



WLAN Pi Deep Dive

Instructors: Jerry Olla & Ferney Munoz

Twitter: @jolla / @Ferney_Munoz

Session Overview

The goal of this session is to provide you with the knowledge and hands-on experience using the WLAN Pi to analyze and monitor Wi-Fi networks. The WLAN Pi is a versatile hardware and software platform that is specifically designed with wireless professional in mind. It has been loaded up with many Wi-Fi tools that can assist you in analyzing and troubleshooting Wi-Fi issues.

Essentials you'll need

Laptop - Windows or macOS

- With enough battery to make it through the 2hr session, power may be limited
- [Wireshark](#) (with SSHdump)
- SSH client (Windows 10 and macOS have built in SSH clients)
 - If needed, [Putty](#) is a free SSH client for Windows

What's included

1. **Custom WLAN Pi kit** - includes a 16GB microSD card, preloaded with all the fun-toys important Wi-Fi tools you'll be using, a Wi-Fi adapter (Comfast CF-912ac), USB-C cable, and USB-A adapter.



WLAN Pi Project

The WLAN Pi project started in 2016 at WLPC. The goal was to create a portable, ready-to-use device that could function as a network endpoint for measuring network performance and throughput.

Since then, it has been widely embraced in the awesome wireless community and after many contributions, this tiny box has evolved well beyond a network performance testing device.

Today, it can also be used as a remote Wi-Fi scanner, packet capture tool, portable Wi-Fi signal generator and much more! These capabilities assist wireless professionals with designing better wireless networks, troubleshooting issues more quickly, and validating wireless network performance.

Hardware

CPU

At the heart of the WLAN Pi is the NanoPi NEO2, a super tiny quad-core single-board computer (SBC). The NanoPi NEO2 shares some similarities to the more well known Raspberry Pi, however the NanoPi is less than ½ its size, consumes less power, and it can push over 900 Mbps over its gigabit ethernet connection.

Specs:

- CPU: Allwinner H5, Quad-core 64-bit high-performance Cortex A53
- DDR3 RAM: 1GB
- Storage: microSD (16 GB card included), can be expanded up to 128GB

More info: http://wiki.friendlyarm.com/wiki/index.php/NanoPi_NEO2

Wi-Fi Adapter

The Wi-Fi adapter is a Comfast CF-912ac, Realtek rtl8812au chipset, one of the few 802.11ac chipsets that support monitor mode and packet injection in linux.

Important Specs: Dual-band, 2x2:2, 802.11ac

Custom Handheld Case

Designed specifically for use with the WLAN Pi, houses the Wi-Fi adapter and USB-C cable, while still exposing the USB port and ethernet port on the bottom, while still allowing access to the MicroSD slot. Creation of Joel Crane (@Potato_Fi)



More info: <http://www.potatofi.com/2019/10/the-wlan-pi-handheld-case.html>

Software

Operating System

The WLAN Pi image is built on Armbian, a minimal Debian based Linux distro with a powerful configurator and software installer.

armbian

More info: <https://www.armbian.com/>

WLAN Pi image

The WLAN Pi image has been customized and pre-loaded with a comprehensive set of tools with Wi-Fi professionals in mind.



Latest releases: <https://github.com/WLAN-Pi/wlanpi/releases>

The Tools

Network Performance Measurement Tools

HTML5 network speed test - web based network speed test implemented in Javascript, using XMLHttpRequest and Web Workers

iPerf (iPerf2, iPerf3, and ePerf) - CLI tool for active measurements of the maximum achievable bandwidth on IP networks

Ruckus SpeedFlex (Zap) - Measure uplink and downlink throughput for UDP or TCP traffic between any AP and a Wi-Fi device, or between 2 devices, iOS and Android supported

Wi-Fi Analysis Tools

Kismet - wireless network and device detector, sniffer, wardriving tool, and WIDS (wireless intrusion detection) framework.

H.O.R.S.T. - lightweight 802.11 wireless LAN analyzer with a text interface

Scapy - packet crafting and analysis tool that uses Python

TCPDUMP - a powerful command-line packet analyzer

Aircrack-ng - a complete suite of tools to assess WiFi network security

bettercap - the Swiss army knife for network attacks and monitoring

WiFi Explorer Pro Remote Sensor - connects Wi-Fi Explorer Pro to a WLAN Pi remotely or locally to perform a passive Wi-Fi scan

WLAN Pi custom tools

WLAN Pi Profiler - tricks a Wi-Fi device into sending its association request frame so it can be decoded and analyzed to determine a devices Wi-Fi capabilities

WLAN Pi Hotspot - Turn your WLAN Pi into a Wi-Fi hotspot

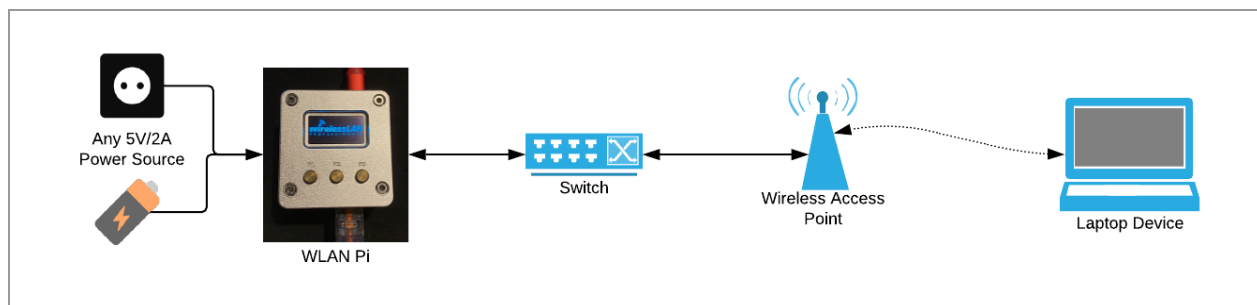
Wi-Fi Console - Turn your WLAN Pi in to a wireless serial console cable



Common Use Cases

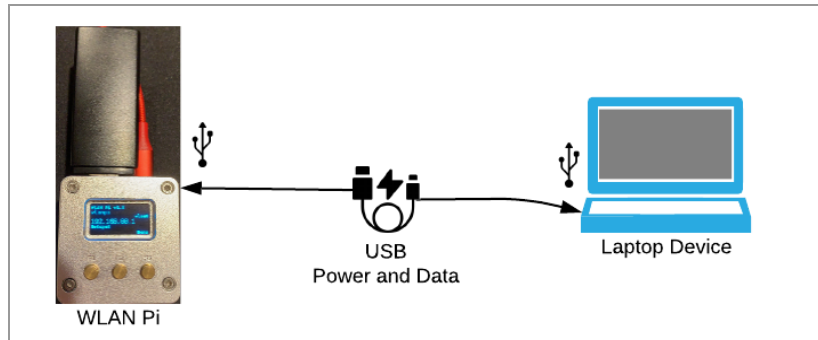
There are four common methods of using the WLAN Pi, either locally or remotely.

1. Wired - Remote access over ethernet port



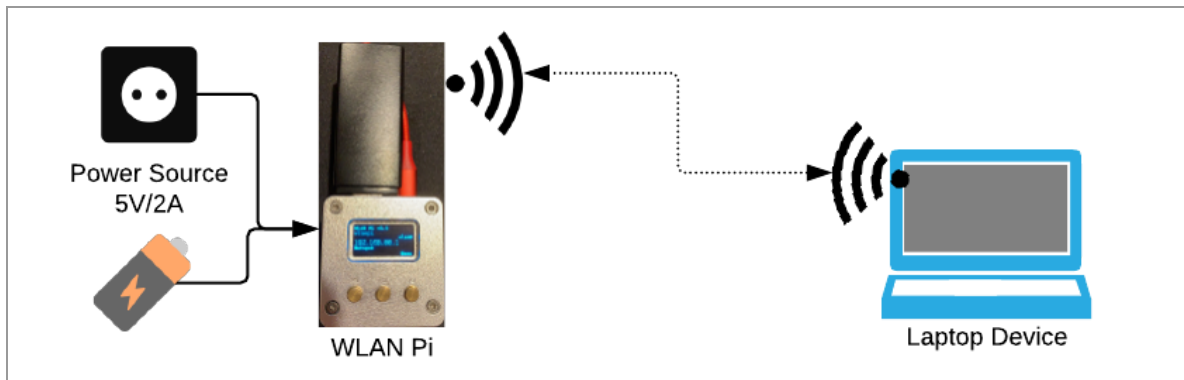
Ideal for network performance testing and Wi-Fi analysis, it allows the WLAN Pi to be positioned anywhere in the network chain to test the performance between 2 points of interest or through multiple hops.

2. Wired - Direct access over USB



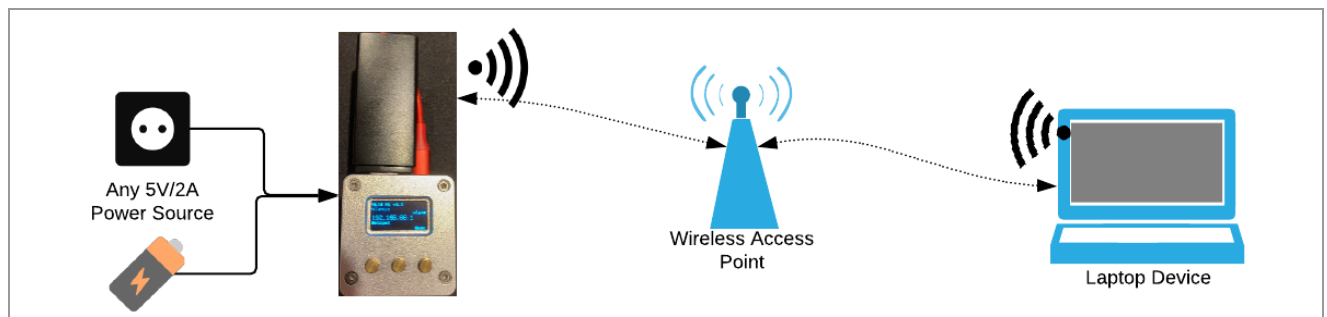
Ideal for portability, allows accessing the WLAN Pi and its included tools directly from any laptop, tablet, or even an iPad Pro. The WLAN Pi can be powered and accessed using a virtual ethernet connection over a single USB cable. No ethernet cable required!

3. Wireless - Direct access over Wi-Fi in hotspot mode



Ideal for using the WLAN Pi as a “Wi-Fi signal generator” or portable AP to test a Wi-Fi clients connectivity. Another use case is turning your WLAN Pi in to a wireless serial console cable, by connecting a USB serial cable to the USB port of the WLAN Pi.

4. Wireless - Remote access over Wi-Fi in client mode



Ideal for still accessing the WLAN Pi when you can't wire it, or when you need a Wi-Fi device to test Wi-Fi connectivity.



WLAN Pi Quick Reference Sheet

Default Username & Password

	Username	Password
Linux/SSH login	wlanpi	wlanpi
Bettercap web login	wlanpi	wlanpi

Network interfaces

Interface	Mode	IP
usb0 (usb OTG port)	Static	192.168.42.1
eth0 (wired ethernet)	Dynamic (DHCP)	Varies
wlan0 (Wi-Fi client)	Dynamic (DHCP)	Varies
wlan0 (hotspot mode)	Static	192.168.88.1

mDNS / Avahi Address

wlanpi.local

Common Ports

Application	Port
SSH	22
iperf2	5001
iperf3	5202
eperf	5203
zap (SpeedFlex)	18301
Kismet	http://{ip}:2501
Bettercap	https://{ip}:443
Wi-Fi Explorer Sensor	26999



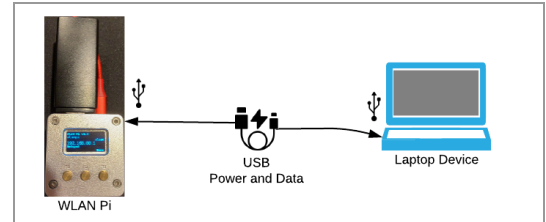
Hands-On Labs

Lab 1 - Accessing the WLAN Pi	10
Lab 2 - Linux Basics	15
Lab 3 - Remote Packet Capture	19
Lab 4 - Kismet - Detect Wi-Fi APs and Devices	25
Lab 5 - HORST - Analyze Wi-Fi Traffic in Real-Time	34
Lab 6 - Bettercap - Wi-Fi Reconnaissance	38
Lab 7 - Profiler - Determine Wi-Fi Device Capabilities	43
Lab 8 - Measure Wi-Fi Network Performance	48
Lab 9 - Portable Wi-Fi Signal Generator	53
Lab 10 - Customize Your WLAN Pi	55
Bonus Lab - Wi-Fi Explorer Pro	61
Bonus Lab - iPad Pro	65
Bonus Lab - Metageek Eye P.A.	69

Lab 1 - Accessing the WLAN Pi

In this lab we'll establish an SSH session to your WLAN Pi to gain access to the linux shell.

Connection Method: **Wired - Direct access over USB**

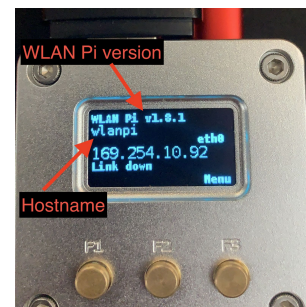


1.1 - Connect your WLAN Pi

Plug your WLAN Pi into any available USB port on your laptop. Your WLAN Pi should power on automatically and begin booting up. This process takes about 30 seconds.



When finished booting, the front display should look similar to this:



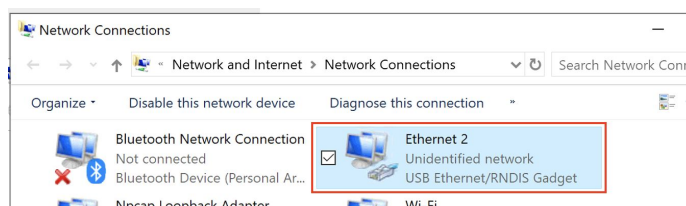
1.2 - Verify USB Communication

The WLAN Pi utilizes a USB OTG port to automatically create a virtual ethernet connection between the WLAN Pi and the host device it is plugged into. This enables the ability to power and communicate with the WLAN Pi using a signal USB cable.

Check to make sure the WLAN Pi was able to successfully create a virtual ethernet connection.



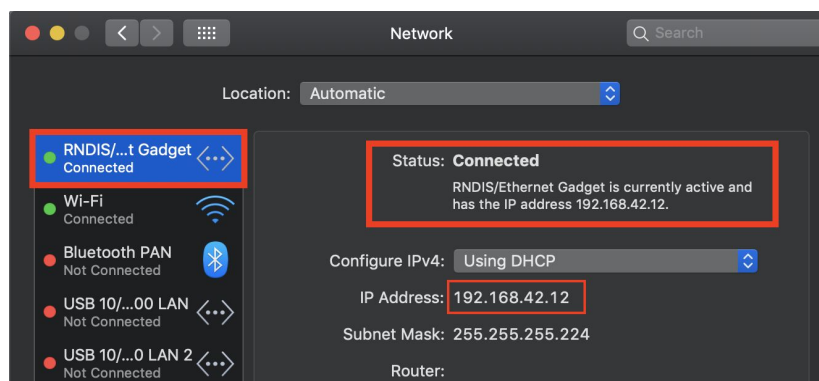
Verify there is a new **USB Ethernet/RNDIS Gadget** connection listed under Network Connections



If successful, the interface will have been assigned an IP address in the range of **192.168.42.2 - 50**



Check your network preferences, you should see something similar to the following. If so you are good to proceed to the next step.



1.3 - Establish an SSH connection

Open your preferred SSH client and connect to the WLAN Pi using the following:

IP: **192.168.42.1**

Username: **wlanpi**

Password: **wlanpi**

Step by Step Instructions:



Windows 10 finally includes an SSH client built in with the April 2018 update!

Note: If your Windows computer doesn't have an SSH client built in, you'll need a 3rd party SSH client. For your convenience, Putty and Solar-PuTTY installers have been included on the WLAN Pi image. Browse to <http://192.168.42.1> and click on Downloads to find the installers.

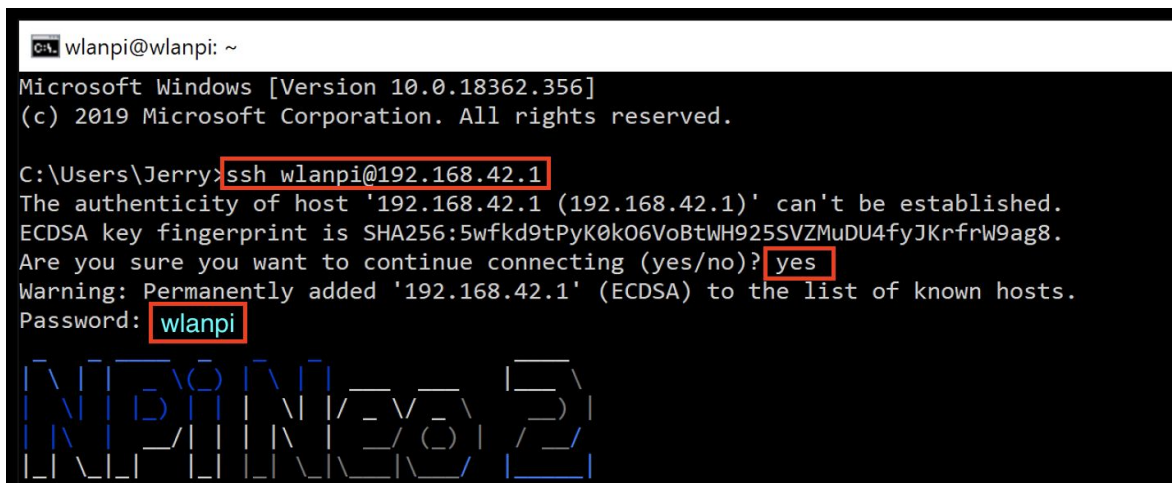
Using the built in Windows 10 SSH client:

Open a command prompt, and enter the following command:

```
ssh wlanpi@192.168.42.1
```

Password = wlanpi

You should see something like this:

A screenshot of a Windows command prompt window. The title bar shows "wlanpi@wlanpi: ~". The text in the window is as follows:
Microsoft Windows [Version 10.0.18362.356]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Jerry>ssh wlanpi@192.168.42.1
The authenticity of host '192.168.42.1 (192.168.42.1)' can't be established.
ECDSA key fingerprint is SHA256:5wfkD9tPyK0k06VoBtWH925SVZMuDU4fyJKrfrW9ag8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.42.1' (ECDSA) to the list of known hosts.
Password: wlanpi

At the bottom of the terminal, the text "NEON" is displayed in a large, stylized, blue, outlined font.

Congratulations, you now have shell access to your WLAN Pi!

Step 1 - Open terminal and type the following:

 γ_{\perp}

```
| ~ @ MacBook-Pro (jolla)
| => ssh wlanpi@192.168.42.1
The authenticity of host '192.168.42.1 (192.168.42.1)' can't be established.
ECDSA key fingerprint is SHA256:5wfkD9tPyK0kO6VoBTWH92SSVZMuDU4fyJKrfrW9ag8.
Are you sure you want to continue connecting (yes/no)?
```

If successful, you should see the following:

[illegible]

Congratulations, you now have shell access to your WLAN Pi!



Lab 2 - Linux Basics

The WLAN Pi runs Linux and many of the commands and features available are operated from the command line. Therefore, it makes sense to learn some basic Linux navigation and commands for controlling the WLAN Pi.

Basic commands to help you navigate the Linux operating system

ls	Display directory contents
cat filename	Display file contents
more filename	Display file contents one page at a time
cd directory	Change to the specified directory
cp source destination	Copy a file
mv source destination	Move a file or directory
df [-m]	Shows disk usage 1K blocks (or 1M)
rm filename	Remove a file
apt-get update	Update software package repository
apt-get upgrade	Upgrade updated software packages
sudo command	Run the specified command as root
nano filename	Text file editor
CTRL-C	Abort running command

The root user is the administrative user on Linux/UNIX computers. The **sudo** command allows you to execute a command as the root user without needing to login as the root user. When you run a command this way it will prompt you for the wlanpi user password and will not prompt you for the password again for 5 minutes after the last **sudo** command is executed. Using **sudo** is a best practice when working on Linux systems. However, entering **sudo** for every single command can get tedious at times. One option to simplify entering multiple **sudo** commands is to run **sudo bash**, which elevates all your commands to run as the root user.

Note: See the other laminated card with Linux Commands for further reference.

2.1 - Using the Linux Shell

The command prompt you see after login to the WLAN Pi is provided by the Bash shell. When you are a standard user, the prompt looks like this:

```
wlanpi@wlanpi:~$
```

To the left of the @ shows that you are logged in as the user wlanpi. To the right of the @ shows that you are on the computer named wlanpi. The colon (:) is a separator and the text between the colon and the dollar sign shows you the current working directory. The tilde (~) is a special character that indicates the user's home directory. In this case, the tilde represents **/home/wlanpi**. The dollar sign (\$) shows us that we are logged in as a standard user.

After you run the **sudo bash** command your prompt will change and will look like this:

```
root@wlanpi:/home/wlanpi#
```

The same information is present as before, but the username now shows you that you are the user "root", the current directory is no longer listed as tilde (~) because the root home directory is **/root**. Also, the dollar sign has changed to an octothorpe (aka pound sign, hash symbol, or number sign). These changes let you know you are running commands with elevated privileges.

Now it's time to put the above information to use.

Step 1. Change to the /var/log directory. This directory holds most of the Linux log files.

```
cd /var/log
```

Step 2. Let's use **cat** and **more** to look at a few log files that can help you troubleshoot.

Here's a list of log files of interest that you can take a look through:

messages Logs for many system processes and wireless subsystems

syslog DHCP and NTP logs

ufw.log Host firewall logs

auth.log Shows authentication logs

2.2 - Firewall

The WLAN Pi uses UFW (uncomplicated firewall), configuration of the firewall is out of scope for this session, but if you think the firewall is causing trouble, you can check the settings with **ufw status** and disable it with the **ufw disable** command. The **ufw disable** command saves across reboots, so if you want to turn it back on use **ufw enable**.

2.3 - Controlling Network Interfaces

The network interfaces can be controlled from the shell. The command for this is **ip**. The **ip** command allows you view interface status and to bring an interface up or down.

To show the status of all interfaces, run **ip addr**

To show the status of an individual interface, run **ip addr show [interface]**

To disable a network interface, **ip set [interface] down**

To enable a network interface, **ip set [interface] up**

Example:

```
wlanpi@wlanpi:~$ ip addr show usb0
```

2.4 - Controlling WLAN Interface

The wireless interface can also be controlled from the shell. The commands for this use **iwconfig**. The **iwconfig** command allows you view wireless specific status and change various wireless settings.

To show the status of all wireless interfaces, run **iwconfig**

To put the WLAN interface into monitor mode, run **iwconfig [interface] mode monitor**

To put the WLAN interface into managed (default) mode, run **iwconfig [interface] mode managed**

Example - putting wlan0 into monitor mode

```
wlanpi@wlanpi:~$ sudo iwconfig wlan0 mode monitor
```

Note: To sniff Wi-Fi traffic the WLAN interface needs to be in **Monitor** mode

Set the channel for the WLAN interface to listen on, run **iw [interface] set channel [channel#]**

Example - setting the adapter to listen on channel 36 using 40 MHz channel width

```
wlanpi@wlanpi:~$ sudo iw wlan0 set channel 36 HT40+
```



Lab 3 - Remote Packet Capture

Use your WLAN Pi as a remote network interface in Wireshark to capture raw wireless frames from the Wi-Fi NIC using tcpdump on the WLAN Pi. This method can also be used to do packet captures from a WLAN Pi deployed in a remote location.

3.1 - Install Wireshark (include SSHdump tool)

The Wireshark installers have been loaded onto the WLAN Pi image and can be accessed from the Downloads folder: <http://192.168.42.1>



macOS users - Install Wireshark, SSHdump tool is included by default

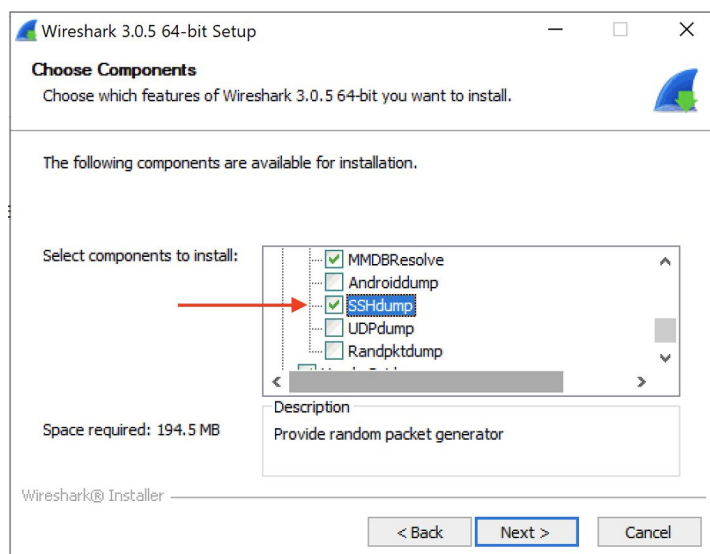


Windows Users - Install Wireshark and specify to include SSHdump

Step 1 - Start Wireshark installer

Step 2 - When you get to the “Choose Components” step, expand the **Tools** drop

Step 3 - Scroll down to the bottom of the list, check the box for **SSHdump**

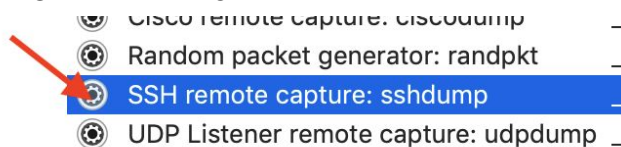


3.2 - Configure Wireshark

Step 1 - Open Wireshark

Step 2 - Under the **Capture** window, find **SSH remote capture: sshdump**

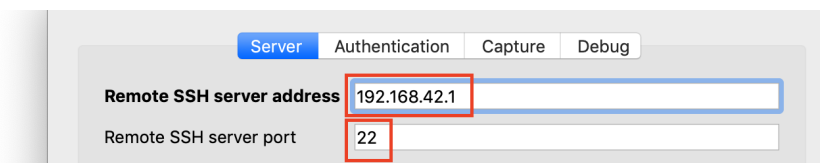
Step 3 - Click the **gear** to configure



Step 4 - Configure **Server** settings

Remote SSH Server address = **192.168.42.1**

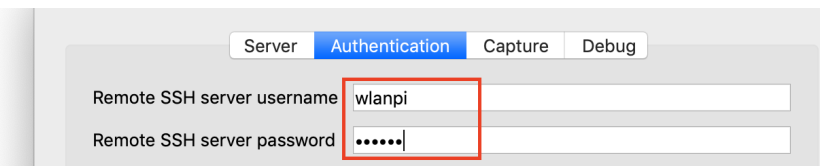
Remote SSH server port = **22**



Step 5 - Configure **Authentication** settings

Remote SSH server username = **wlanpi**

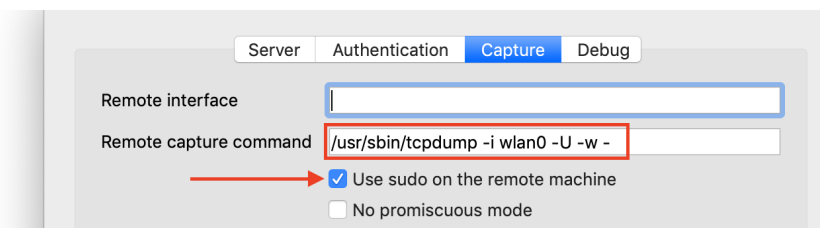
Remote SSH server password = **wlanpi**



Step 6 - Configure **Capture** settings

Remote capture command, type **/usr/sbin/tcpdump -i wlan0 -U -w -**

Check the box **Use sudo on the remote machine**



3.3 - Start capture

Step 1 - Click **Start**, frames should start showing up after a few seconds

3.4 - Change the capture channel

When troubleshooting a client or AP issue, you probably want to capture frames only on the channel the client and/or AP are using. Otherwise, you may miss the information you need for troubleshooting.

Set the capture channel on the WLAN Pi with **iw**:

```
wlanpi@wlanpi:~$ sudo iw wlan0 set channel <channel>
[NOHT | HT20 | HT40+ | HT40- | 5MHz | 10MHz | 80MHz]
```

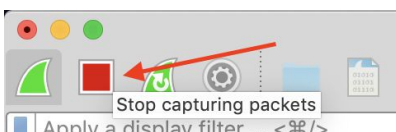
Example

To set channel 64 with a 80MHz wide channel:

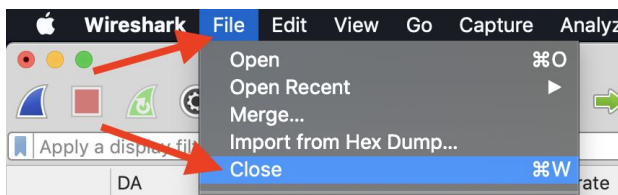
```
wlanpi@wlanpi:~$ sudo iw wlan0 set channel 64 80MHz
```

3.5 - Stop the capture

Step 1 - Click the stop button



Step 2 - Close the current capture to return back to the main screen



Step 3 - Click **Continue without Saving** (unless you really want to save= it)

BONUS Activity

Try out the new WLAN Pi EXTCAP plugin by Adrian Granados

<https://github.com/adriangranados/wlanpi-extcap>



Lab 4 - Kismet - Detect Wi-Fi APs and Devices

Kismet is a wireless network and device detector, sniffer, wardriving tool, and WIDS (wireless intrusion detection) framework.

In this lab, we'll explore the basics of using Kismet to sniff 802.11 traffic to detect nearby Wi-Fi networks and Wi-Fi devices.

More info: <https://www.kismetwireless.net>

4.1 - Starting Kismet

Kismet can be started and stopped using a Shell command or FPMS (Front Panel Menu System)

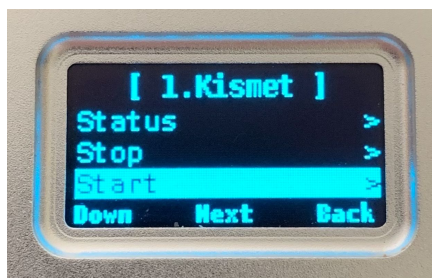
Using CLI

Kismet can be started by typing **kismet** into the WLAN Pi shell, using the SSH session we established in the previous section.

```
wlanpi@wlanpi:~$ kismet
```

Alternatively - Using FPMS (Front Panel Menu System)

Alternatively, Kismet can also be started or stopped using the front panel menu system, under the **Apps** menu page.



4.2 - Accessing Kismet Web UI

Step 1 - Open a web browser on your local machine and navigate to:
<http://192.168.42.1>

Step 2 - Click on “Kismet” from the top nav bar



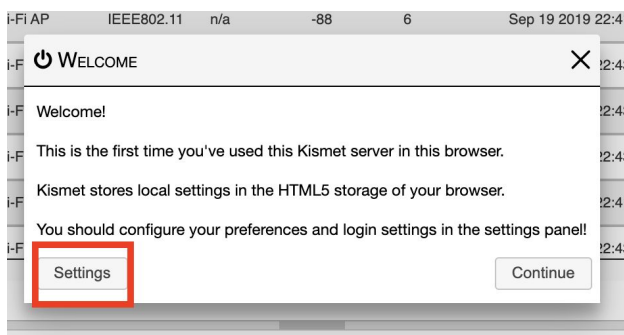
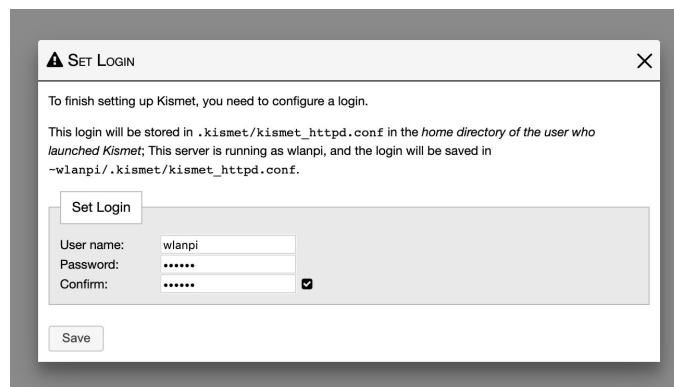
4.3 - Configuring Kismet

The first time you launch Kismet you'll be prompted to create a username and password.

Step 1 - Create a username and password for Kismet, for the purpose of this lab I would recommend using **wlanpi/wlanpi**

Step 2 - Click **Save**

Step 3 - Click **settings**



Step 4 - Edit the **Device List Columns** settings

Step 5 - Check the box to included **Manufacturer**

Step 6 - Click **Save Changes**

Step 7 - Edit the Device Row Highlighting settings

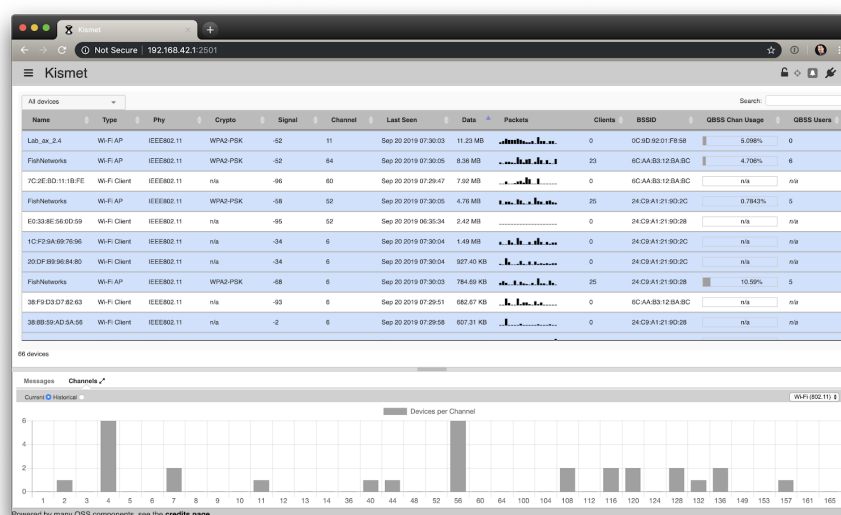
Step 8 - Check the box for **Active - this will highlight devices in blue that have been active in the past 10 seconds**

Step 9 - Click **Save Changes**

Step 10 - Close the settings window by clicking the X

4.4 - Explore the Kismet UI

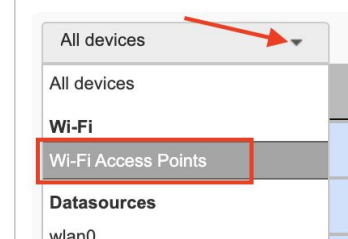
Step 1 - Kismet Web UI Overview



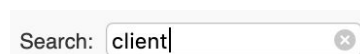
Step 2 - Filter to only show **Wi-Fi Access Points**

Step 3 - Click on a **Wi-Fi AP to drill down into more info about that AP**

Explore what info can you learn about the AP



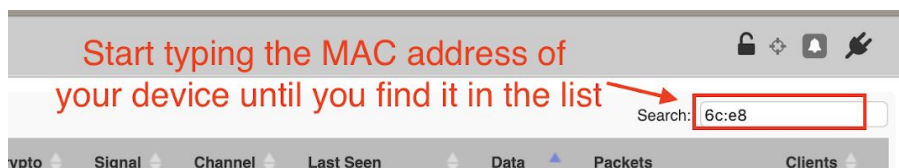
Step 3 - Use the search to filter only Wi-Fi Clients, type **client**



- Click on a **Wi-Fi Client** from the filtered list to reveal device details
- Explore what info can you find out about the Wi-Fi Client

4.5 - Search for your Wi-Fi device

Step 1 - Use the search box to filter the results, start typing your devices Wi-Fi radio MAC Address and see if it is in the list.

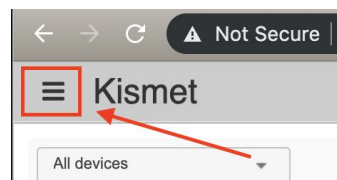


Step 2 - Click on your device from the list and see if you can determine which band and channel your device is connected.

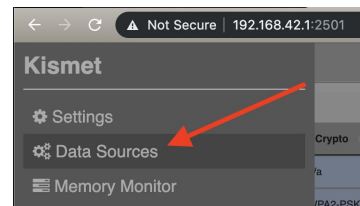
Note: You may notice your device was detected on multiple channels, why is this? Probe requests!

4.6 - Configure the Wi-Fi adapter

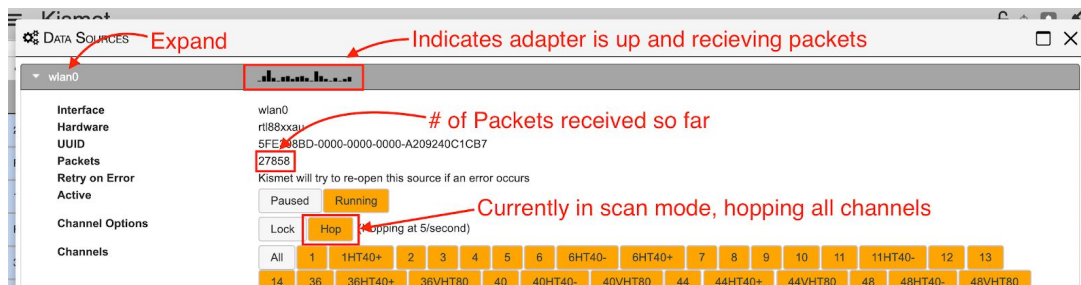
Step 1 - Select the Kismet menu



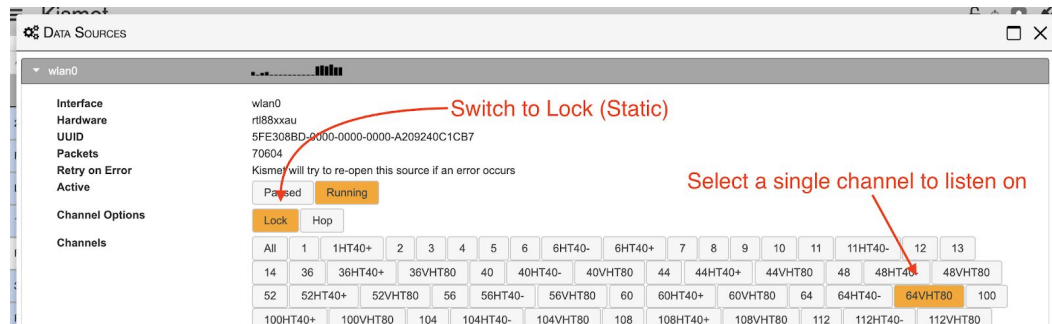
Step 2 - Select Data Sources



Step 3 - Expand the wlan0 interface, explore the Data Sources UI



Step 4 - Configure wlan0 to listen on a single channel, let's try the channel we identified in the previous step



4.7 - Shutting Down Kismet

Using SSH

Return to your SSH session and press CTRL+C to shutdown kismet or use the front panel menu system to shutdown kismet from the **Apps** menu.

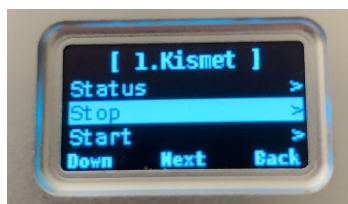
```
jolla — wlanpi@wlanpi: ~ — ssh wlanpi@192.168.42.1 — 80x19
KISMET - Point your browser to http://localhost:2501 (or the address of this sys
INFO: Detected new 802.11 Wi-Fi device 48:C7:96:D3:8A:95

INFO: Detected new 802.11 Wi-Fi device 00:0D:C5:D2:24:FC
^C
*** KISMET IS SHUTTING DOWN ***
Shutting down plugins...
WARNING: Kismet changes the configuration of network devices.
In most cases you will need to restart networking for
your interface (varies per distribution/OS, but
typically one of:
sudo service networking restart
sudo /etc/init.d/networking restart
or
nmcli device set [device] managed true

Kismet exiting.
wlanpi@wlanpi:~$
```

Alternatively - Use the FPMS (Front Panel Menu System)

From the front panel menu navigate to the App submenu, then Kismet, and Shutdown



Lab 5 - HORST - Analyze Wi-Fi Traffic in Real-Time

HORST is a powerful but lightweight 802.11 WLAN analyzer that uses a text interface. Its basic function is similar to tcpdump, Wireshark or Kismet, but it's much smaller and shows different, aggregated information which is not easily available from other tools. It is made for debugging wireless LANs with a focus on getting a quick overview instead of deep packet inspection.

Important note: One major caveat, HORST hasn't been updated in several years and thus is not able to analyze 802.11ac traffic properly. It will not include these frames in the statistical output.

5.1 - Prepare Wi-Fi adapter

Step 1 - Set Country Code using SSH

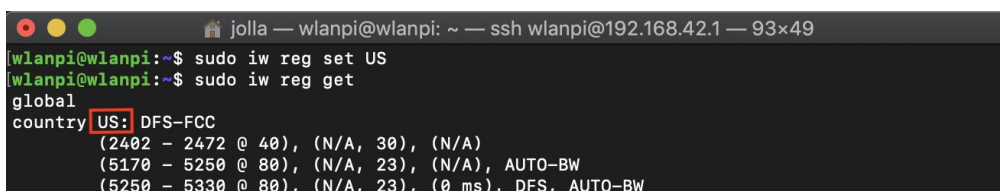
Type the following command into SSH:

```
wlanpi@wlanpi:~$ sudo iw reg set US
```

Step 2 - Verify the country code has been set correctly

Type the following command into SSH:

```
wlanpi@wlanpi:~$ sudo iw reg get
```



```
jolla — wlanpi@wlanpi: ~ — ssh wlanpi@192.168.42.1 — 93x49
[wlanpi@wlanpi:~$ sudo iw reg set US
[wlanpi@wlanpi:~$ sudo iw reg get
global
country US: DFS-FCC
(2402 - 2472 @ 40), (N/A, 30), (N/A)
(5170 - 5250 @ 80), (N/A, 23), (N/A), AUTO-BW
(5250 - 5330 @ 80), (N/A, 23), (0 ms), DFS, AUTO-BW
```

5.2 - Start HORST

Use SSH to start horst

```
wlanpi@wlanpi:~$ sudo horst
Password = wlanpi
```

5.3 - Change channel mode to scan

Since horst uses a text based UI we need to use some keyboard commands to configure things.

Step 1 - Press **c** - this will bring up the channel settings menu

Step 2 - Press **s** - this will put the adapter into scanning mode and it will start scanning Wi-Fi traffic across all channels

```

e0:33:8e:56:0d:59 -ST A 20
20:df:b9:72:8a:12 -ST AC 80 2x2
6c:a Channel Settings 000003d
7c:2
38:f 2.4GHz: HT40 2x2 5GHz: VHT80 2x2
4c:5 1 : 2412 HT40+ 36 : 5180 VHT80+
6c:e 2 : 2417 HT40+ 40 : 5200 VHT80+
38:8 3 : 2422 HT40+ 44 : 5220 VHT80+
6c:a 4 : 2427 HT40+ 48 : 5240 VHT80+ 0000006
24:c 5 : 2432 HT40+ 52 : 5260 VHT80+ 0000006
64:1 6 : 2437 HT40+ 56 : 5280 VHT80+
38:8 7 : 2442 HT40+ 60 : 5300 VHT80+
64:5 8 : 2447 HT40+ 64 : 5320 VHT80+
9 : 2452 HT40+ 100: 5500 VHT80+
24:c 10: 2457 HT40+ 104: 5520 VHT80+ 0000006
6c:a 11: 2462 HT40+ 108: 5540 VHT80+ 000003d
6c:a 112: 5560 VHT80+ 0000006
E24:c 116: 5580 VHT80+ 0000006
12:ba 120: 5600 VHT80+ rks' 3d
52:ba s: [*] Scan 128: 5640 VHT80+ ' 36e62
d7:82 d: Dwell: 250 ms 132: 5660 VHT80+ 2:ba:bc
21:9d u: Upper limit: 0 136: 5680 VHT80+ rks' 1d
61:9d m: Set channel: 64 140: 5700 VHT80+ ' 15d9d
VHT8 1: [ ] 20 (no HT) 144: 5720 VHT80+
VHT8 2: [ ] HT20 148: 5745 VHT80+
VHT8 4: [ ] HT40+ 153: 5765 VHT80+
VHT8 5: [ ] HT40+ 157: 5785 VHT80+
12:ba 8: [*] VHT80 161: 5805 VHT80+ rks' 3d
52:ba 6: [ ] VHT160 165: 5825 HT40+ ' 36e62
12:ba [ Press keys and ENTER to apply ] rks' 3d
52:ba:bc (6c:aa:b3:52:ba:bc) BEACON 'FishGuest' 36e62
VHT80 (center 5610)

```

Step 3 - Press **Enter** to close the Channel Settings window

5.4 - Explore the main HORST window

```

Pk/Re%-Cha-Sig-RAT-TRANSMITTER MODE-ST-MHz-TxR-ENCR-INFO
ESSID: 'FishNetworks' % of this node's packets in relation to all received packets
\ 13/26% 6 -53 % of retried frames of all frames this node sent 00000000eTc24324 (100)
/ 1/13% 6 -72
\ 1/6% 6 -7 spinning indicates there is activity 1x1
ESSID: 'FishGuest'
| 5/0% 6 -52 2 Signal value (RSSI) in dBm 0 0x3 WPA2 TSF 00000000efc095b8 (100)
- 1/0% 6 -67 6
NO ESSID: 6 -66 1 PHY data rate Lc AP N 20 0x2 TSF 0000000394635007 (100)
/ 4/0% 6 -46 1 4:8f:ca:50:52:be AP N 20 2x2 TSF 00000010a51d5180 (100)
| 1/0% 6 -67 24 00:25:9c:47:af:b0 AP G 20 WEP? TSF 0000000000000000 (0)
\ 11/6% 6 -67 1 fa:8f:ca:99:5c:14 AP N 20 0x1 TSF 000000126c810140 (100)
| 3/0% 6 -75 1 ac:bc:32:cd:84:29 AP N 20 0x3
- 0/0% 4 -29
Channel number Operating Mode (AP, AHD, PRB, STA, WDS)

```

5.5 - Explore other views

HORST Keyboard commands

Key	Command	Description
q	Quit	Quits HORST
p or <space>	Pause	Can be used to pause/resume horst. When horst is paused it will lose packets received in the meantime.
r	Reset	Clears all history and aggregated statistical data.
h	History	The history screen scrolls from right to left and shows a bar for each packet indicating the signal level. In the line below that, the packet type is indicated by one character (See NAMES AND ABBREVIATIONS) and the rough physical data rate is indicated below that in blue.
e	ESSID	The ESSID screen groups information by ESSID and shows the mode (AP, IBSS), the MAC address of the sender, the BSSID, the TSF, the beacon interval, the channel, the signal, a "W" when encryption is used and the IP address if known.
a	Statistics	The statistics screen groups packets by physical rate and by packet type and shows other kinds of aggregated and statistical information based on packets.
s	Spectrum Analyzer	It shows the available channels horizontally and vertical bars for each channel: Signal in green, Physical rate in blue, Channel usage in orange/brown By pressing the 'n' key, the display can be changed to show only the average signal level on each channel and the last 4 digits of the MAC address of the individual nodes at the level (height) they were received. This can give a quick graphical overview of the distance of nodes.
f	Filters	This configuration dialog can be used to define the active filters.
c	Channel Settings	This configuration dialog can be used to change the channel changing behaviour of horst or to change to a different channel manually.
o	Sort	Only active in the main screen, can be used to sort the node list in the upper area by Signal, Time, BSSID or Channel.

5.6 - Shutdown HORST

Press **[q]** or **[ctrl] + [c]** - either of these will exit HORST



Lab 6 - Bettercap - Wi-Fi Reconnaissance

bettercap is a powerful, easily extensible and portable framework written in Go which aims to offer to security researchers, red teamers and reverse engineers an easy to use, all-in-one solution with all the features they might possibly need for performing reconnaissance and attacking WiFi networks, Bluetooth Low Energy devices, wireless HID devices and Ethernet networks.

For the purpose of this lab we'll only be exploring the wifi module, this module includes WiFi network scanning, deauthentication attacks, clientless PMKID association attack and automatic WPA/WPA2 client handshakes capture.

More info: <https://www.bettercap.org>

6.1 - Start bettercap (with https web UI caplet)

Using SSH

```
wlanpi@wlanpi:~$ sudo bettercap -caplet https-ui
```

You should see the following:

```
jolla — wlanpi@wlanpi: ~ — ssh wlanpi@192.168.42.1 — 109x53
wlanpi@wlanpi:~$ sudo bettercap -caplet https-ui
[sudo] password for wlanpi:
bettercap v2.24.1 (built for linux arm64 with go1.11.4) [type 'help' for a list of commands]

[09:36:54] [sys.log] [war] Could not detect gateway.
[09:36:54] [sys.log] [inf] api.rest loading TLS key from /root/.bettercap-https.key.pem
[09:36:54] [sys.log] [inf] api.rest loading TLS certificate from /root/.bettercap-https.cert.pem
[09:36:54] [sys.log] [inf] api.rest api server starting on https://0.0.0.0:8083
[09:36:54] [sys.log] [inf] https.server loading server TLS key from /root/.bettercap-https.key.pem
[09:36:54] [sys.log] [inf] https.server loading server TLS certificate from /root/.bettercap-https.cert.pem
[09:36:54] [sys.log] [inf] https.server starting on https://0.0.0.0:443
169.254.0.0/16 > 169.254.10.92 »
```

Alternatively - Using FPMS (Front Panel Menu System)



6.2 - Access bettercap's web UI

Step 1 - Open a web browser and navigate to: <https://192.168.42.1>

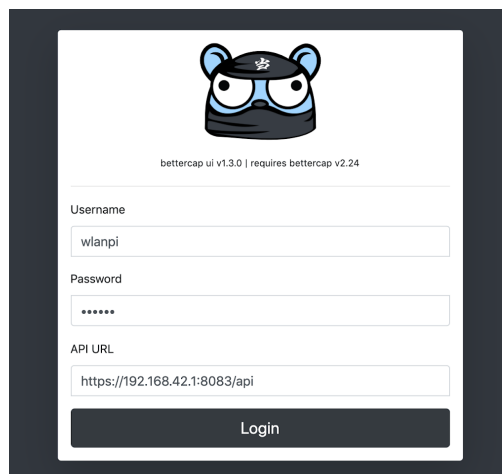
Notice: Requires using HTTPS

Step 2 - Accept the invalid certificate warning and proceed to the login page

6.3 - Login to bettercap

Username: **wlanpi**

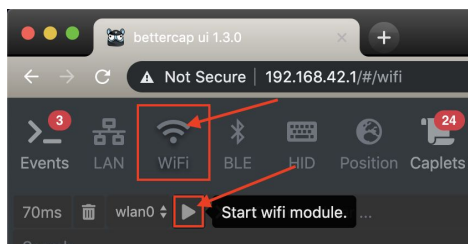
Password: **wlanpi**



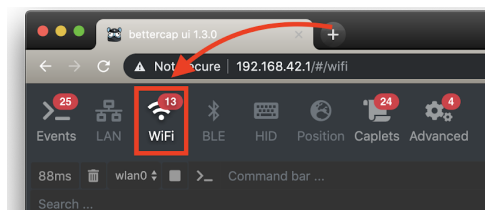
6.4 - Start the WiFi module

Step 1 - Select the **WiFi** module

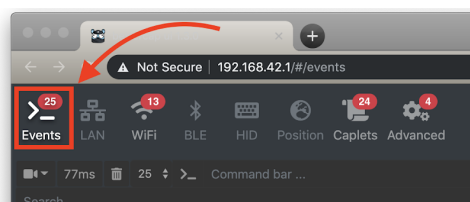
Step 2 - Click **play** to start the WiFi module



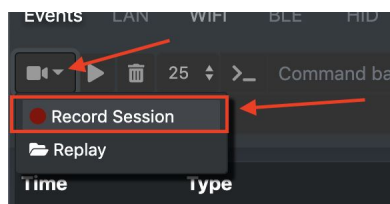
6.5 - Explore the WiFi module



6.6 - Select the Events module and explore



6.7 - Record a session

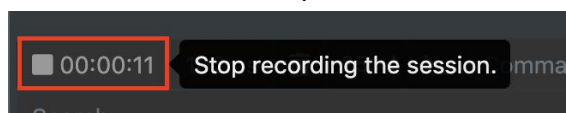


Step 1 - Click Record Session

Step 2 - Give the session a name, Example: "wlpc.record"

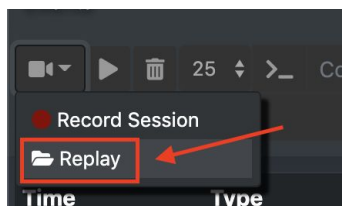


Step 3 - After a minute or so, click the stop button to end the recording

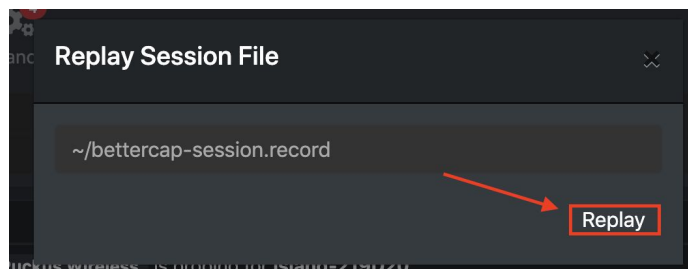


6.8 - Replay a recorded session

Step 1 - Click Replay



Step 2 - Specify the recorded session, then click Replay



Step 3 - Shutdown Bettercap from SSH or the FPMS: **CTRL + C**



Lab 7 - Profiler - Determine Wi-Fi Device Capabilities

One of the challenges we face working with Wi-Fi is determining the capabilities of a Wi-Fi device. Mike Albano (@mike_albano) maintains a database of device capabilities at <https://clients.mikealbano.com>. Where does this information come from? The clients themselves! When a client attempts to associate to an AP, it tells the AP what its capabilities are so that the AP knows how to talk to it.

Gathering this data used to be easier said than done, but thanks to [Nigel Boden](#) he has created the [WLAN Pi Profiler](#) which greatly simplifies this process.

WLAN Pi Profiler performs two primary functions:

1. Utilizes Scapy (python library) to create a "fake" access point by transmitting a forged beacon frames.
2. Listens for an association frame, decodes the frame, parsing out any relevant Wi-Fi capability information about the device.

Device information that Profiler can reveal:

- 802.11k/r/v/w support
- 802.11n/ac support
- Max Tx Power
- Max No. of Spatial Streams
- Supported 5 GHz channels

7.1 - Start Profiler

Step 1 - From the WLAN Pi shell, start the profiler script using a SSID of your choosing. (It helps if the channel and SSID are unique compared to the others in the class.)

Example:

```
wlanpi@wlanpi:~$ sudo profiler -c 48 -s "Firstname.Lastname"
```

-c sets the channel

-s specifies the SSID for the Fake AP

You should see something like this:

```
jolla — wlanpi@wlanpi: ~ — ssh wlanpi@192.168.42.1 — 106x20
wlanpi@wlanpi:~$ sudo profiler -c 48 -s "jerry.olla"
[sudo] password for wlanpi:

-----
Profiler AP Started

SSID: jerry.olla
PSK: 12345678
Channel: 48
Interface: wlan0
Results: http://192.168.42.1/profiler/
-----

#####
('Connect a Wi-Fi client to SSID:', 'jerry.olla', 'enter any random 8 characters for the PSK')
we don't really need the device to associate, we only need get the client to send an association request
#####

Created TUN interface fakeap at 10.0.0.1/24. Bind it to your services if needed.
```

7.2 - "Profile" a Wi-Fi Device

Step 1 - Choose the device you want to "profile" (Ex. your mobile phone). Attempt to associate it to the SSID you created. Enter any string of at least 8 characters when prompted for the PSK. Don't worry, your password will be wrong and that's OK. This network doesn't actually work. The goal is just to get the client device to transmit its association request frame which contains the info we are looking for.

Note: It may take your device a few scans before it detects the profiler SSID. The way Profiler forges and transmits beacon frames they aren't consistently transmitted every 102.5ms.

Once successful, you should see a report similar to this:

```
jolla — wlanpi@wlanpi: ~ — ssh wlanpi@192.168.42.1 — 88x24

-----
Client capabilities report - Client MAC: 6c:e8:5c:f2:84:5e
(OUI manufacturer lookup: Apple)
-----

802.11k      Supported
802.11r      Supported
802.11v      Supported
802.11w      Supported
802.11n      Supported (2ss)
802.11ac     Supported (2ss), SU BF not supported, MU BF not supported
Max_Power    21 dBm
Min_Power    -7 dBm
Supported_Channels 36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124,
128, 132, 136, 140, 144, 149, 153, 157, 161, 165

* Reported client capabilities are dependent on these features being available from the
wireless network at time of client association

[View PCAP & Client Report : http://192.168.42.1/profiler/clients/6c-e8-5c-f2-84-5e ]
```

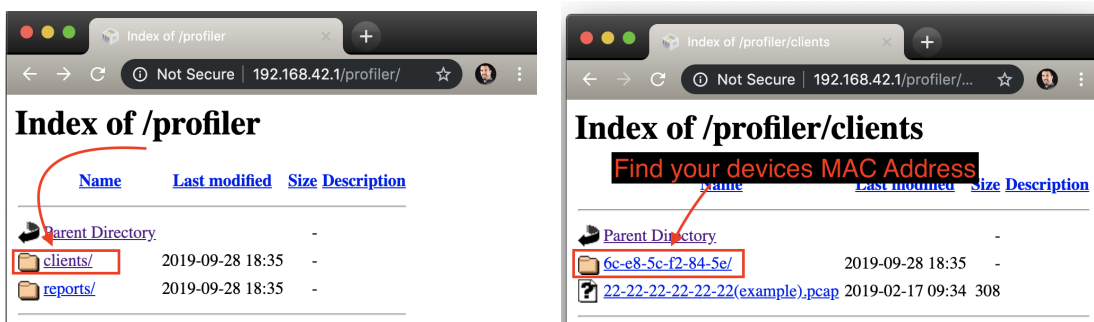
7.3 - Analyze the results

Step 1 - Browse to your WLAN Pi homepage (<http://192.168.42.1/>)

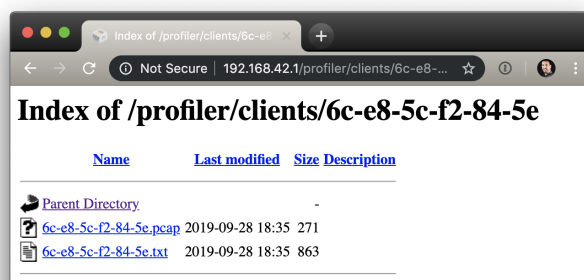
Step 2 - Click on the Profiler tab



Step 3 - You will have the option to choose clients or reports. The clients directory has a directory for the MAC address of each client that has been profiled. **Click on the clients directory -> find the MAC address of your client.**



Step 4 - In the MAC address directory, you will find a .txt file with the same textual report that we saw from the script. There is also a .pcap file with the packet data captured when the client tried to connect.



Step 5 - Download the .pcap file and open it in Wireshark.

Note: The clients file always shows the last time your client connected... it gets overwritten each time you restart the client profiler.

7.4 - Profile additional devices

Step 1 - Test all the devices you have with you to see their capabilities. You might also try testing devices with power save modes with the power save feature both enabled and disabled to see if it impacts the wireless capabilities. To perform the test, you just need each device to try connecting once.

Step 2 - This test only shows the capabilities for the band you are testing. Press **Ctrl-C** to stop the profiler. Start a new profiler on a 2.4GHz channel. Retest all your devices to obtain the 2.4GHz capabilities.

Note: Did it work? Were the files overwritten? How about the overall report?

7.5 - Download the results

In the Profiler section of the WLAN Pi webpage, click on **reports**. This includes a .csv file with all the data you just collected. There is one file for each band. Download the file and add a column for Device Type. Enter the type of device for each device. For example, "Samsung Galaxy 10." You now have started your own database of client capabilities.

7.6 - Share your results

Browse to <https://clients.mikealbano.com/> and look through the list to see what is there. Look to see if the devices you captured are already there. If not, you can click on the menu item "How to Contribute" and skip to step 2 since you have already captured the data!



Lab 8 - Measure Wi-Fi Network Performance

Wi-Fi problems often turn out to be problems on the wired side of the network. One of the tools we can use to validate performance is a throughput test. The WLAN Pi includes multiple tools to test network throughput, in this lab you'll learn how to use 2 different methods for measuring network performance.

Role of the WLAN Pi

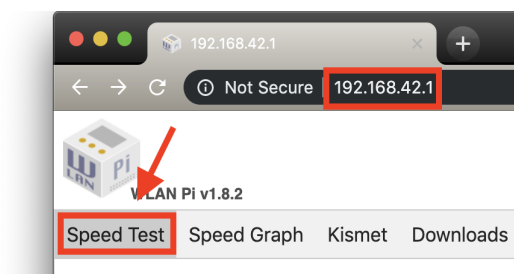
The WLAN Pi can act as a network endpoint. In the real world you would connect the WLAN Pi to the network using its gigabit ethernet port. It will attempt to obtain an IP address automatically via DHCP and can be placed at any location in a network where you need it for your testing. Remember, you are testing the entire network path between your client and the WLAN Pi.

For the purpose of these labs, we'll test our network throughput over the USB virtual ethernet connection. This will represent the routers, switches, and APs that would typically make up an enterprise network.

HTML5 Network Speed Test

The HTML5 network speed test is run between a web browser and the WLAN Pi. Any computer that can connect to the WLAN Pi and has a web browser that supports HTML5 can run this network speed test, including iOS and Android devices.

Step 1. Browse to the WLAN Pi homepage -> select Speed Test.



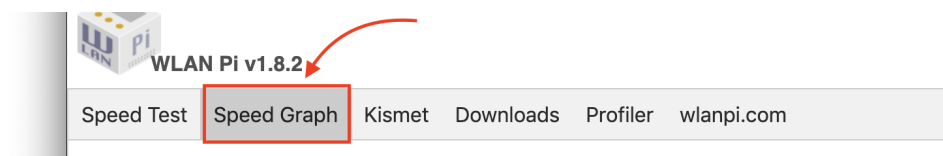
Step 2. Click Start.

Step 3. Observe the results of the speed test.

Graph the Network speed test output

It can be helpful to also visualize the network performance over a period of time. The Speed Graph will run the same test but will output the results on a graph showing performance over a period of time.

Switch to the Speed Graph from the top nav bar, and click Start



iPerf

iPerf is a widely used tool for network performance measurement and is supported on many platforms. While iPerf2 and iPerf3 share the same name and purpose, they are not compatible because iPerf3 is a complete rewrite. While iPerf2 does have some advantages for multiple streams and multiple users, iPerf3 is simpler and better supported. While the WLAN Pi does have iPerf2 and iPerf3, we will focus on iPerf3 for this lab.

Note: To use iPerf on mobile devices, checkout HE.NET Network Tools, available for [Android](#) and [iOS](#).



Windows Users

Step 1 - Download and extract the iPerf3 binaries from here:

<https://iperf.fr/download/windows/iperf-3.1.3-win64.zip>

Step 2 - Start Command Prompt

Step 3 - Use the **cd** command to change to the directory path where you extracted the iperf binaries

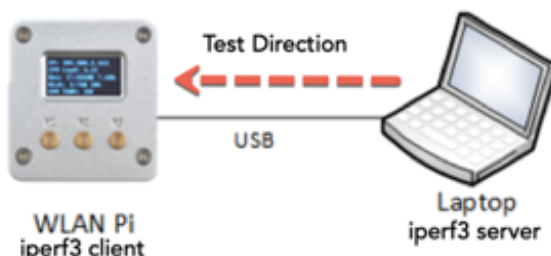
Example:

```
cd C:\Users\Jerry\Documents\iperf-3.1.3-win64\iperf-3.1.3-win64\
```

*Tip: Type **cd** and then drag and drop the folder into the command prompt window, it will auto populate the folder path*

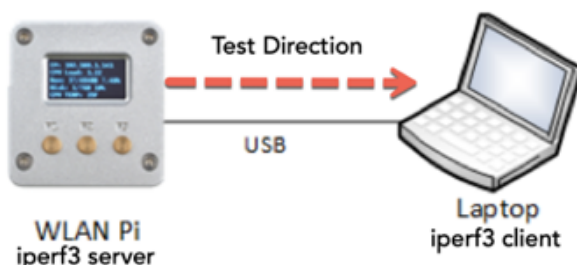
Step 4 - Run a test from your Laptop to your WLAN Pi (192.168.42.1) or the instructor WLAN Pi. This will test the upload speed from your host device to the WLAN Pi.

iperf3 -c 192.168.42.1



Step 5 - Run a reverse test from the WLAN Pi to your laptop. This will test the download speed from the WLAN Pi to your host device.

iperf3 -c 192.168.42.1 -R



macOS users

Step 1 - Download and extract the iPerf3 binaries from here:

https://iperf.fr/download/apple/iperf-3.1.3-macos-x86_64.zip

Step 2 - Start **Terminal**

Step 3 - Use the **cd** command to change to the directory path where you extracted the iperf binaries

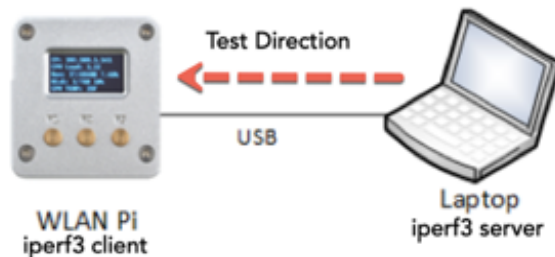
Example: **cd /Users/jolla/Downloads/**

*Tip: Type **cd** and then drag and drop the folder into the terminal window, it will auto populate the folder path*

Step 4. Run a test from your Macbook to your WLAN Pi (192.168.42.1) or the instructor WLAN Pi. This will test the upload speed from your host device to the WLAN Pi.

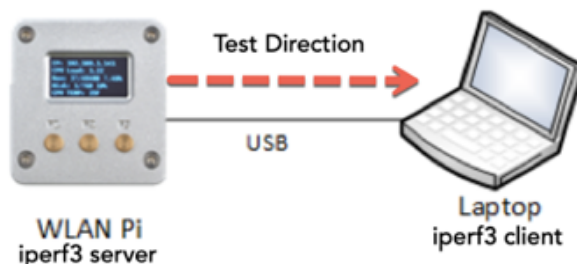
`./iperf3 -c 192.168.42.1`

Note: The `./` is important. If omitted the command will not be found.



Step 5. Run a reverse test from the WLAN Pi to your Macbook. This will test the download speed from the WLAN Pi to your host device.

`./iperf3 -c 192.168.42.1 -R`



Lab 9 - Portable Wi-Fi Signal Generator

Having a portable way to generate a Wi-Fi signal can be handy for several reasons, but one in particular is to measure RF attenuation of objects. The Front Panel Menu System (FPMS) makes it easy to turn the WLAN Pi into a Wi-Fi Signal Generator using Hotspot mode.

More info: <https://github.com/WLAN-Pi/wlanpi-hotspot>

9.1 Switch to Hotspot mode

In the real world you'll likely want to use a battery for this, but for the purpose of this lab, use your laptops USB Port to provide the WLAN Pi with power.

Step 1 - If the WLAN Pi front panel display isn't on press one of the buttons to power on the screen

Step 2 - Press the 3rd button to access the **Menu** on the FPMS

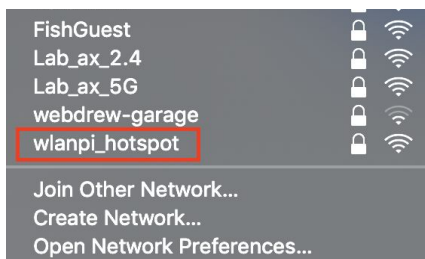
Step 3 - Press **button 1** several times to move down in the menu, until you get to **4. Actions**

Step 4 - Press **button 2** to access the **Actions** submenu

Step 5 - Press **button 1** to move down to **Hotspot**, press **button 2** to access the **Hotspot** submenu

Step 6 - Press **button 1** to move down to **Confirm**, press **button 2** to **Confirm**, the WLAN Pi will apply the hotspot configuration and reboot

Step 7 - You should see an SSID being broadcast from your WLAN Pi



Default PSK = wifipros

9.2 - Switch back to classic mode

When you want to return back to the classic mode,

Step 1 - On the Front Panel Menu System, access the **Menu** (button 3)

Step 2 - Navigate down the menu (button 1), select **Actions** (button 2)

Step 3 - Select **Classic Mode** (button 2), select **Confirm** and press button 2 to confirm.

The WLAN Pi will reboot and should return back to its defaults.



Lab 10 - Customize Your WLAN Pi

Since WLAN Pi runs on linux there are many things that can be done to customize your WLAN Pi! In this lab we provide several tasks you can try out to customize your WLAN Pi. If you mess something you, don't worry! You can always re-image the microSD card and reset everything back to its defaults.

Change the user password

Type the following linux command to change the wlanpi user password:

```
wlanpi@wlanpi:~$ passwd
```

Setup passwordless login

To set up passwordless SSH login all you need to do is to generate a public authentication key and append it to the remote hosts ~/.ssh/authorized_keys file.

The following steps will describe the process for configuring passwordless SSH login:

Step 1 - Before generating a new SSH key pair first check if you already have an SSH key on your client machine, you don't want to overwrite your existing keys.

macOS

Open **Terminal**, run the following **ls** command to see if existing SSH keys are present. If a key already exists skip to Step 3, otherwise proceed to Step 2

```
ls -al ~/.ssh/id_*.pub
```

Windows 10

Open **Command Prompt**, use the **dir** command to see if existing SSH keys are present. If a key exists skip to Step 3, otherwise proceed to Step 2

```
dir \.ssh /s /p
```

Step 2 - Generate a new SSH key pair.

 Use **Terminal**  Use **Command Prompt**

1. The following command will generate a new 4096 bits SSH key pair with your email address as a comment.

```
ssh-keygen -t rsa -b 4096 -C "your_email@domain.com"
```

2. Press **Enter** to accept the default file location and file name:

Enter file in which to save the key (/home/yourusername/.ssh/id_rsa):

3. Next, the ssh-keygen tool will ask you to type a secure passphrase.

Whether you want to use passphrase it's up to you, if you choose to use passphrase you will get an extra layer of security. In most cases, developers and system administrators use SSH without a passphrase because they are useful for fully automated processes. If you don't want to use passphrase just press **Enter**

Enter passphrase (empty for no passphrase):

Step 3 - Copy your public key to the WLAN Pi



In **Terminal**:

```
ssh-copy-id wlanpi@192.168.42.1
```



Use **Command Prompt**

```
scp .ssh\id_rsa.pub wlanpi@192.168.42.1:~\.ssh\authorized_keys
```

Step 4 - SSH into the WLAN Pi without a password

```
ssh wlanpi@192.168.42.1
```

Note: If successful, you should not be prompted for a password.

Change the Hostname

The hostname is configured under the /etc/hostname file. In order to change the hostname permanently, you will have to modify this file.

Step 1 - Open the file using the following command: **sudo nano /etc/hostname**

Step 2 - Modify the hostname to your liking

Step 3 - Save and close your file editor

Step 4 - Reboot the WLAN Pi

Customize the startup splash screen

Replace the image file with your own:

/home/wlanpi/NanoHatOLED/BakeBit/Software/Python/**wlanprologo.png**



Image size: 128x64px

Customize the front panel display and/or menu

This one is a bit more advanced, but if you are good with python or feel adventurous feel free to dig around in the code for the front panel display.

More info: <https://github.com/WLAN-Pi/BakeBit>

Edit the following python file:

/home/wlanpi/NanoHatOLED/BakeBit/Software/Python/**bakebit_nanohat_oled.py**

Customize the WLAN Pi web page

If you have some basic HTML knowledge you should be able to navigate and edit the fairly simple HTML home page.

Edit the index.html file:

/var/www/html/**index.html**

Change startup behavior:

There are 3 ways that are used to control what starts up boot up. The majority of the applications are set to run on reboot using Cron or as a Service.

Cron: Type **crontab -e** to view and modify what is configured to run on startup

crontab -e

rc.local: Edit the rc.local file

sudo nano /etc/rc.local

Services: The rest are services that are set to run automatically to view a list of services installed

List services:

```
systemctl list-unit-files
```

Start service:

```
systemctl start {SERVICENAME}
```

Stop service:

```
systemctl stop {SERVICENAME}
```

Enable service:

```
systemctl enable {SERVICENAME}
```

Disable service:

```
systemctl disable {SERVICENAME}
```



Bonus Labs



Bonus Lab - Wi-Fi Explorer Pro

Note: requires macOS

In this lab we'll use the WLAN Pi as an external/remote sensor for Wi-Fi Explorer Pro, this enables you to scan Wi-Fi networks without impacting your internal Wi-Fi adapters performance as well as perform Wi-Fi scans remotely over a network.

More info: <https://www.adriangranados.com/blog/wlanpi-as-a-sensor>

Direct connection over USB

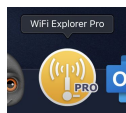
Step 1 - Install Wi-Fi Explorer Pro

Note: If you don't already own WFE Pro, a free 7-day trial can be downloaded from here:

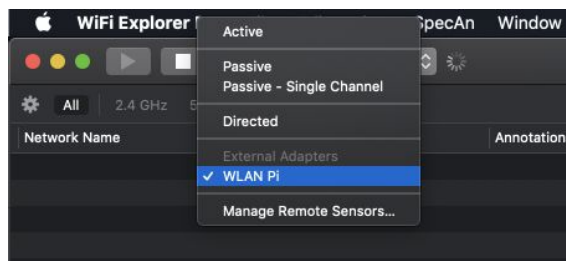
<https://www.adriangranados.com/apps/wifiexplorerpro-download>

Step 2 - Connect your WLAN Pi to one of your Macbooks USB ports

Step 3 - Start Wi-Fi Explorer Pro



Step 4 - Click the scan mode drop down menu, and select the WLAN Pi from the External Adapters list

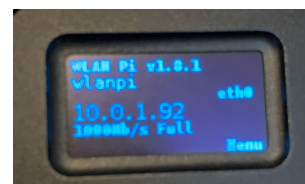


Note: this method only works when the WLAN Pi is connected locally to your Macbook.

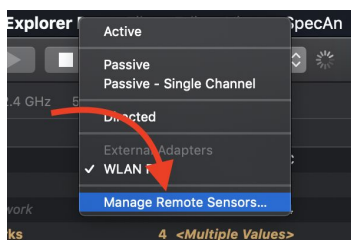
Remote sensor over ethernet

To access the WLAN Pi remotely on a network, add the WLAN Pi as an external sensor using the eth0 IP address of the WLAN Pi.

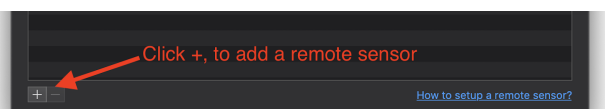
Step 1 - Obtain the eth0 IP address from the WLAN Pi, in this example the IP address = 10.0.1.92



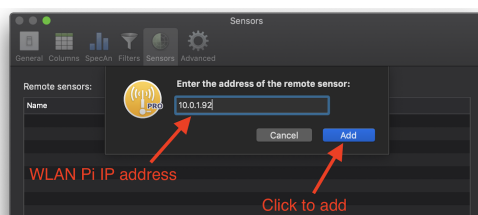
Step 2 - In Wi-Fi Explorer Pro, click on the **scan mode** menu, select **Manage Remote Sensors**



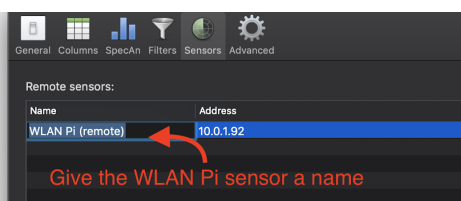
Step 3 - Add the WLAN Pi as a remote sensor



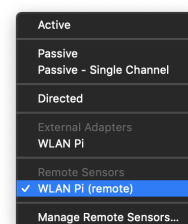
Step 4 - Type the IP address of the WLAN Pi's eth0 interface, and click **Add**



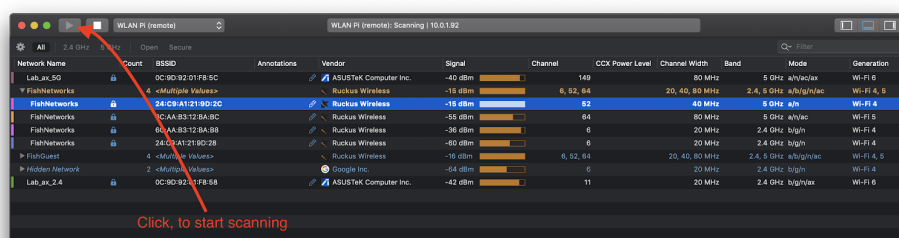
Step 5 - Give it a proper name



Step 6 - Select your remote WLAN Pi from the scan mode menu



Step 7 - Click the Start a scan button, after a few seconds you should start seeing scan results



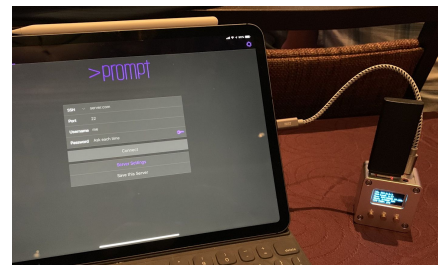


Bonus Lab - iPad Pro

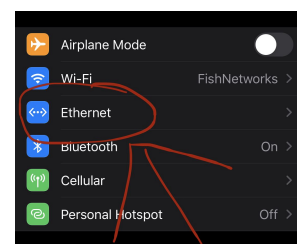
Note: Requires an iPad Pro with USB-C port

In this lab we'll cover how to connect and use a WLAN Pi with an iPad.

Step 1 - Connect the WLAN Pi to the USB-C port on the iPad Pro the WLAN Pi will begin to power on and then create an ethernet connection over the USB OTG port on the WLAN Pi.

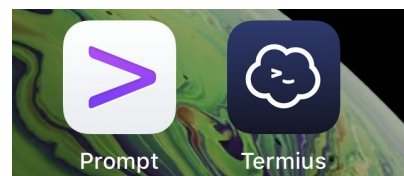


Step 2 - On the iPad, head over to Settings and confirm there is an ethernet connection

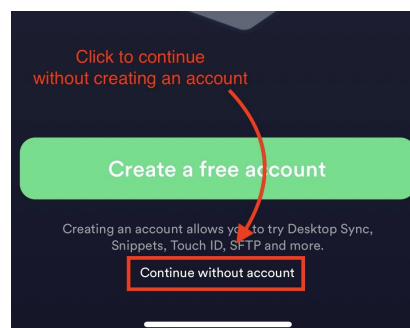


Step 3 - Install an SSH client App on your iPad, to SSH in to the WLAN Pi will require an SSH client. Thankfully there are multiple Apps. Here are two I'd recommend.

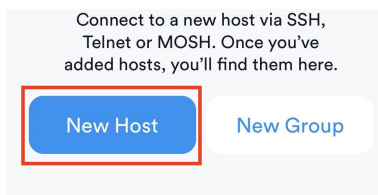
[Termius](#) = Free or [Prompt](#) = \$14.99



Step 4 - Setting up Termius



Step 5 - Create a new host

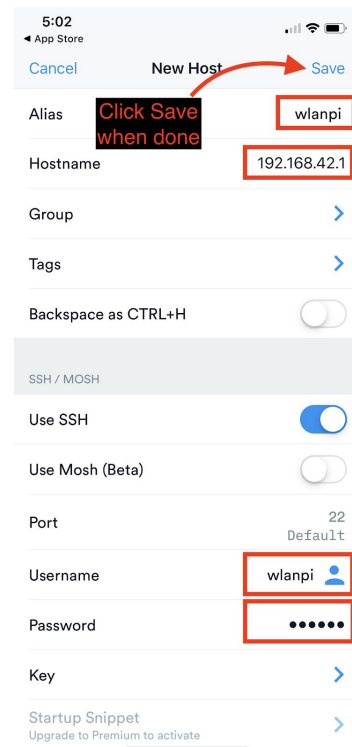
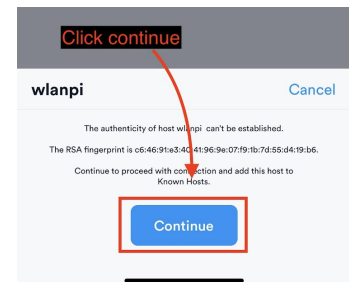
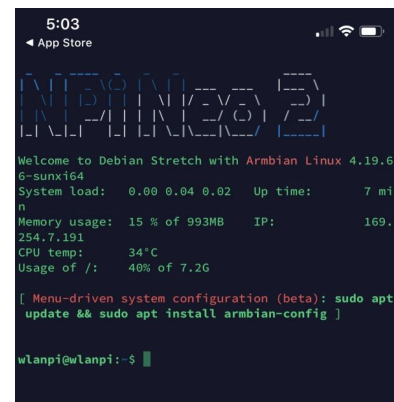


Step 6 - Add the WLAN Pi as a new host

Hostname = **192.168.42.1**

Username = **wlanpi**

Password = **wlanpi**

**Step 7 - Tap the alias for the WLAN Pi, click continue****Step 8 - You should now be at the linux shell**



Bonus Lab - Metageek Eye P.A.

Note: Requires Windows

Step 1 - Download and install the EyePA beta (2.3.0)

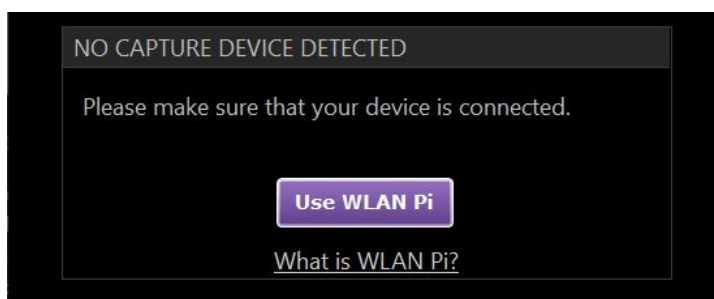
<http://metageek.link/wlanpi-installer>

Step 2 - Request an eval license

<https://www.metageek.com/products/eye-pa/request-trial.html>

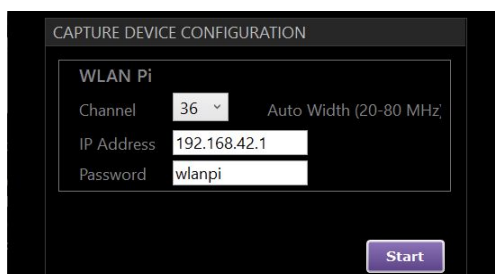
Alternatively, if you already have Eye PA installed, opt into the beta channel under **Help->Software Updates** and update to beta 2.3.0

Step 3 - Open EyePA and click the **CAPTURE** button on the right side of the START screen and you'll see this window:



Step 4 - Click **Use WLAN Pi**

Step 5 - Set the IP address of your WLAN Pi, click **Start** and it will send the commands to the WLAN Pi (over Plink) and start capturing.



Notes