

**Hardware Development Platform**

**USER GUIDE**

TABLE OF CONTENTS

[TABLE OF FIGURES 2](#_Toc2869057)

[LIST OF TABLES 3](#_Toc2869058)

[1 Introduction 4](#_Toc2869059)

[2 Purpose 4](#_Toc2869060)

[3 Required Equipment 4](#_Toc2869061)

[4 System Set-up 5](#_Toc2869062)

[5 RDK-B CONFIGURATION 6](#_Toc2869063)

[5.1 Image Flashing 6](#_Toc2869064)

[5.2 Basic Configuration 7](#_Toc2869065)

[6 WebUI 7](#_Toc2869066)

[6.1 Login 7](#_Toc2869067)

[7 Internet Access - LAN clients 10](#_Toc2869068)

[8 Advanced - Remote Management 10](#_Toc2869069)

[8.1 Setup: 10](#_Toc2869070)

[8.2 Steps: 11](#_Toc2869071)

[9 Advanced - DMZ 11](#_Toc2869072)

[9.1 Setup: 11](#_Toc2869073)

[9.2 Steps: 11](#_Toc2869074)

[10 Troubleshooting – Diagnostic Tools 11](#_Toc2869075)

[10.1 Setup: 11](#_Toc2869076)

[10.2 Steps: 12](#_Toc2869077)

[11 BRIDGE MODE 12](#_Toc2869078)

[11.1 Setup: 12](#_Toc2869079)

[11.2 Steps: 12](#_Toc2869080)

[12 Xfinity - Wifi 12](#_Toc2869081)

[12.1 Setup: 12](#_Toc2869082)

[12.2 Steps: 13](#_Toc2869083)

[13 Boot Time Measurement 13](#_Toc2869084)

[14 WebPA Client SupporT 17](#_Toc2869085)

[15 MAC FILTERING 19](#_Toc2869086)

[16 CAPTIVE PORTAL 22](#_Toc2869087)

[17 Wireless Protection Setup PORTAL 26](#_Toc2869088)

# TABLE OF FIGURES

|  |  |  |
| --- | --- | --- |
| S. No | Feature | Figure Name/Page No. |
| 1 | System Set up | Setup Diagram / 5 |

# LIST OF TABLES

**No tables entries found.**

# Introduction

Raspberry Pi series are low cost credit card sized single board computers. The Raspberry Pi models feature a Broadcom SoC that includes an [ARM](https://en.wikipedia.org/wiki/ARM_architecture) [CPU](https://en.wikipedia.org/wiki/Central_processing_unit) and an on chip [graphics-processing unit](https://en.wikipedia.org/wiki/Graphics_processing_unit) (GPU). Raspberry Pi uses SD card that facilitates users to swap out different versions of the operating system or software updates, hence, speeding the development and testing process. This is a great way to develop new applications/software and show these to customers in remote locations or at trade shows.

The RDK-B gateway stack runs on the Raspberry Pi 2 and 3 devices and provides all the features of gateway.

Please contact us at info@RDKcentral.com for more information.

# Purpose

This document contains set-up and other information to run the RDK-B gateway stack on a Raspberry Pi 2 and Raspberry Pi 3 devices.

# Required Equipment

The equipment listed below is required for complete functionality of the standalone environment. The only devices that are tested to work properly in this solution are the specific brands and models listed below. Support for other types of equipment is not available at this time.

Raspberry Pi 2 or Raspberry Pi 3 device

Standard USB keyboard

Television set/monitor with HDMI input.

Ethernet cables

USB-Ethernet Adapter

SD Card

# System Set-up

Raspberry Pi

USB

Keyboard

Client

Machine

LAN

USB

TV/Monitor

USB-Ethernet Adapter

HDMI

ETH

WAN

Power

Figure 1. Setup Diagram

1. Connect TV/Monitor to HDMI Video Output.
2. Connect Ethernet cable to ETH port.
   1. The other end of the Ethernet cable should be connected to the network where DHCP server is running so that the Raspberry Pi device gets assigned an IP address on boot-up.
   2. Connectivity to Internet is required so that the client connected can access the internet via RDK-B gateway.
3. Connect USB-Ethernet adapter to one of the USB slots present in Raspberry Pi.
4. Connect Ethernet cable from USB-Ethernet adapter to the client device/machine.
5. Insert the SD card into Micro SD Card Slot.
6. Connect Power cable to Micro USB power Input.

.

# RDK-B CONFIGURATION

## Image Flashing

1. **Linux:**

* Insert an SD card in either the SD card port of the laptop or the USB SD card reader.
* Verify that the SD card has been detected by executing either of the commands listed below:
  + # lsblk
  + # sudo fdisk –l

**Notes:** Remember to look at the size of the device to be sure that you have identified the device number associated with your SD card. We use “dd” command to write the generated image on the SD card, thus an incorrect identification may delete data from your hard drive or other device of importance. In most case, the hard disk drive will be at /dev/sda.

* Type the following command to ensure that the partitions, if present, on the SD card are not mounted
  + # mount
* Repeat the below command to unmount all the mounted partition present on the SD card.
  + # unmount <partition-mountpoint>
* Execute the following command to flash the image on the SD card
  + # sudo dd if=<r-pi sdimg> of=<SD card device> bs=1M

for ex:

# sudo dd if=rdk-mc-rpi.sdimg of=/dev/sdb bs=1M

* Repeat the below command to unmount all the mounted partition present on the SD card.
  + # unmount <partition-mountpoint>
* Remove the SD card and insert it to the Raspberry Pi SD card slot

1. **Mac**

* Follow the instructions provided in the link below to flash the image:

<https://www.raspberrypi.org/documentation/installation/installing-images/mac.md>

1. **Windows**

* Follow the instructions provided in the link below to flash the image:

<https://www.raspberrypi.org/documentation/installation/installing-images/windows.md>

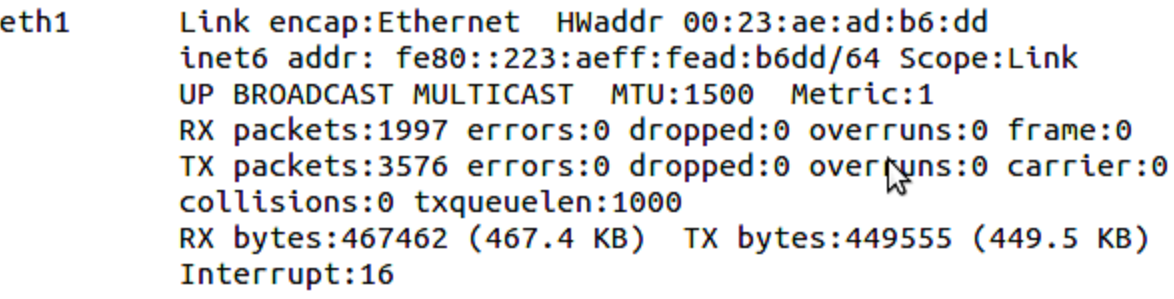
## Basic Configuration

* Insert the micro SD card with the flashed RDK-B image in the SD card slot of Raspberry Pi.
* Plug in the power cable to boot up the Raspberry Pi device
* Type “root” when the login prompt is displayed
* Run following in the terminal

# ifconfig

Check the interface shown for USB-Ethernet adapter

e.g eth1, eth2 … so on



* In case you are running in Ethernet mode, add interface of USB-Ethernet adapter in file “/etc/utopia/system\_defaults. Change lan\_ethernet\_physical\_ifnames. It will be your lan side.

e.g: lan\_ethernet\_physical\_ifnames=eth1

* Reboot the Raspberry Pi.

# WebUI

## Login

* WebUi can be accessed by both the Lan clients and from the WAN Side.

For Lan Clients:

* Open an internet browser on the Lan client/machine.
* Give the following url in the browser window:

From LAN Side:

<http://10.0.0.1>

From WAN Side:

http://<WAN IP Address of RaspberryPi>:8080

For e.g

<http://10.213.51.122>:8080

* Login page opens as shown in figure1 below.

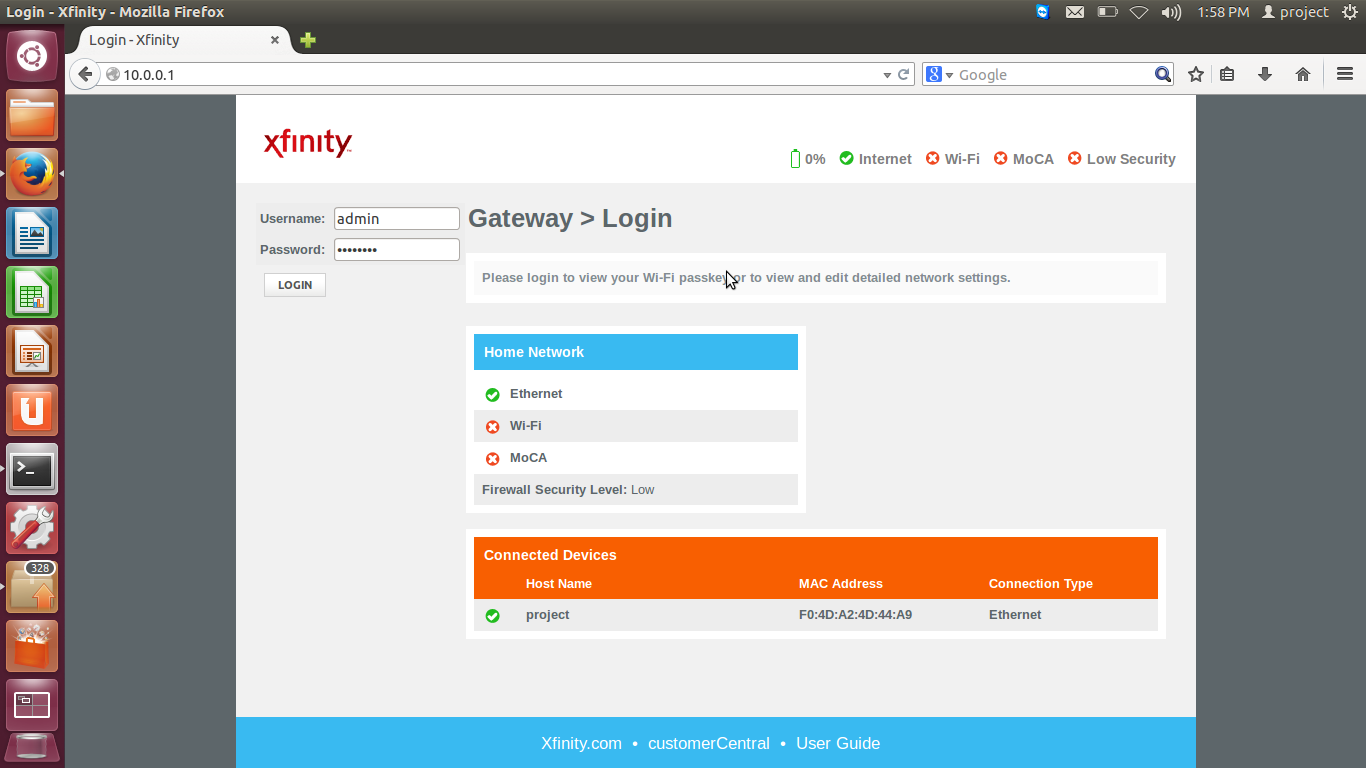


Figure 1: Login - Xfinity

* Enter Credentials:

Username: admin

Password: password

* Figure 2 will open as shown below

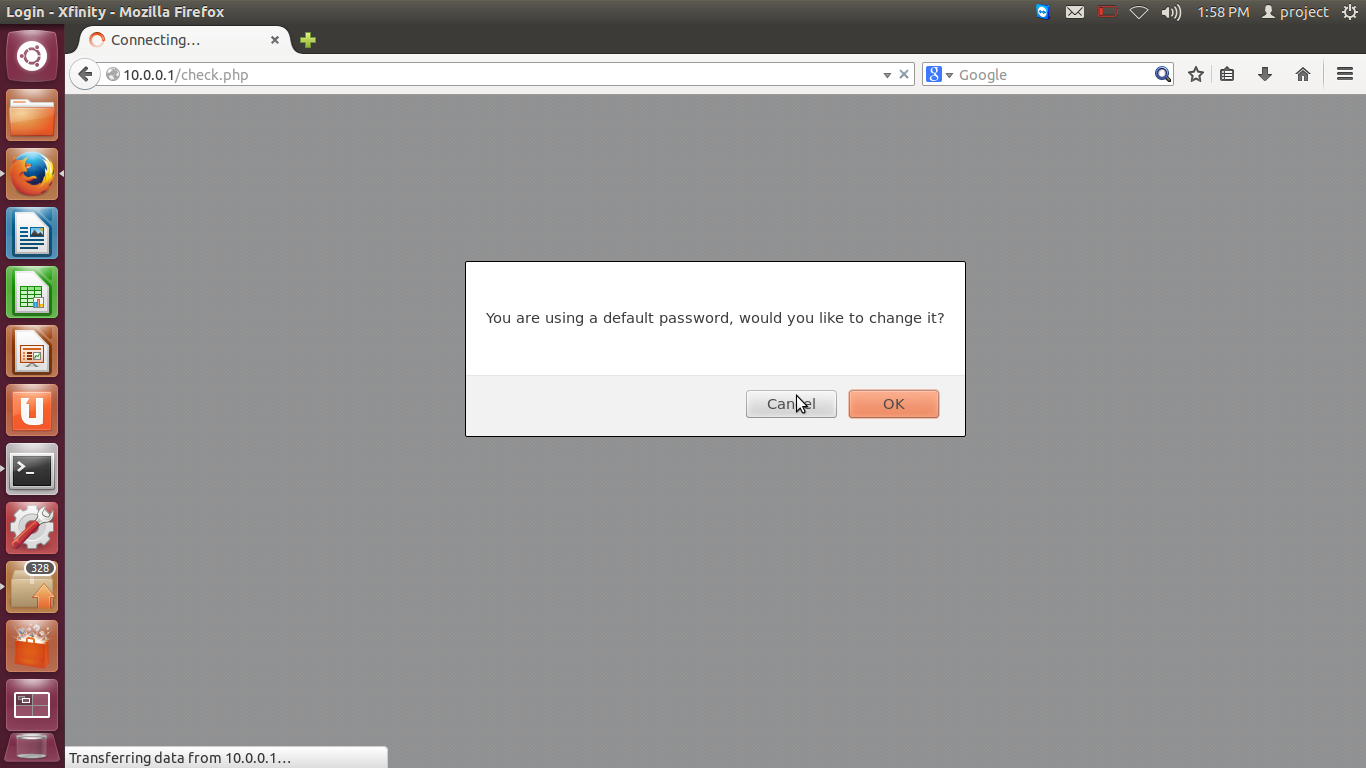


Figure 2

* Press the cancel tab, Figure 3 will open as shown below.

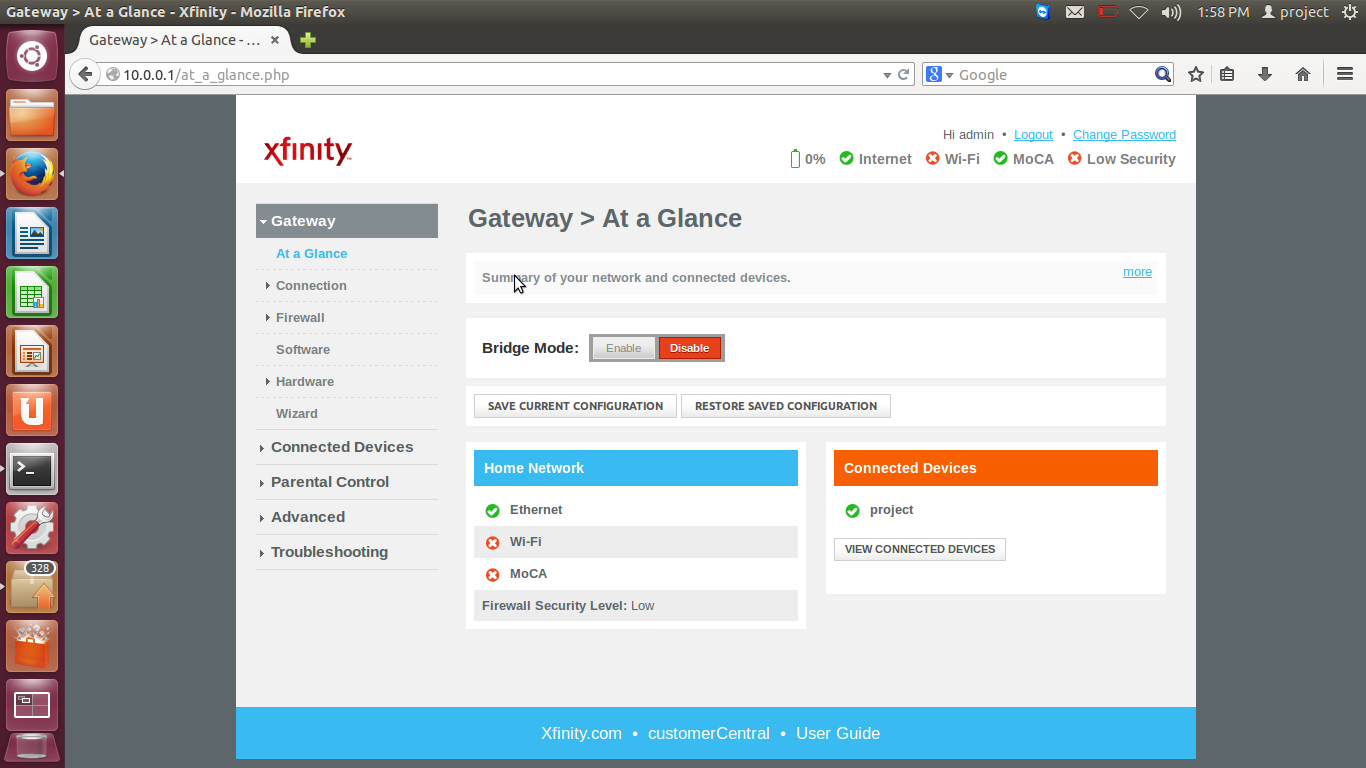


Figure 3: Xfinity

# Internet Access - LAN clients

* Lan clients can access the internet through RDK-B gateway.
* Open an internet browser on the Lan client/machine.
* Give the url in the browser window.

For e.g.

[www.google.com](http://www.google.com)

* Internet will be accessed as shown below in figure 4:

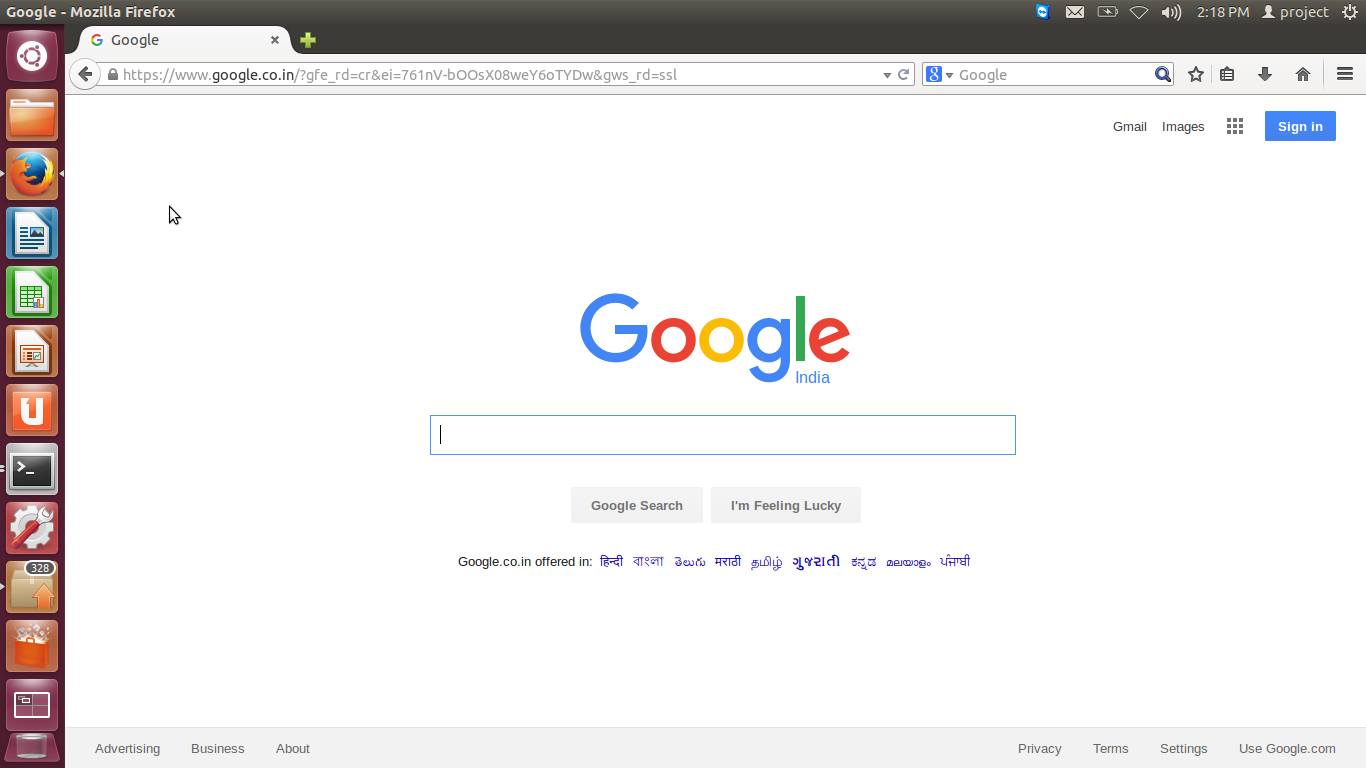


Figure 4: Internet Page

# Advanced - Remote Management

## Setup:

RPI box is connected to the LAN network and it gets erouter0 IP as 192.168.2.51. HDMI cable is connected to the TV and a USB keyboard is attached. Wired Client is connected through USB-ethernet adapter from RPI box and has IP 10.0.0.46.

erouter0 IP of RPI box is 192.168.2.51

eth0 IP of wired client is 10.0.0.46

## Steps:

* Remote Management allows the gateway to be remotely accessed by a Customer account representative to perform troubleshooting or maintenance
* Remote Management can be used via HTTP and HTTPS
* In Web UI, enable the HTTP option and enter the value for HTTP Port, then you can access your device from HTTP. For example, if the erouter0 IP address is 192.168.2.51 and the HTTP port number is 8080, then you would use http://192.168.2.51:8080 in an external machine to access
* It is the same way to configure HTTPS
* Select whether you would like to have Remote Management open to all Internet IP Addresses, an Internet IP Address range, or a single Internet IP Address
* Click Advanced -> Select Remote Management -> To Enable HTTP option: Click Enable Button -> Enter the value for HTTP Port -> Click Single Computer -> Enter IPV4 Address (Ex: 192.168.2.19)
* Both HTTP and HTTPs ports can also be enabled at the same time

# Advanced - DMZ

## Setup:

Follow the same setup as mentioned in the Section 8.1

## Steps:

* DMZ allows single computer on the LAN to open all its ports
* Connected client is running ‘dropbear’ service on port 22
* On external machine, do ssh <username of connected\_client>@erouter0 ip
* In WebUI, click on Advanced -> Select DMZ -> Click Enable button and enter the Ipv4 address of the connected client where the ports have to be opened and click on save button
* On external machine, do ssh <username of connected\_client>@erouter0 ip – ssh should get through successfully

# Troubleshooting – Diagnostic Tools

## Setup:

Follow the same setup as mentioned in the Section 8.1

## Steps:

* This feature enables the user to troubleshoot the issues related to network connectivity
* Path in WebUI, Troubleshooting --> Diagnostic Tools
* Test Connectivity Results: Checks your connectivity to the Internet. Count field shows the number of Packets sent and received
* Check IPv4 Address Results: Identifies accessibility to specific IP addresses. Key in 192.168.2.19 and it shows that Connectivity is Ok
* Traceroute Results: Displays the route of packets across an Internet Protocol (IP) network. Key in 10.0.0.46 (IP of connected client) and the trace route table will be displayed

# BRIDGE MODE

## Setup:

Follow the same setup as mentioned in the Section 8.1

## Steps:

* By default, RPI box will be Router mode
* To enable Bridge mode, in WebUI, Go to Gateway -> At A Glance -> Bridge Mode should be Enabled. This may take few minutes
* Once BridgeMode is ON, the below functionalities are disabled –
* Private Wifi should be OFF
* Ethernet connected client should get public IP address (in erouter series)
* Local\_ip\_configuration, DMZ, Wireless connected clients should be disabled
* Xfinity\_WIFI should be ON
* After reboot also, RPI box should come up in Bridge Mode if it was in Bridge Mode before reboot
* In external machine, we should have access to WebUI using eth0 IP if Remote Management is enabled

# Xfinity - Wifi

## Setup:

Follow the same setup as mentioned in the Section 8.1

## Steps:

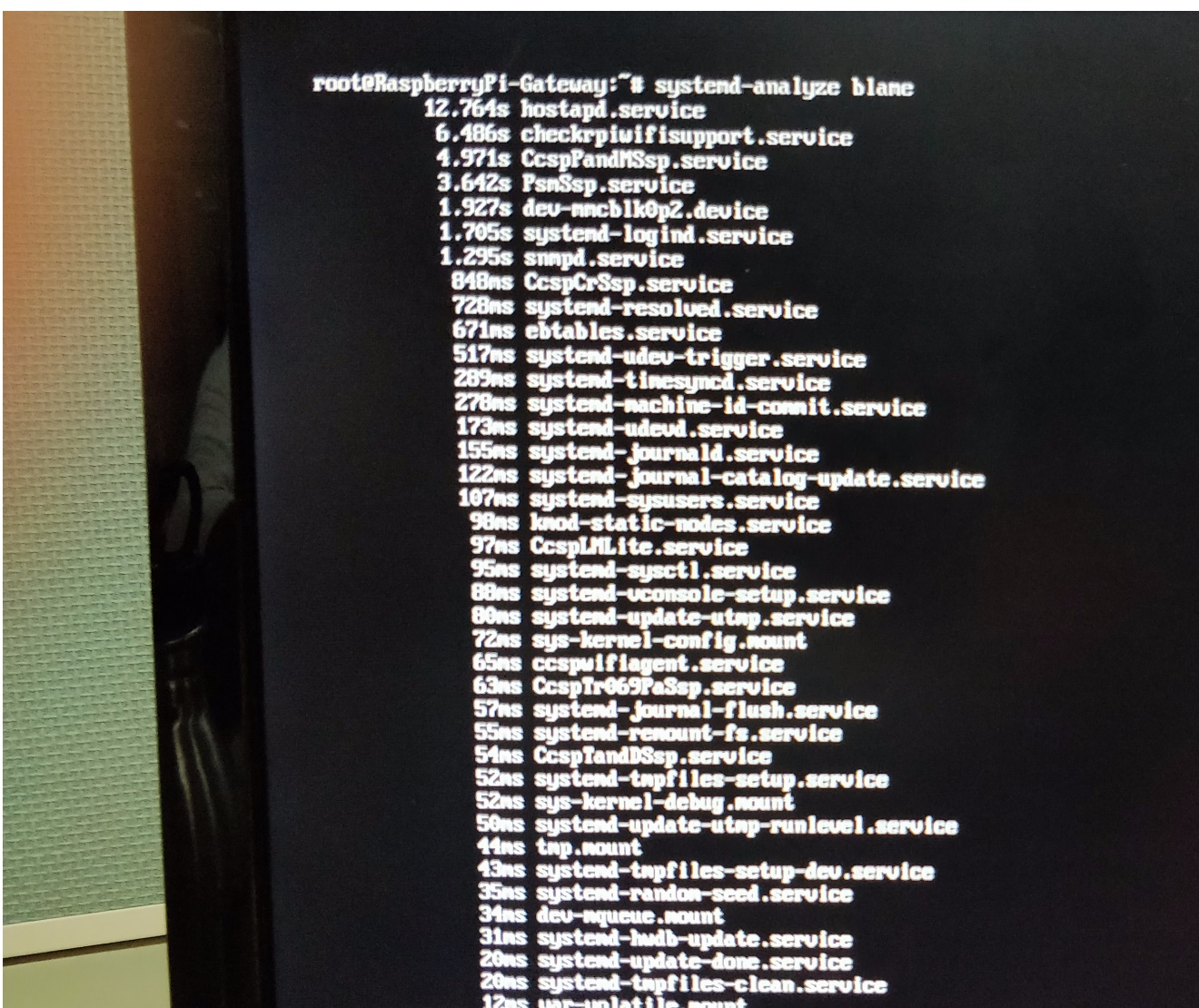
* Xfinity-Wi-Fi is a feature through which customer can connect to internet (not Home Device) using their Comcast account from anywhere. For example, suppose we are in a restaurant which has Xfinity with hotspot enabled, that Restaurant Xfinity Box will broadcast Xfinity Wi-Fi SSID. We can connect our Mobile/Device to that Xfinity SSID but once we connect, it will ask our Comcast account details and verify the same. Once verified, we can browse the internet with our account
* By default, Xfinity Wifi will be enabled in RPI box
* In mobile, connect to broadcasted 2.4GHz SSID ‘RPi3\_RDKB-AP4’. Xfinity wifi login page should be displayed
* For 5GHz, connect to SSID ‘Rpi3\_RDKB-AP5’. Xfinity Wifi login page should be displayed
* Check the IP address in mobile. It should be 172.20.20.20 (allocated from remote DHCP server)
* Enter the credentials (user name and password) for Xfinity Wifi
* Open browser and browse. Internet connectivity should be working

# Boot Time Measurement

List of commands to validate the Boot Time data:

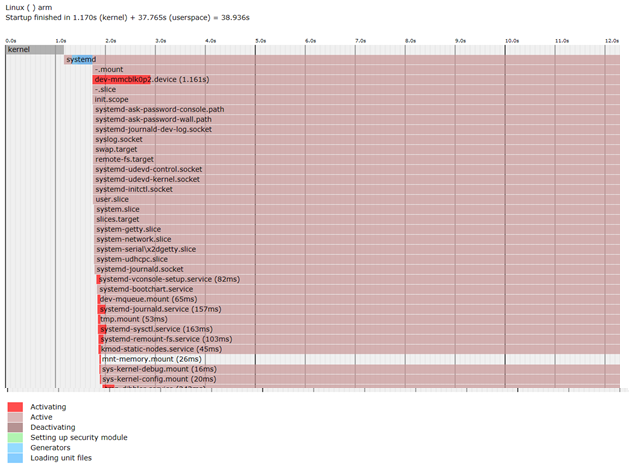
# systemd-analyze

# systemd-analyze blame - This command prints a list of all running units, ordered by the time they took to initialize



# systemd-analyze critical-chain - This command prints a tree of the time-critical chain of units. The time after the unit is active or started is printed after the "@" character. The time the unit takes to start is printed after the "+" character

# systemd-analyze plot > plot.svg - This command prints an SVG graphic detailing which system services have been started at what time, highlighting the time they spent on initialization



# systemd-bootchart - By default,systemd-bootchart.service is disabled. In order to start the process,execute the following command:

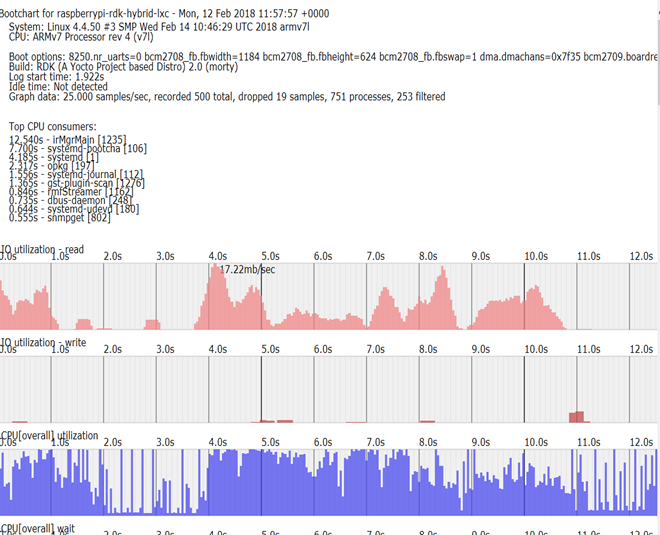
systemctl enable systemd-bootchart

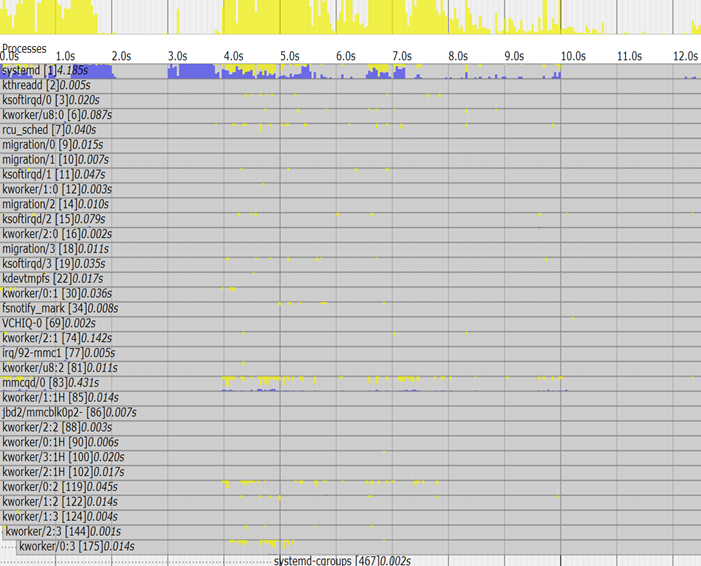
systemctl start systemd-bootchart

Systemd-bootchart command collects the CPU utilization, disk load, memory usage, as well as per-process information from a running system.

Collected results are output as an SVG graph at /run/log

Systemd-bootchart is available only in morty version (yocto 2.2)





# WebPA Client SupporT

Websocket based protocol hence bi-directional asynchronous communication is possible between gateway and WEBPA Server in real-time. This is similar to SNMP-PA and TR069-PA.

Cloud PA will be implemented as a RDK-B component which would register with RDKB-CR (Component Registry). Server PA would act like an interface between Cloud UI and RDK-B stack for faster real-time communication. Persistent websocket connection is established between Cloud PA and Server PA during the client boot up. This would eliminate initial connection handshake overhead but would have an impact on the scalability aspects on the server side.

Data communication and messaging format will be based on JSON format.

**Curl Commands**:

Below data needs to be furnished in the Curl Get & Set commands

URL: CommunityWebpaserverurl

MAC: 080027D7C5A7 (eth0 MAC of Emulator/erouter0 MAC of RPI)

SECRET TOKEN: Needs to be generated on daily basis

**Sample Get Requests in Console to get RadioNumberofEntries:**

* Curl -X GET

'URL/api/v2/device/mac:DEVICE MAC/config?names=Device.WiFi.RadioNumberOfEntries' -H 'authorization: Bearer SECRET TOKEN'

**Output in Console:**

{"parameters”: [{"name":"Device.WiFi.RadioNumberOfEntries","value":"2","dataType":2,"parameterCount":1,"message":"Success"}],"statusCode":200}

* Curl -X GET 'URL/api/v2/device/mac:DEVICE MAC/config?names=Device.WiFi.SSID.10001.SSID' -H 'authorization: Bearer SECRET TOKEN'

**Output in Console:**

{"parameters":[{"name":"Device.WiFi.SSID.10001.SSID","value":"RDKB\_EMU-2.4G","dataType":0,"parameterCount":1,"message":"Success"}],"statusCode":200}

**Sample Set Request in Console to set Wifi SSID for Privatewifi 2.4Ghz:**

* curl -X PATCH URL/api/v2/device/mac:DEVICE MAC/config -d '{"parameters": [ {"dataType": 0, "name": "Device.WiFi.SSID.10001.SSID", "value": "Testing"}]}' -H 'authorization: Bearer SECRETTOKEN'

**Output in Console:**

{"parameters":[{"name":"Device.WiFi.SSID.10001.SSID","message":"Success"}],"statusCode":200}

* curl -X PATCH URL/api/v2/device/mac:DEVICE MAC/config -d '{"parameters": [ {"dataType": 3, "name": "Device.WiFi.Radio.10000.X\_CISCO\_COM\_ApplySetting", "value": "true"}]}' -H 'authorization: Bearer SECRET TOKEN'

**Output in Console:**

{"parameters":[{"name":"Device.WiFi.Radio.10000.X\_CISCO\_COM\_ApplySetting","message":"Success"}],"statusCode":200}

**Sample Success & Failure Parameters Response in WebPA Server:**

{

          "parameters": [

            {

              "name": "Device.X\_COM",

              "value": "Low",

                 “status”:”Invalid Parameter Name”

            },

            {

              "name": "Device.Radio.2.Status",

              "value": "Enable",

                 “status”:”Read only parameter”

            },

            {

              "name": "Device.XHOSTS.X\_CISCO\_COM\_ConnectedDeviceNumber",

              "value": "0",

             “status”:”Success”

            },

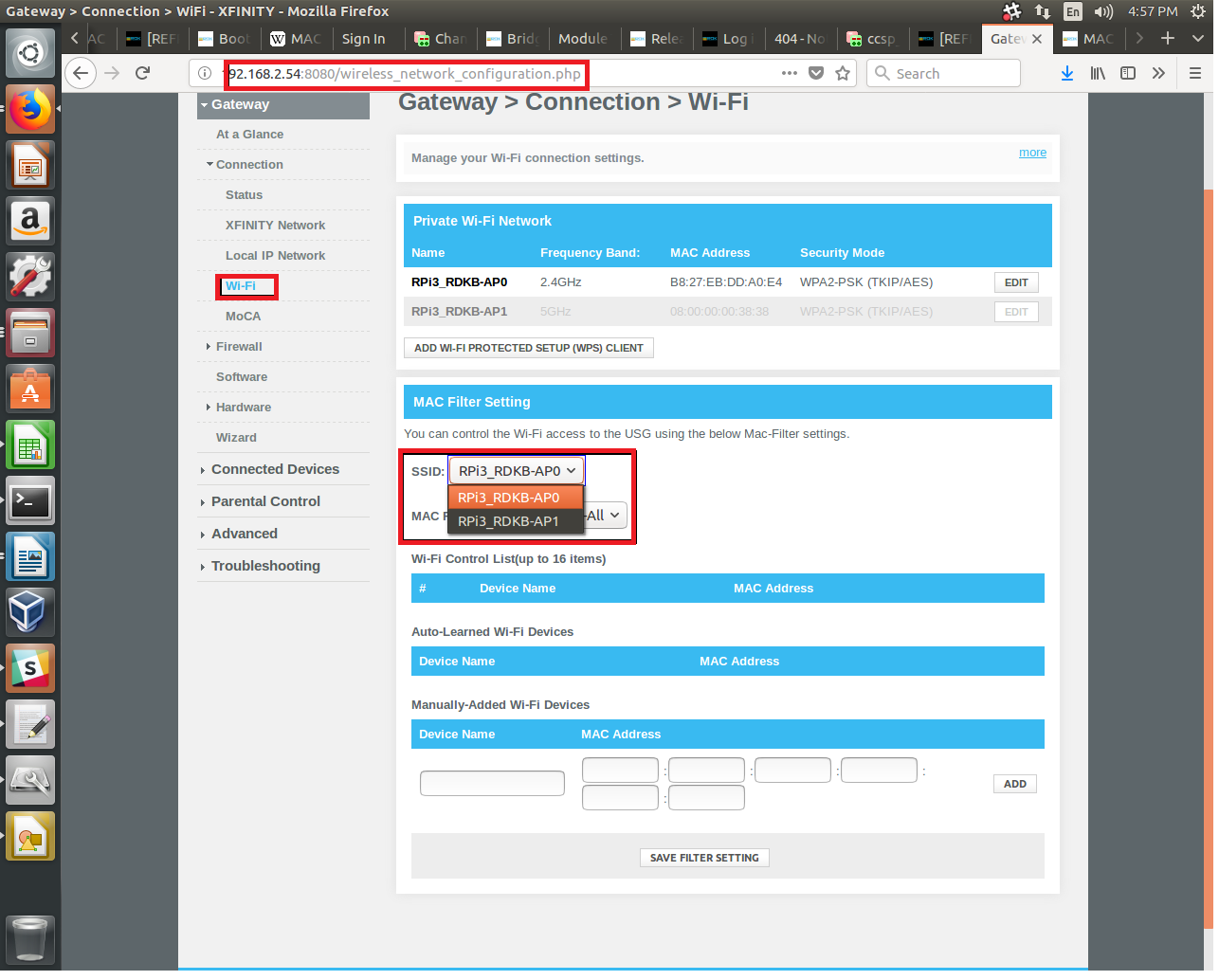
          ],

}

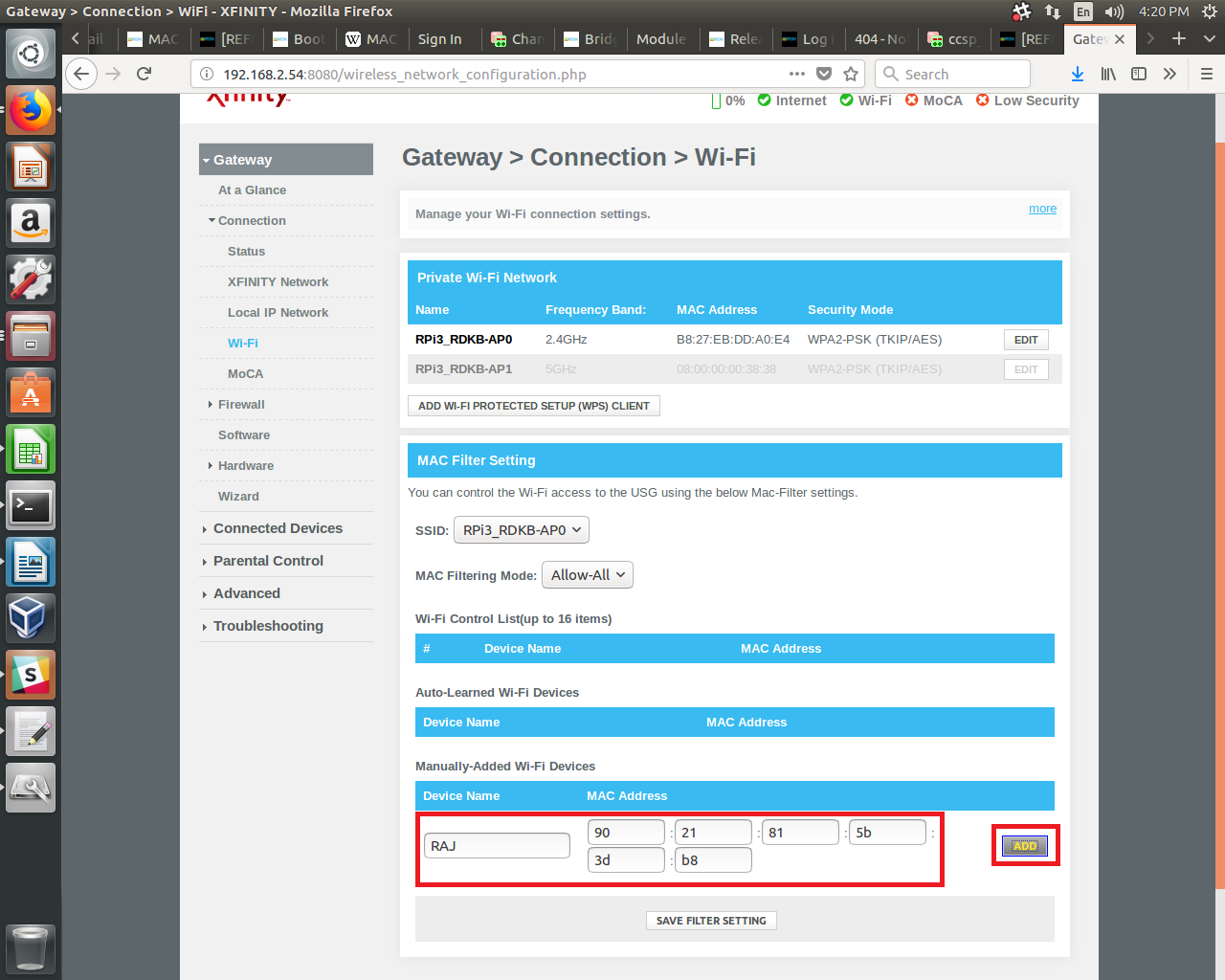
# MAC FILTERING

**Steps with Screenshots:**

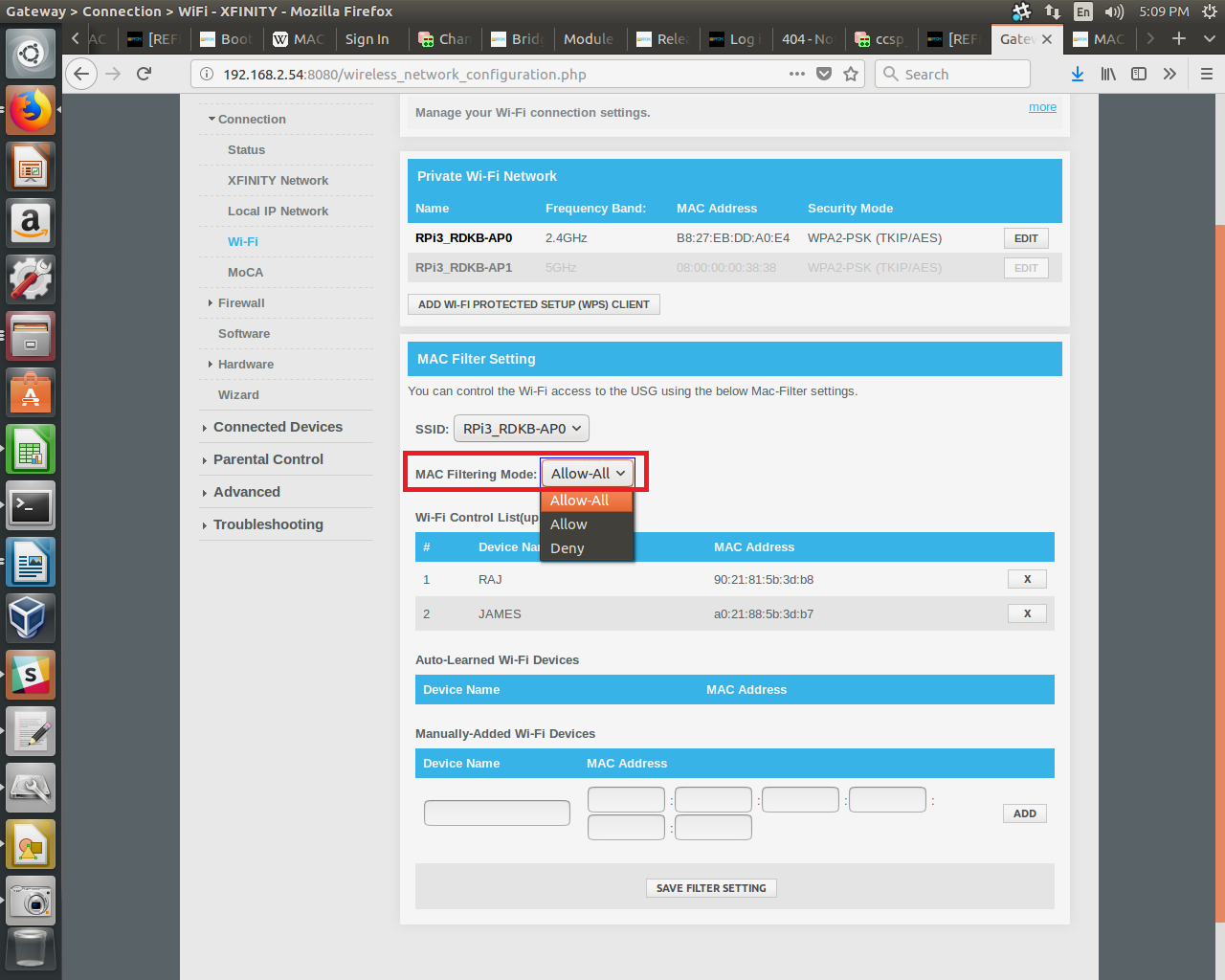
1. Select the **SSID** you want to apply MAC Filtering on



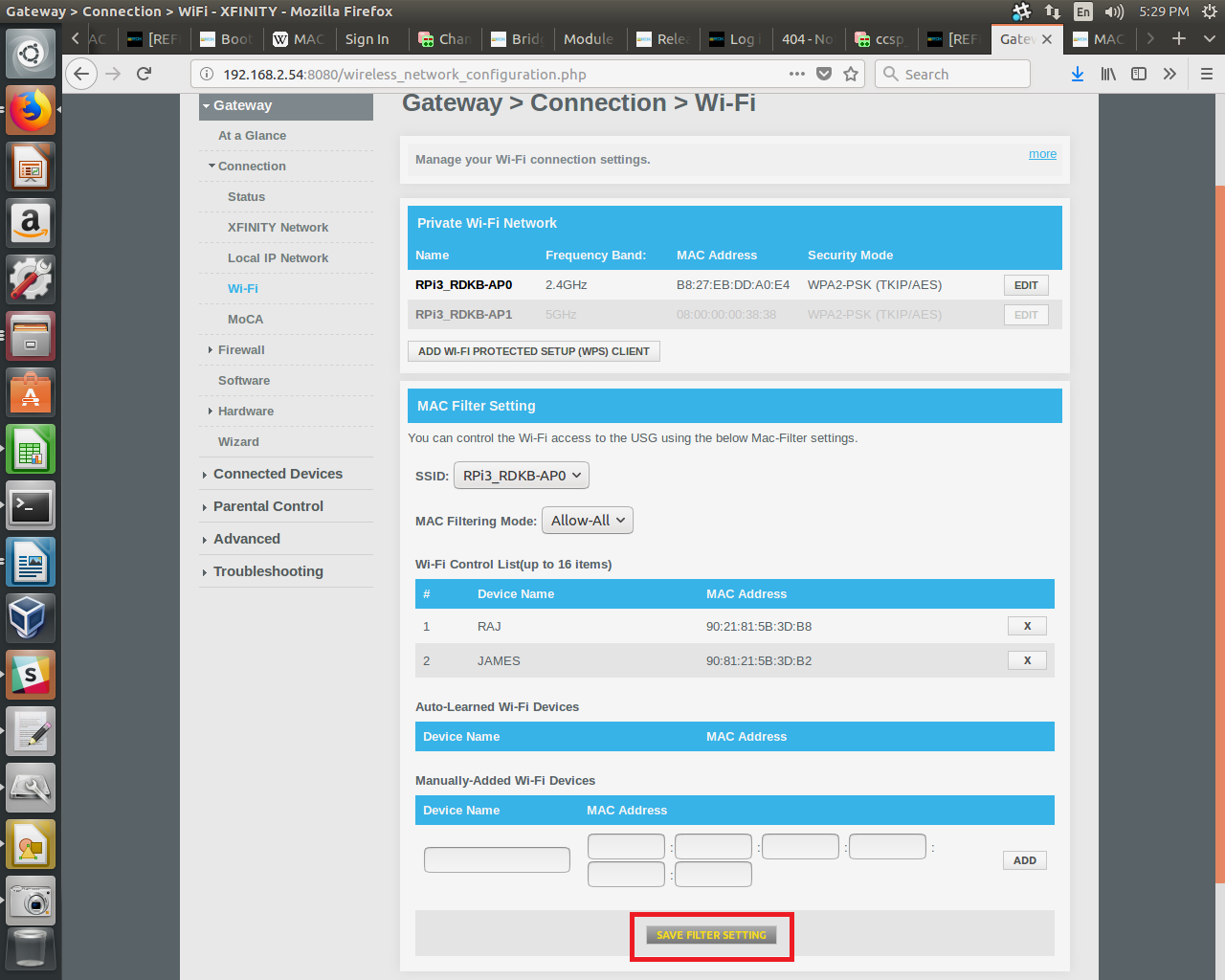
1. Add the Mac Address into **ACL** (Access Control List). After Entering MAC and Device Name Click on **ADD** button



1. Set the Mode - ALLOW, DENY, ALLOW-ALL to be applied on ACL



1. Click on SAVE FILTER SETTINGS

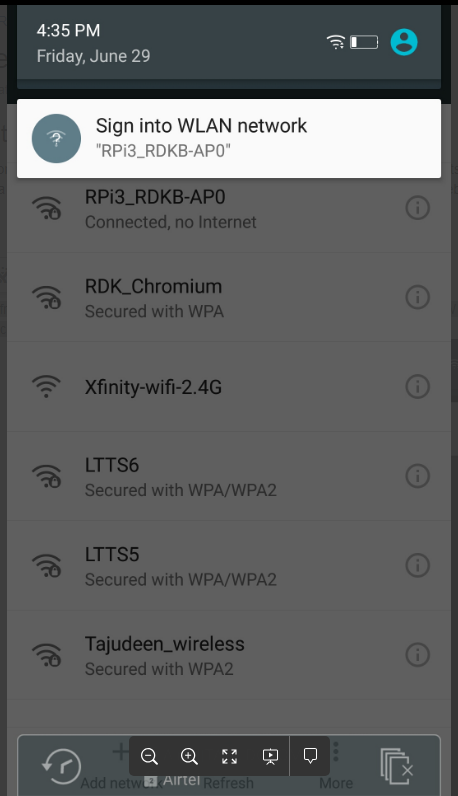


# CAPTIVE PORTAL

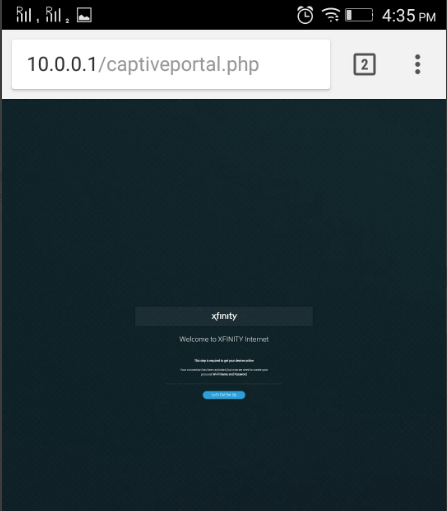
Captive portal provides user to configure their Wi-Fi SSID, credentials. Also redirects to Captive portal page on fresh boot up and factory reset in such a way it should not have impact on the remote management access of webUI.

**User Testing Process:**

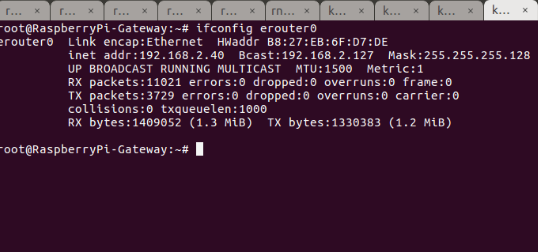
* During fresh bootup or a factory reset, the user should associate with R-Pi G/W network Wi-Fi Access point broadcasting in client mobile
* Wi-Fi Access point broadcasting in client mobile

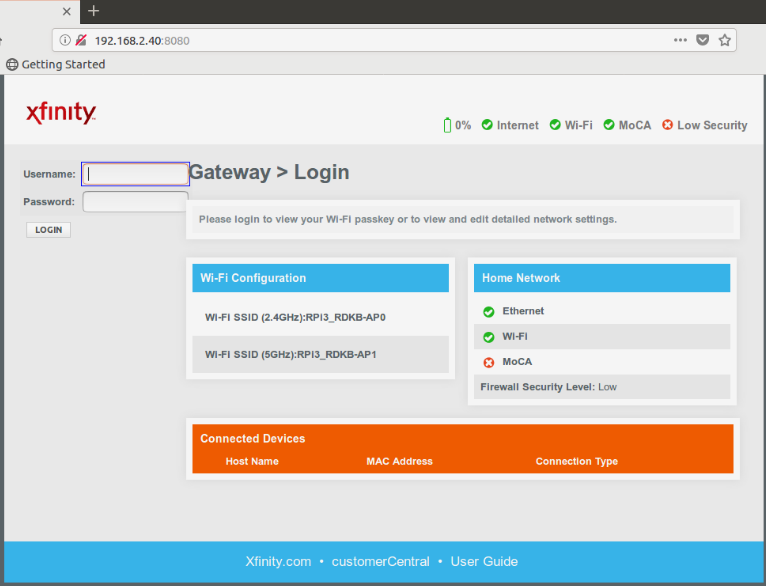


* Then in the client pc/ mobile, If the user tries to access any website url or webui then the browser will have the captive portal page redirected as shown below
* Captive Portal page in client mobile



* Remote management access:





* Enabling and Disabling the captive portal using dmcli command on fresh bootup

root@RaspberryPi-Gateway:~# dmcli eRT getv Device.DeviceInfo.X\_RDKCENTRAL-COM\_CaptivePortalEnable

**Console Output:**

CR component name is: eRT.com.cisco.spvtg.ccsp.CR

subsystem\_prefix eRT.

getv from/to component(eRT.com.cisco.spvtg.ccsp.pam): Device.DeviceInfo.X\_RDKCENTRAL-COM\_CaptivePortalEnable

Execution succeed.

Parameter    1 name: Device.DeviceInfo.X\_RDKCENTRAL-COM\_CaptivePortalEnable

              type:       bool,    value: true

root@RaspberryPi-Gateway:~# dmcli eRT setv Device.DeviceInfo.X\_RDKCENTRAL-COM\_CaptivePortalEnable bool false

**Console Output:**

CR component name is: eRT.com.cisco.spvtg.ccsp.CR

subsystem\_prefix eRT.

setv from/to component(eRT.com.cisco.spvtg.ccsp.pam): Device.DeviceInfo.X\_RDKCENTRAL-COM\_CaptivePortalEnable

Execution succeed.

root@RaspberryPi-Gateway:~# dmcli eRT getv Device.DeviceInfo.X\_RDKCENTRAL-COM\_CaptivePortalEnable

**Console Output:**

CR component name is: eRT.com.cisco.spvtg.ccsp.CR

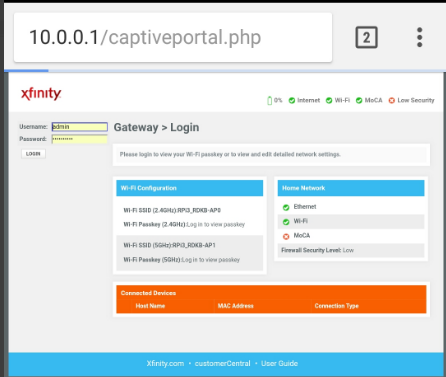
subsystem\_prefix eRT.

getv from/to component(eRT.com.cisco.spvtg.ccsp.pam): Device.DeviceInfo.X\_RDKCENTRAL-COM\_CaptivePortalEnable

Execution succeeds.

Parameter    1 name: Device.DeviceInfo.X\_RDKCENTRAL-COM\_CaptivePortalEnable

              type:       bool,   value: false



# Wireless Protection Setup PORTAL

WPS stands for Wi-Fi Protected Setup. It is a wireless network security standard that tries to make connections between a router and wireless devices faster and easier. WPS works only for wireless networks that use a password that is encrypted with the WPA Personal or WPA2 Personal security protocols. WPS doesn't work on wireless networks that are using the deprecated WEP security, which can be cracked easily by any hacker with a basic set of tools and skills. Wi-Fi Users don't want to know the broadcasting SSID and password.

By default, WPS Feature was enabled in RPI as well as Emulator.

<screen1>

Wifi Users will try to the network through WPS (PIN/PUSH)

* **PUSH BUTTON Method:**

In Client Devices, please go to Wifi settings then select the Advanced options/More Preferences. In that Advanced Options Setting, please click the WPS Push Button

<screen2>

If you want to connect the network through PUSH BUTTON Method, please select the Push Button from Connection Options and then click the pair button in WEBUI

<screen3-screen5>

* **PIN Method:**

In Client Devices, please go to Wifi settings then select the Advanced options/More Preferences. In that Advanced Options Setting, please click the WPS PIN Entry

<screen-6>

If you want to connect the Wifi network through PIN Method. Please select the PIN Entry from Connection Options then enter the Wireless Client's PIN (It’s generates from client devices) and then click the pair button in WebUI.

<screen7-10>

* **Disable WPS PIN Method**

If you want to disable the WPS Pin Entry Method, please click the disable button in WPS Pin Method in WebUI.

<screen11>

* **Disable WPS Feature**

If you want to disable the WPS Feature, please select the disable button in Wi-Fi Protected Setup(WPS) in WebUI

<screen12>