TTS 10.0 COOKBOOK

(NSD SECURITY DAY04)

版本编号 10.0

2019-06

达内 IT 培训集团

NSD SECURITY DAY04

1. 案例 1：加密与解密应用

- **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

- **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）

- **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

```
[root@proxy ~]# vim file1.txt
abcdef
123456779
[root@proxy ~]# cp file1.txt file2.txt
[root@proxy ~]# cat file1.txt > file3.txt
[root@proxy ~]# md5sum file?.txt                //文件内容一致，则校验和也不变
b92aa0f8aa5d5af5a47c6896283f3536  file1.txt
b92aa0f8aa5d5af5a47c6896283f3536  file2.txt
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

```
[root@proxy ~]# echo "x" >> file1.txt
[root@proxy ~]# md5sum file?.txt
6be3efe71d8b4b1ed34ac45f4edd2ba7  file1.txt
b92aa0f8aa5d5af5a47c6896283f3536  file2.txt
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）

```
[root@proxy ~]# yum -y install gnupg2          //安装软件
[root@proxy ~]# gpg --version                  //查看版本
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：

```
[root@proxy ~]# gpg -c file2.txt
.. ..
```

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。

```
[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。

```
[root@proxy ~]# gpg -d file2.txt.gpg > file2.txt    //解密后保存
gpg: 3DES 加密过的数据
.. ..                                                //根据提示输入正确密码
```

```
[root@proxy ~]# cat file2.txt                          //查看解密后的文件
abcdef
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)

```
[root@client ~]# gpg --gen-key                          //创建密钥对
… …
请选择您要使用的密钥种类:
   (1) RSA and RSA (default)                            //默认算法为 RSA
   (2) DSA and Elgamal
   (3) DSA (仅用于签名)
   (4) RSA (仅用于签名)
您的选择?                                               //直接回车默认(1)
RSA 密钥长度应在 1024 位与 4096 位之间。
您想要用多大的密钥尺寸?(2048)                          //接受默认 2048 位
您所要求的密钥尺寸是 2048 位
请设定这把密钥的有效期限。
        0 = 密钥永不过期
     <n>  = 密钥在 n 天后过期
     <n>w = 密钥在 n 周后过期
     <n>m = 密钥在 n 月后过期
     <n>y = 密钥在 n 年后过期
密钥的有效期限是?(0)                                   //接受默认永不过期
密钥永远不会过期
以上正确吗?(y/n)y                                      //输入 y 确认

You need a user ID to identify your key; the software constructs the user ID
from the Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"
真实姓名：UserA
电子邮件地址：UserA@tarena.com
注释：UserA
您选定了这个用户标识:
    “UserA (UserA) <UserA@tarena.com>”

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
您需要一个密码来保护您的私钥。

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
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。
```

```
gpg：正在检查信任度数据库
gpg：需要 3 份勉强信任和 1 份完全信任，PGP 信任模型
gpg：深度：0 有效性：  1 已签名：  0 信任度：0-，0q，0n，0m，0f，1u
pub   2048R/421C9354 2017-08-16
密钥指纹 = 8A27 6FB5 1315 CEF8 D8A0  A65B F0C9 7DA6 421C 9354
uid                  UserA (UserA) <UserA@tarena.com>
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：

```
[root@client ~]# gpg --list-keys                          //查看公钥环
/root/.gnupg/pubring.gpg
------------------------------
pub   2048R/421C9354 2017-08-16
uid                  UserA (User A) <UserA@tarena.com>
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：

```
[root@client ~]# gpg -a --export UserA > UserA.pub
//--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
[root@client ~]# scp UserA.pub 192.168.4.5:/tmp/
//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。

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

```
[root@proxy ~]# echo "I love you ." > love.txt
[root@proxy ~]# gpg -e -r UserA love.txt
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
//-e 选项是使用密钥加密数据
//-r 选项后面跟的是密钥，说明使用哪个密钥对文件加密
[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）

```
[root@client ~]# gpg -d love.txt.gpg > love.txt
您需要输入密码，才能解开这个用户的私钥：“UserA (UserA) <UserA@tarena.com>”
2048 位的 RSA 密钥，钥匙号 9FA3AD25，建立于 2017-08-16 (主钥匙号 421C9354)
                                                  //验证私钥口令
```

```
gpg: 由 2048 位的 RSA 密钥加密，钥匙号为 9FA3AD25、生成于 2017-08-16
     “UserA (UserA) <UserA@tarena.com>”
[root@client ~]# cat love.txt                          //获得解密后的文件内容
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。

```
[root@client ~]# tar zcf log.tar /var/log              //建立测试软件包
[root@client ~]# gpg -b log.tar                        //创建分离式数字签名
[root@client ~]# ls -lh log.tar*
-rw-rw-r--. 1 root root 170 8月  17 21:18 log.tar
-rw-rw-r--. 1 root root 287 8月  17 21:22 log.tar.sig
[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

```
[root@proxy ~]# gpg --verify log.tar.sig log.tar
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
gpg: 完好的签名，来自于 “UserA (UserA) <UserA@tarena.com>”
.. ..
```

2. 案例 2：使用 AIDE 做入侵检测

- **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：

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

- **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。

- **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

```
[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

```
[root@proxy ~]# vim /etc/aide.conf
@@define DBDIR /var/lib/aide                                    //数据库目录
@@define LOGDIR /var/log/aide                                   //日志目录
database_out=file:@@{DBDIR}/aide.db.new.gz                      //数据库文件名
//一下内容为可以检查的项目（权限，用户，组，大小，哈希值等）
#p:      permissions
#i:      inode:
#n:      number of links
#u:      user
#g:      group
#s:      size
#md5:    md5 checksum
#sha1:   sha1 checksum
#sha256:        sha256 checksum
DATAONLY =  p+n+u+g+s+acl+selinux+xattrs+sha256
//以下内容设置需要对哪些数据进行入侵校验检查
//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
//仅保留/root 目录，其他目录都注释掉
/root   DATAONLY
#/boot   NORMAL                                                 //对哪些目录进行什么校验
#/bin    NORMAL
#/sbin   NORMAL
#/lib    NORMAL
#/lib64  NORMAL
#/opt    NORMAL
#/usr    NORMAL
#!/usr/src                                                      //使用[!]，设置不校验的目录
#!/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

```
[root@proxy ~]# aide --init
AIDE, version 0.15.1
AIDE database at /var/lib/aide/aide.db.new.gz initialized.
//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）

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

3）入侵后检测

```
[root@proxy ~]# cd /var/lib/aide/
[root@proxy ~]# mv aide.db.new.gz aide.db.gz
[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

- **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

- **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

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

2）检查 192.168.4.100 主机是否可以 ping 通

```
[root@proxy ~]# nmap  -sP  192.168.4.100
Starting Nmap 6.40 ( http://nmap.org ) at 2018-06-06 21:59 CST
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for host3 (192.168.4.100)
Host is up (0.00036s latency).
MAC Address: 52:54:00:71:07:76 (QEMU Virtual NIC)
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

```
[root@proxy ~]# nmap -n -sP  192.168.4.100
Starting Nmap 6.40 ( http://nmap.org ) at 2018-06-06 22:00 CST
Nmap scan report for 192.168.4.100
Host is up (0.00046s latency).
MAC Address: 52:54:00:71:07:76 (QEMU Virtual NIC)
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

```
[root@proxy ~]# nmap  -n  -sP  192.168.4.0/24
Starting Nmap 5.51 ( http://nmap.org ) at 2017-05-17 18:01 CST
Nmap scan report for 192.168.4.1
Host is up.
Nmap scan report for 192.168.4.7
Host is up.
```

```
Nmap scan report for 192.168.4.120
Host is up (0.00027s latency).
MAC Address: 00:0C:29:74:BE:21 (VMware)
Nmap scan report for 192.168.4.110
Host is up (0.00016s latency).
MAC Address: 00:50:56:C0:00:01 (VMware)
Nmap scan report for 192.168.4.120
Host is up (0.00046s latency).
MAC Address: 00:0C:29:DB:84:46 (VMware)
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

```
[root@proxy ~]# nmap -sT 192.168.4.100
Starting Nmap 5.51 ( http://nmap.org ) at 2018-05-17 17:55 CST
Nmap scan report for 192.168.4.100
Host is up (0.00028s latency).
Not shown: 990 closed ports
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
25/tcp  open  smtp
80/tcp  open  http
110/tcp open  pop3
111/tcp open  rpcbind
143/tcp open  imap
443/tcp open  https
993/tcp open  imaps
995/tcp open  pop3s
MAC Address: 00:0C:29:74:BE:21 (VMware)

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

```
[root@proxy ~]# nmap -p 21-22 192.168.4.0/24
Starting Nmap 5.51 ( http://nmap.org ) at 2017-05-17 18:00 CST
Nmap scan report for 192.168.4.1
Host is up (0.000025s latency).
PORT   STATE SERVICE
21/tcp open  ftp
22/tcp open  ssh

Nmap scan report for 192.168.4.7
Host is up.
PORT   STATE    SERVICE
21/tcp filtered ftp
22/tcp filtered ssh

Nmap scan report for 192.168.4.120
Host is up (0.00052s latency).
PORT   STATE SERVICE
21/tcp open  ftp
22/tcp open  ssh
MAC Address: 00:0C:29:74:BE:21 (VMware)

Nmap scan report for pc110.tarena.com (192.168.4.110)
Host is up (0.00038s latency).
PORT   STATE  SERVICE
21/tcp closed ftp
22/tcp closed ssh
MAC Address: 00:50:56:C0:00:01 (VMware)

Nmap scan report for 192.168.4.120
Host is up (0.00051s latency).
```

```
PORT   STATE  SERVICE
21/tcp closed ftp
22/tcp closed ssh
MAC Address: 00:0C:29:DB:84:46 (VMware)

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

```
[root@proxy ~]# nmap   -sU  192.168.4.100              //指定-sU 扫描 UDP
53/udp   open         domain
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

```
[root@proxy ~]# nmap -A 192.168.4.100,5

Starting Nmap 5.51 ( http://nmap.org ) at 2017-05-17 18:03 CST
Nmap scan report for 192.168.4.100                    //主机 mail 的扫描报告
Host is up (0.0016s latency).
Not shown: 990 closed ports
PORT    STATE SERVICE  VERSION
21/tcp  open  ftp      vsftpd 2.2.2
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
| -rw-r--r--    1 0        0            1719 Aug 17 13:33 UserB.pub
| -rw-r--r--    1 0        0             122 Aug 13 05:27 dl.txt
| drwxr-xr-x    2 14       0            4096 Aug 13 09:07 pub
| -rw-rw-r--    1 505      505           170 Aug 17 13:18 tools-1.2.3.tar.gz
|_-rw-rw-r--    1 505      505           287 Aug 17 13:22 tools-1.2.3.tar.gz.sig
22/tcp  open  ssh      OpenSSH 5.3 (protocol 2.0)
| ssh-hostkey: 1024 86:be:d6:89:c1:2d:d9:1f:57:2f:66:d1:af:a8:d3:c6 (DSA)
|_2048 16:0a:15:01:fa:bb:91:1d:cc:ab:68:17:58:f9:49:4f (RSA)
25/tcp  open  smtp     Postfix smtpd
80/tcp  open  http     Apache httpd 2.2.15 ((Red Hat))
|_http-methods: No Allow or Public header in OPTIONS response (status code 302)
| http-title: 302 Found
|_Did not follow redirect to https://192.168.4.100//
110/tcp open  pop3     Dovecot pop3d
|_pop3-capabilities:  USER  CAPA  UIDL  TOP  OK(K)  RESP-CODES  PIPELINING  STLS
SASL(PLAIN)
111/tcp open  rpcbind
MAC Address: 00:0C:29:74:BE:21 (VMware)
No  exact  OS  matches  for  host  (If  you  know  what  OS  is  running  on  it,  see
http://nmap.org/submit/ ).
TCP/IP fingerprint:
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
OS:11ED90%P=x86_64-redhat-linux-gnu)SEQ(SP=106%GCD=1%ISR=10B%TI=Z%CI=Z%II=I
OS:%TS=A)OPS(O1=M5B4ST11NW6%O2=M5B4ST11NW6%O3=M5B4NNT11NW6%O4=M5B4ST11NW6%O
OS:5=M5B4ST11NW6%O6=M5B4ST11)WIN(W1=3890%W2=3890%W3=3890%W4=3890%W5=3890%W6
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
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=
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%
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(
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=
OS:N%T=40%CD=S)

Network Distance: 1 hop
Service Info: Host:  mail.tarena.com; OS: Unix

TRACEROUTE
HOP RTT     ADDRESS
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）

```
[root@proxy ~]# yum -y install vsftpd
[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!!!

```
[root@proxy ~]# tcpdump -A host 192.168.4.5 and tcp port 21
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
.. ..                                                    //进入等待捕获数据包的状态
//监控选项如下：
// -i，指定监控的网络接口（默认监听第一个网卡）
// -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
//tcpdump 的过滤条件：
// 类型：host、net、port、portrange
// 方向：src、dst
// 协议：tcp、udp、ip、wlan、arp、……
// 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。

```
[root@client ~]# yum -y install ftp
[root@client ~]# ftp 192.168.4.5
Connected to 192.168.4.200 (192.168.4.200).
220 (vsFTPd 3.0.2)
Name (192.168.4.200:root): tom       //输入用户名
331 Please specify the password.
Password:                            //输入密码
530 Login incorrect.
Login failed.
ftp>quit                             //退出
```

观察抓包的结果（回到 porxy 主机观察 tcpdump 抓包的结果）：

```
[root@proxy ~]#
... …
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
E..5..@.@......x...d.*..G.\c.1BvP.......USER tom
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
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。

```
[root@proxy ~]# tcpdump  -A  -w  ftp.cap  \
> 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

```
[root@proxy ~]# tcpdump  -A  -r  ftp.cap | egrep  '(USER|PASS)'  //分析数据包
.. ..
E..(..@.@.. ...x...d.*..G.\c.1BbP.............
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
E..<FJ@.@.jE...d...x...*.1BbG.\cP...V...220 (vsFTPd 2.2.2)
… …
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
E..5..@.@......x...d.*..G.\c.1BvP.......USER mickey
… …
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
E..JFL@.@.j5...d...x...*.1BvG.\pP...i~..331 Please specify the password.
… …
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
E..5..@.@......x...d.*..G.\p.1B.P.......PASS pwd123
… …
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
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

```
[root@proxy ~]# cd /usr/local/nginx/conf/
[root@proxy ~]# cp nginx.conf.default  nginx.conf      //还原配置文件
[root@proxy ~]# vim /usr/local/nginx/conf/nginx.conf
server {
  listen 80;
  server_name localhost;
auth_basic "xx";
auth_basic_user_file "/usr/local/nignx/pass";
… …
[root@proxy ~]# htpasswd -c /usr/local/nginx/pass jerry   //创建账户文件
New password:123                  //输入密码
Re-type new password:123          //确认密码
[root@proxy ~]# nginx -s reload
```

2）在 proxy 主机使用 tcpdump 命令抓包

```
[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

```
[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

```
[root@proxy ~]# tcpdump -A host 192.168.4.5 and tcp port 80
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
```

```
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
… …
Authorization: Basic dG9tOjEyMzQ1Ng==
… …
```

5）查看 base64 编码内容

```
[root@proxy ~]# echo "dG9tOjEyMzQ1Ng==" | base64 -d
tom:123456
[root@proxy ~]# echo "tom:123456" | base64
dG9tOjEyMzQ1Ngo=
```