **22/**tcp open ssh
8. Nmap scan report **for** **192.168.4.7**
9. Host is up**.**
10. PORT STATE SERVICE
11. **21/**tcp filtered ftp
12. **22/**tcp filtered ssh
13. Nmap scan report **for** **192.168.4.120**
14. Host is up **(0.**00052s latency**).**
15. PORT STATE SERVICE
16. **21/**tcp open ftp
17. **22/**tcp open ssh
18. MAC Address**:** **00:**0C**:29:74:**BE**:21** **(**VMware**)**
19. Nmap scan report **for** pc110**.**tarena**.**com **(192.168.4.110)**
20. Host is up **(0.**00038s latency**).**
21. PORT STATE SERVICE
22. **21/**tcp closed ftp
23. **22/**tcp closed ssh
24. MAC Address**:** **00:50:56:**C0**:00:01** **(**VMware**)**
25. Nmap scan report **for** **192.168.4.120**
26. Host is up **(0.**00051s latency**).**
27. PORT STATE SERVICE
28. **21/**tcp closed ftp
29. **22/**tcp closed ssh
30. MAC Address**:** **00:**0C**:29:**DB**:84:46** **(**VMware**)**
31. Nmap done**:** **256** IP addresses **(5** hosts up**)** scanned **in** **4.88** seconds

6）检查目标主机所开启的UDP服务

1. **[**root@proxy **~]**# nmap **-**sU **192.168.4.100**             //指定-sU扫描UDP
2. **53/**udp open domain
3. **111/**udp open rpcbind

7）全面分析目标主机192.168.4.100和192.168.4.5的操作系统信息

1. **[**root@proxy **~]**# nmap **-**A **192.168.4.100,5**
2. Starting Nmap **5.51** **(** http**:**//nmap.org ) at 2017-05-17 18:03 CST
3. Nmap scan report **for** **192.168.4.100**                     //主机mail的扫描报告
4. Host is up **(0.**0016s latency**).**
5. Not shown**:** **990** closed ports
6. PORT STATE SERVICE VERSION
7. **21/**tcp open ftp vsftpd **2.2.2**
8. **|** ftp**-**anon**:** Anonymous FTP login allowed **(**FTP code **230)**
9. **|** **-**rw**-**r**--**r**--** **1** **0** **0** **1719** Aug **17** **13:33** UserB**.**pub
10. **|** **-**rw**-**r**--**r**--** **1** **0** **0** **122** Aug **13** **05:27** dl**.**txt
11. **|** drwxr**-**xr**-**x **2** **14** **0** **4096** Aug **13** **09:07** pub
12. **|** **-**rw**-**rw**-**r**--** **1** **505** **505** **170** Aug **17** **13:18** tools**-1.2.3.**tar**.**gz
13. **|**\_**-**rw**-**rw**-**r**--** **1** **505** **505** **287** Aug **17** **13:22** tools**-1.2.3.**tar**.**gz**.**sig
14. **22/**tcp open ssh OpenSSH **5.3** **(**protocol **2.0)**
15. **|** ssh**-**hostkey**:** **1024** **86:**be**:**d6**:89:**c1**:**2d**:**d9**:**1f**:57:**2f**:66:**d1**:**af**:**a8**:**d3**:**c6 **(**DSA**)**
16. **|**\_2048 **16:**0a**:15:01:**fa**:**bb**:91:**1d**:**cc**:**ab**:68:17:58:**f9**:49:**4f **(**RSA**)**
17. **25/**tcp open smtp Postfix smtpd
18. **80/**tcp open http Apache httpd **2.2.15** **((**Red Hat**))**
19. **|**\_http**-**methods**:** No Allow or Public header **in** OPTIONS response **(**status code **302)**
20. **|** http**-**title**:** **302** Found
21. **|**\_Did not follow redirect to https**:**//192.168.4.100//
22. **110/**tcp open pop3 Dovecot pop3d
23. **|**\_pop3**-**capabilities**:** USER CAPA UIDL TOP OK**(**K**)** RESP**-**CODES PIPELINING STLS SASL**(**PLAIN**)**
24. **111/**tcp open rpcbind
25. MAC Address**:** **00:**0C**:29:74:**BE**:21** **(**VMware**)**
26. No exact OS matches **for** host **(**If you know what OS is running on it**,** see http**:**//nmap.org/submit/ ).
27. TCP**/**IP fingerprint**:**
28. OS**:**SCAN**(**V**=5.51%**D**=8/19%**OT**=21%**CT**=1%**CU**=34804%**PV**=**Y**%**DS**=1%**DC**=**D**%**G**=**Y**%**M**=**000C29**%**TM**=52**
29. OS**:**11ED90**%**P**=**x86\_64**-**redhat**-**linux**-**gnu**)**SEQ**(**SP**=106%**GCD**=1%**ISR**=**10B**%**TI**=**Z**%**CI**=**Z**%**II**=**I
30. OS**:%**TS**=**A**)**OPS**(**O1**=**M5B4ST11NW6**%**O2**=**M5B4ST11NW6**%**O3**=**M5B4NNT11NW6**%**O4**=**M5B4ST11NW6**%**O
31. OS**:5=**M5B4ST11NW6**%**O6**=**M5B4ST11**)**WIN**(**W1**=3890%**W2**=3890%**W3**=3890%**W4**=3890%**W5**=3890%**W6
32. OS**:=3890)**ECN**(**R**=**Y**%**DF**=**Y**%**T**=40%**W**=3908%**O**=**M5B4NNSNW6**%**CC**=**Y**%**Q**=)**T1**(**R**=**Y**%**DF**=**Y**%**T**=40%**S**=**O
33. OS**:%**A**=**S**+%**F**=**AS**%**RD**=0%**Q**=)**T2**(**R**=**N**)**T3**(**R**=**N**)**T4**(**R**=**Y**%**DF**=**Y**%**T**=40%**W**=0%**S**=**A**%**A**=**Z**%**F**=**R**%**O**=%**RD**=**
34. OS**:0%**Q**=)**T5**(**R**=**Y**%**DF**=**Y**%**T**=40%**W**=0%**S**=**Z**%**A**=**S**+%**F**=**AR**%**O**=%**RD**=0%**Q**=)**T6**(**R**=**Y**%**DF**=**Y**%**T**=40%**W**=0%**
35. OS**:**S**=**A**%**A**=**Z**%**F**=**R**%**O**=%**RD**=0%**Q**=)**T7**(**R**=**Y**%**DF**=**Y**%**T**=40%**W**=0%**S**=**Z**%**A**=**S**+%**F**=**AR**%**O**=%**RD**=0%**Q**=)**U1**(**
36. OS**:**R**=**Y**%**DF**=**N**%**T**=40%**IPL**=164%**UN**=0%**RIPL**=**G**%**RID**=**G**%**RIPCK**=**G**%**RUCK**=**G**%**RUD**=**G**)**IE**(**R**=**Y**%**DFI**=**
37. OS**:**N**%**T**=40%**CD**=**S**)**
38. Network Distance**:** **1** hop
39. Service Info**:** Host**:** mail**.**tarena**.**com**;** OS**:** Unix
40. TRACEROUTE
41. HOP RTT ADDRESS
42. **1** **1.55** ms **192.168.4.100**

**步骤二：使用tcpdump分析FTP访问中的明文交换信息**

1）准备Vsftpd服务器（192.168.4.5操作）

1. **[**root@proxy **~]**# yum **-**y install vsftpd
2. **[**root@proxy **~]**# systemctl restart vsftpd

2）启用tcpdump命令行抓包

执行tcpdump命令行，添加适当的过滤条件，只抓取访问主机192.168.4.5的21端口的数据通信 ，并转换为ASCII码格式的易读文本。

这里假设，192.168.4.5主机有vsftpd服务，如果没有需要提前安装并启动服务！！！

1. **[**root@proxy **~]**# tcpdump **-**A host **192.168.4.5** and tcp port **21**
2. tcpdump**:** verbose output suppressed**,** use **-**v or **-**vv **for** full protocol decode
3. listening on eth0**,** link**-**type EN10MB **(**Ethernet**),** capture size **65535** bytes
4. **..** **..**                                            //进入等待捕获数据包的状态
5. //监控选项如下：
6. // -i，指定监控的网络接口（默认监听第一个网卡）
7. // -A，转换为 ACSII 码，以方便阅读
8. // -w，将数据包信息保存到指定文件
9. // -r，从指定文件读取数据包信息
10. //tcpdump的过滤条件：
11. // 类型：host、net、port、portrange
12. // 方向：src、dst
13. // 协议：tcp、udp、ip、wlan、arp、……
14. // 多个条件组合：and、or、not

3）执行FTP访问，并观察tcpdump抓包结果

从192.168.4.100访问主机192.168.4.5的vsftpd服务。

1. **[**root@client **~]**# yum **-**y install ftp
2. **[**root@client **~]**# ftp **192.168.4.5**
3. Connected to **192.168.4.200** **(192.168.4.200).**
4. **220** **(**vsFTPd **3.0.2)**
5. Name **(192.168.4.200:**root**):** tom //输入用户名
6. **331** Please specify the password**.**
7. Password**:** //输入密码
8. **530** Login incorrect**.**
9. Login failed**.**
10. ftp**>**quit //退出

观察抓包的结果（回到porxy主机观察tcpdump抓包的结果）：

1. **[**root@proxy **~]**#
2. **...** …
3. **18:47:27.960530** IP **192.168.4.100.**novation **>** **192.168.4.5.**ftp**:** Flags **[**P**.],** seq **1:14,** ack **21,** win **65515,** length **13**
4. E**..5..**@**.**@**......**x**...**d**.\*..**G**.\**c**.**1BvP**.......**USER tom
5. **18:47:29.657364** IP **192.168.4.100.**novation **>** **192.168.4.5.**ftp**:** Flags **[**P**.],** seq **14:27,** ack **55,** win **65481,** length **13**
6. E**..5..**@**.**@**......**x**...**d**.\*..**G**.\**p**.**1B**.**P**.......**PASS **123**

4)再次使用tcpdump抓包，使用-w选项可以将抓取的数据包另存为文件，方便后期慢慢分析。

1. **[**root@proxy **~]**# tcpdump **-**A **-**w ftp**.**cap **\**
2. **>** host **192.168.4.5** and tcp port **21**                            //抓包并保存

tcpdump命令的-r选项，可以去读之前抓取的历史数据文件

1. **[**root@proxy **~]**# tcpdump **-**A **-**r ftp**.**cap **|** egrep '(USER|PASS)'    //分析数据包
2. **..** **..**
3. E**..(..**@**.**@**..** **...**x**...**d**.\*..**G**.\**c**.**1BbP**.............**
4. **18:47:25.967592** IP **192.168.4.5.**ftp **>** **192.168.4.100.**novation**:** Flags **[**P**.],** seq **1:21,** ack **1,** win **229,** length **20**
5. E**..<**FJ@**.**@**.**jE**...**d**...**x**...\*.**1BbG**.\**cP**...**V**...220** **(**vsFTPd **2.2.2)**
6. … …
7. **18:47:27.960530** IP **192.168.4.100.**novation **>** **192.168.4.5.**ftp**:** Flags **[**P**.],** seq **1:14,** ack **21,** win **65515,** length **13**
8. E**..5..**@**.**@**......**x**...**d**.\*..**G**.\**c**.**1BvP**.......**USER mickey
9. … …
10. **18:47:27.960783** IP **192.168.4.5.**ftp **>** **192.168.4.100.**novation**:** Flags **[**P**.],** seq **21:55,** ack **14,** win **229,** length **34**
11. E**..**JFL@**.**@**.**j5**...**d**...**x**...\*.**1BvG**.\**pP**...**i**~..331** Please specify the password**.**
12. … …
13. **18:47:29.657364** IP **192.168.4.5.**ftp **>** **192.168.4.100.**novation**:** Flags **[**P**.],** seq **14:27,** ack **55,** win **65481,** length **13**
14. E**..5..**@**.**@**......**x**...**d**.\*..**G**.\**p**.**1B**.**P**.......**PASS pwd123
15. … …
16. **18:47:29.702671** IP **192.168.4.100.**novation **>** **192.168.4.5.**ftp**:** Flags **[**P**.],** seq **55:78,** ack **27,** win **229,** length **23**
17. E**..?**FN@**.**@**.**j**>...**d**...**x**...\*.**1B**.**G**.\}**P**.......230** Login successful**.**

**步骤三：扩展知识，使用tcpdump分析Nginx的明文账户认证信息信息**

1）在proxy主机(192.168.4.5)准备一台需要用户认证的Nginx服务器

1. **[**root@proxy **~]**# cd **/**usr**/**local**/**nginx**/**conf**/**
2. **[**root@proxy **~]**# cp nginx**.**conf**.default** nginx**.**conf //还原配置文件
3. **[**root@proxy **~]**# vim **/**usr**/**local**/**nginx**/**conf**/**nginx**.**conf
4. server **{**
5. listen **80;**
6. server\_name localhost**;**
7. auth\_basic "xx"**;**
8. auth\_basic\_user\_file "/usr/local/nignx/pass"**;**
9. … …
10. **[**root@proxy **~]**# htpasswd **-**c **/**usr**/**local**/**nginx**/**pass jerry //创建账户文件
11. New password**:123** //输入密码
12. Re**-**type **new** password**:123** //确认密码
13. **[**root@proxy **~]**# nginx **-**s reload

2）在proxy主机使用tcpdump命令抓包

1. **[**root@proxy **~]**# tcpdump **-**A host **192.168.4.5** and tcp port **80**

3)在真实机使用浏览器访问192.168.4.5

1. **[**root@pc001 **~]**# firefox http**:**//192.168.4.5 //根据提示输入用户名与密码

4）回到proxy查看抓包的数据结果

1. **[**root@proxy **~]**# tcpdump **-**A host **192.168.4.5** and tcp port **80**
2. tcpdump**:** verbose output suppressed**,** use **-**v or **-**vv **for** full protocol decode
3. listening on eth0**,** link**-**type EN10MB **(**Ethernet**),** capture size **262144** bytes
4. … …
5. Authorization**:** Basic dG9tOjEyMzQ1Ng**==**
6. … …

5) 查看base64编码内容

1. **[**root@proxy **~]**# echo "dG9tOjEyMzQ1Ng==" **|** base64 **-**d
2. tom**:123456**
3. **[**root@proxy **~]**# echo "tom:123456" **|** base64
4. dG9tOjEyMzQ1Ngo**=**