网络设定

cat /etc/rc.local

iptables-restore < /etc/iptables.ipv4.nat

# Generated by iptables-save v1.4.21 on Wed Jul 5 18:44:19 2017

\*filter

:INPUT ACCEPT [66:4636]

:FORWARD ACCEPT [0:0]

:OUTPUT ACCEPT [53:4692]

-A FORWARD -i eth0 -o wlan0 -m state --state RELATED

-A FORWARD -i eth0 -o wlan0 -m state --state RELATED,ESTABLISHED -j ACCEPT

-A FORWARD -i wlan0 -o eth0 -j ACCEPT

COMMIT

# Completed on Wed Jul 5 18:44:19 2017

# Generated by iptables-save v1.4.21 on Wed Jul 5 18:44:19 2017

\*nat

:PREROUTING ACCEPT [2:736]

:INPUT ACCEPT [2:736]

:OUTPUT ACCEPT [18:1196]

:POSTROUTING ACCEPT [8:480]

-A POSTROUTING -o eth0 -j MASQUERADE

COMMIT

# Completed on Wed Jul 5 18:44:19 2017

sudo iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE

sudo iptables -A FORWARD -i eth0 -o wlan0 -j ACCEPT

sudo cat /etc/network/interfaces

# interfaces(5) file used by ifup(8) and ifdown(8)

# Please note that this file is written to be used with dhcpcd

# For static IP, consult /etc/dhcpcd.conf and 'man dhcpcd.conf'

# Include files from /etc/network/interfaces.d:

source-directory /etc/network/interfaces.d

auto lo

iface lo inet loopback

auto eth0

iface eth0 inet static

address 192.168.1.2

netmask 255.255.255.0

network 192.168.1.0

broadcast 192.168.8.255

gateway 192.168.1.1

#allow-hotplug wlan0

#iface wlan0 inet static

# address 192.168.8.102

# netmask 255.255.255.0

# wpa-conf /etc/wpa\_supplicant/wpa\_supplicant.conf

allow-hotplug wlan0

iface wlan0 inet static

address 172.24.1.1

netmask 255.255.255.0

network 172.24.1.0

broadcast 172.24.1.255

allow-hotplug wlan1

iface wlan1 inet manual

wpa-conf /etc/wpa\_supplicant/wpa\_supplicant.conf

cat /etc/dnsmasq.conf

interface=wlan0 # Use interface wlan0

listen-address=172.24.1.1

bind-interfaces # Bind to the interface to make sure we aren't sending things elsewhere

server=8.8.8.8 # Forward DNS requests to Google DNS

domain-needed # Don't forward short names

bogus-priv # Never forward addresses in the non-routed address spaces.

dhcp-range=172.24.1.50,172.24.1.150,12h # Assign IP addresses between 172.24.1.50 and 172.24.1.150 with a 12 hour lease time

cat /etc/network/interfaces

# interfaces(5) file used by ifup(8) and ifdown(8)

# Please note that this file is written to be used with dhcpcd

# For static IP, consult /etc/dhcpcd.conf and 'man dhcpcd.conf'

# Include files from /etc/network/interfaces.d:

source-directory /etc/network/interfaces.d

auto lo

iface lo inet loopback

auto eth0

iface eth0 inet static

address 192.168.1.1

netmask 255.255.255.0

network 192.168.1.0

broadcast 192.168.1.255

gateway 192.168.1.1

post-up route del default dev eth0

allow-hotplug wlan0

iface wlan0 inet manual

wpa-conf /etc/wpa\_supplicant/wpa\_supplicant.conf

allow-hotplug wlan1

iface wlan1 inet manual

wpa-conf /etc/wpa\_supplicant/wpa\_supplicant.conf

:

https://frillip.com/using-your-raspberry-pi-3-as-a-wifi-access-point-with-hostapd/

:

https://wireless.wiki.kernel.org/en/users/Documentation/hostapd

重启服务

sudo systemctl status hostapd -l

sudo systemctl restart hostapd

sudo systemctl stop hostapd

sudo /home/pi/hostapd-2.6/hostapd/hostapd /etc/hostapd/hostapd.conf

sudo vim /etc/hostapd/hostapd.conf

interface=wlan0

driver=nl80211

ctrl\_interface=/var/run/hostapd

ctrl\_interface\_group=0

ssid=AirPi

hw\_mode=g

channel=3

#ap\_max\_inactivity=300

#disassoc\_low\_ack=0

wpa=2

wpa\_passphrase=superSecretPW

wpa\_key\_mgmt=WPA-PSK

wpa\_pairwise=TKIP CCMP

rsn\_pairwise=CCMP

auth\_algs=1

wmm\_enabled=1

beacon\_int=100

logger\_syslog=-1

logger\_syslog\_level=1

===============

# This is the name of the WiFi interface we configured above

interface=wlan0

# Use the nl80211 driver with the brcmfmac driver

driver=nl80211

# This is the name of the network

ssid=banana2

# Use the 2.4GHz band

hw\_mode=g

# Use channel 6

channel=3

# Enable 802.11n

# Enable WMM

# Enable 40MHz channels with 20ns guard interval

# Accept all MAC addresses

macaddr\_acl=0

# Use WPA authentication

auth\_algs=1

# Require clients to know the network name

ignore\_broadcast\_ssid=0

# Use WPA2

wpa=2

# Use a pre-shared key

wpa\_key\_mgmt=WPA-PSK

# The network passphrase

wpa\_passphrase=12345678

# Use AES, instead of TKIP

rsn\_pairwise=CCMP

~

===========================

sudo vim /etc/hostapd/hostapd.conf

interface=wlan0

# Use the nl80211 driver with the brcmfmac driver

driver=nl80211

# This is the name of the network

ssid=banana2

# Use the 2.4GHz band

hw\_mode=g

# Use channel 6

channel=6

# Enable WMM

wmm\_enabled=0

# Enable 40MHz channels with 20ns guard interval

# Accept all MAC addresses

macaddr\_acl=0

# Use WPA authentication

auth\_algs=1

# Require clients to know the network name

ignore\_broadcast\_ssid=0

# Use WPA2

wpa=2

# Use a pre-shared key

wpa\_key\_mgmt=WPA-PSK

# The network passphrase

wpa\_passphrase=12345678

# Use AES, instead of TKIP

rsn\_pairwise=CCMP

ap\_max\_inactivity=300

disassoc\_low\_ack=0

wpa\_group\_rekey=1800

post-up重启

==========================

cat /etc/network/interfaces

# interfaces(5) file used by ifup(8) and ifdown(8)

# Please note that this file is written to be used with dhcpcd

# For static IP, consult /etc/dhcpcd.conf and 'man dhcpcd.conf'

# Include files from /etc/network/interfaces.d:

source-directory /etc/network/interfaces.d

auto lo

iface lo inet loopback

auto eth0

iface eth0 inet static

address 192.168.1.2

netmask 255.255.255.0

network 192.168.1.0

broadcast 192.168.8.255

gateway 192.168.1.1

#allow-hotplug wlan0

#iface wlan0 inet static

# address 192.168.8.102

# netmask 255.255.255.0

# wpa-conf /etc/wpa\_supplicant/wpa\_supplicant.conf

allow-hotplug wlan0

iface wlan0 inet static

address 172.24.1.1

netmask 255.255.255.0

network 172.24.1.0

broadcast 172.24.1.255

post-up /home/pi/hostapd-2.6/hostapd/hostapd /etc/hostapd/hostapd.conf -B

allow-hotplug wlan1

iface wlan1 inet manual

wpa-conf /etc/wpa\_supplicant/wpa\_supplicant.conf

换了新版

编译的配置文件:=====================================================================

cat .config

# Example hostapd build time configuration

#

# This file lists the configuration options that are used when building the

# hostapd binary. All lines starting with # are ignored. Configuration option

# lines must be commented out complete, if they are not to be included, i.e.,

# just setting VARIABLE=n is not disabling that variable.

#

# This file is included in Makefile, so variables like CFLAGS and LIBS can also

# be modified from here. In most cass, these lines should use += in order not

# to override previous values of the variables.

# Driver interface for Host AP driver

CONFIG\_DRIVER\_HOSTAP=y

# Driver interface for wired authenticator

#CONFIG\_DRIVER\_WIRED=y

# Driver interface for drivers using the nl80211 kernel interface

CONFIG\_DRIVER\_NL80211=y

# QCA vendor extensions to nl80211

#CONFIG\_DRIVER\_NL80211\_QCA=y

# driver\_nl80211.c requires libnl. If you are compiling it yourself

# you may need to point hostapd to your version of libnl.

#

#CFLAGS += -I$<path to libnl include files>

#LIBS += -L$<path to libnl library files>

# Use libnl v2.0 (or 3.0) libraries.

#CONFIG\_LIBNL20=y

# Use libnl 3.2 libraries (if this is selected, CONFIG\_LIBNL20 is ignored)

#CONFIG\_LIBNL32=y

# Driver interface for FreeBSD net80211 layer (e.g., Atheros driver)

#CONFIG\_DRIVER\_BSD=y

#CFLAGS += -I/usr/local/include

#LIBS += -L/usr/local/lib

#LIBS\_p += -L/usr/local/lib

#LIBS\_c += -L/usr/local/lib

# Driver interface for no driver (e.g., RADIUS server only)

#CONFIG\_DRIVER\_NONE=y

# IEEE 802.11F/IAPP

CONFIG\_IAPP=y

# WPA2/IEEE 802.11i RSN pre-authentication

CONFIG\_RSN\_PREAUTH=y

# PeerKey handshake for Station to Station Link (IEEE 802.11e DLS)

CONFIG\_PEERKEY=y

# IEEE 802.11w (management frame protection)

CONFIG\_IEEE80211W=y

# Integrated EAP server

CONFIG\_EAP=y

# EAP Re-authentication Protocol (ERP) in integrated EAP server

CONFIG\_ERP=y

# EAP-MD5 for the integrated EAP server

CONFIG\_EAP\_MD5=y

# EAP-TLS for the integrated EAP server

CONFIG\_EAP\_TLS=y

# EAP-MSCHAPv2 for the integrated EAP server

CONFIG\_EAP\_MSCHAPV2=y

# EAP-PEAP for the integrated EAP server

CONFIG\_EAP\_PEAP=y

# EAP-GTC for the integrated EAP server

CONFIG\_EAP\_GTC=y

# EAP-TTLS for the integrated EAP server

CONFIG\_EAP\_TTLS=y

# EAP-SIM for the integrated EAP server

#CONFIG\_EAP\_SIM=y

# EAP-AKA for the integrated EAP server

#CONFIG\_EAP\_AKA=y

# EAP-AKA' for the integrated EAP server

# This requires CONFIG\_EAP\_AKA to be enabled, too.

#CONFIG\_EAP\_AKA\_PRIME=y

# EAP-PAX for the integrated EAP server

#CONFIG\_EAP\_PAX=y

# EAP-PSK for the integrated EAP server (this is \_not\_ needed for WPA-PSK)

#CONFIG\_EAP\_PSK=y

# EAP-pwd for the integrated EAP server (secure authentication with a password)

#CONFIG\_EAP\_PWD=y

# EAP-SAKE for the integrated EAP server

#CONFIG\_EAP\_SAKE=y

# EAP-GPSK for the integrated EAP server

#CONFIG\_EAP\_GPSK=y

# Include support for optional SHA256 cipher suite in EAP-GPSK

#CONFIG\_EAP\_GPSK\_SHA256=y

# EAP-FAST for the integrated EAP server

# Note: If OpenSSL is used as the TLS library, OpenSSL 1.0 or newer is needed

# for EAP-FAST support. Older OpenSSL releases would need to be patched, e.g.,

# with openssl-0.9.8x-tls-extensions.patch, to add the needed functions.

#CONFIG\_EAP\_FAST=y

# Wi-Fi Protected Setup (WPS)

#CONFIG\_WPS=y

# Enable UPnP support for external WPS Registrars

#CONFIG\_WPS\_UPNP=y

# Enable WPS support with NFC config method

#CONFIG\_WPS\_NFC=y

# EAP-IKEv2

#CONFIG\_EAP\_IKEV2=y

# Trusted Network Connect (EAP-TNC)

#CONFIG\_EAP\_TNC=y

# EAP-EKE for the integrated EAP server

#CONFIG\_EAP\_EKE=y

# PKCS#12 (PFX) support (used to read private key and certificate file from

# a file that usually has extension .p12 or .pfx)

CONFIG\_PKCS12=y

# RADIUS authentication server. This provides access to the integrated EAP

# server from external hosts using RADIUS.

#CONFIG\_RADIUS\_SERVER=y

# Build IPv6 support for RADIUS operations

CONFIG\_IPV6=y

# IEEE Std 802.11r-2008 (Fast BSS Transition)

#CONFIG\_IEEE80211R=y

# Use the hostapd's IEEE 802.11 authentication (ACL), but without

# the IEEE 802.11 Management capability (e.g., FreeBSD/net80211)

#CONFIG\_DRIVER\_RADIUS\_ACL=y

# IEEE 802.11n (High Throughput) support

CONFIG\_IEEE80211N=y

# Wireless Network Management (IEEE Std 802.11v-2011)

# Note: This is experimental and not complete implementation.

#CONFIG\_WNM=y

# IEEE 802.11ac (Very High Throughput) support

#CONFIG\_IEEE80211AC=y

# Remove debugging code that is printing out debug messages to stdout.

# This can be used to reduce the size of the hostapd considerably if debugging

# code is not needed.

#CONFIG\_NO\_STDOUT\_DEBUG=y

# Add support for writing debug log to a file: -f /tmp/hostapd.log

# Disabled by default.

#CONFIG\_DEBUG\_FILE=y

# Add support for sending all debug messages (regardless of debug verbosity)

# to the Linux kernel tracing facility. This helps debug the entire stack by

# making it easy to record everything happening from the driver up into the

# same file, e.g., using trace-cmd.

#CONFIG\_DEBUG\_LINUX\_TRACING=y

# Remove support for RADIUS accounting

#CONFIG\_NO\_ACCOUNTING=y

# Remove support for RADIUS

#CONFIG\_NO\_RADIUS=y

# Remove support for VLANs

#CONFIG\_NO\_VLAN=y

# Enable support for fully dynamic VLANs. This enables hostapd to

# automatically create bridge and VLAN interfaces if necessary.

#CONFIG\_FULL\_DYNAMIC\_VLAN=y

# Use netlink-based kernel API for VLAN operations instead of ioctl()

# Note: This requires libnl 3.1 or newer.

#CONFIG\_VLAN\_NETLINK=y

# Remove support for dumping internal state through control interface commands

# This can be used to reduce binary size at the cost of disabling a debugging

# option.

#CONFIG\_NO\_DUMP\_STATE=y

# Enable tracing code for developer debugging

# This tracks use of memory allocations and other registrations and reports

# incorrect use with a backtrace of call (or allocation) location.

#CONFIG\_WPA\_TRACE=y

# For BSD, comment out these.

#LIBS += -lexecinfo

#LIBS\_p += -lexecinfo

#LIBS\_c += -lexecinfo

# Use libbfd to get more details for developer debugging

# This enables use of libbfd to get more detailed symbols for the backtraces

# generated by CONFIG\_WPA\_TRACE=y.

#CONFIG\_WPA\_TRACE\_BFD=y

# For BSD, comment out these.

#LIBS += -lbfd -liberty -lz

#LIBS\_p += -lbfd -liberty -lz

#LIBS\_c += -lbfd -liberty -lz

# hostapd depends on strong random number generation being available from the

# operating system. os\_get\_random() function is used to fetch random data when

# needed, e.g., for key generation. On Linux and BSD systems, this works by

# reading /dev/urandom. It should be noted that the OS entropy pool needs to be

# properly initialized before hostapd is started. This is important especially

# on embedded devices that do not have a hardware random number generator and

# may by default start up with minimal entropy available for random number

# generation.

#

# As a safety net, hostapd is by default trying to internally collect

# additional entropy for generating random data to mix in with the data

# fetched from the OS. This by itself is not considered to be very strong, but

# it may help in cases where the system pool is not initialized properly.

# However, it is very strongly recommended that the system pool is initialized

# with enough entropy either by using hardware assisted random number

# generator or by storing state over device reboots.

#

# hostapd can be configured to maintain its own entropy store over restarts to

# enhance random number generation. This is not perfect, but it is much more

# secure than using the same sequence of random numbers after every reboot.

# This can be enabled with -e<entropy file> command line option. The specified

# file needs to be readable and writable by hostapd.

#

# If the os\_get\_random() is known to provide strong random data (e.g., on

# Linux/BSD, the board in question is known to have reliable source of random

# data from /dev/urandom), the internal hostapd random pool can be disabled.

# This will save some in binary size and CPU use. However, this should only be

# considered for builds that are known to be used on devices that meet the

# requirements described above.

#CONFIG\_NO\_RANDOM\_POOL=y

# Should we use poll instead of select? Select is used by default.

#CONFIG\_ELOOP\_POLL=y

# Should we use epoll instead of select? Select is used by default.

#CONFIG\_ELOOP\_EPOLL=y

# Should we use kqueue instead of select? Select is used by default.

#CONFIG\_ELOOP\_KQUEUE=y

# Select TLS implementation

# openssl = OpenSSL (default)

# gnutls = GnuTLS

# internal = Internal TLSv1 implementation (experimental)

# none = Empty template

#CONFIG\_TLS=openssl

# TLS-based EAP methods require at least TLS v1.0. Newer version of TLS (v1.1)

# can be enabled to get a stronger construction of messages when block ciphers

# are used.

#CONFIG\_TLSV11=y

# TLS-based EAP methods require at least TLS v1.0. Newer version of TLS (v1.2)

# can be enabled to enable use of stronger crypto algorithms.

#CONFIG\_TLSV12=y

# If CONFIG\_TLS=internal is used, additional library and include paths are

# needed for LibTomMath. Alternatively, an integrated, minimal version of

# LibTomMath can be used. See beginning of libtommath.c for details on benefits

# and drawbacks of this option.

#CONFIG\_INTERNAL\_LIBTOMMATH=y

#ifndef CONFIG\_INTERNAL\_LIBTOMMATH

#LTM\_PATH=/usr/src/libtommath-0.39

#CFLAGS += -I$(LTM\_PATH)

#LIBS += -L$(LTM\_PATH)

#LIBS\_p += -L$(LTM\_PATH)

#endif

# At the cost of about 4 kB of additional binary size, the internal LibTomMath

# can be configured to include faster routines for exptmod, sqr, and div to

# speed up DH and RSA calculation considerably

#CONFIG\_INTERNAL\_LIBTOMMATH\_FAST=y

# Interworking (IEEE 802.11u)

# This can be used to enable functionality to improve interworking with

# external networks.

#CONFIG\_INTERWORKING=y

# Hotspot 2.0

#CONFIG\_HS20=y

# Enable SQLite database support in hlr\_auc\_gw, EAP-SIM DB, and eap\_user\_file

#CONFIG\_SQLITE=y

# Enable Fast Session Transfer (FST)

#CONFIG\_FST=y

# Enable CLI commands for FST testing

#CONFIG\_FST\_TEST=y

# Testing options

# This can be used to enable some testing options (see also the example

# configuration file) that are really useful only for testing clients that

# connect to this hostapd. These options allow, for example, to drop a

# certain percentage of probe requests or auth/(re)assoc frames.

#

#CONFIG\_TESTING\_OPTIONS=y

# Automatic Channel Selection

# This will allow hostapd to pick the channel automatically when channel is set

# to "acs\_survey" or "0". Eventually, other ACS algorithms can be added in

# similar way.

#

# Automatic selection is currently only done through initialization, later on

# we hope to do background checks to keep us moving to more ideal channels as

# time goes by. ACS is currently only supported through the nl80211 driver and

# your driver must have survey dump capability that is filled by the driver

# during scanning.

#

# You can customize the ACS survey algorithm with the hostapd.conf variable

# acs\_num\_scans.

#

# Supported ACS drivers:

# \* ath9k

# \* ath5k

# \* ath10k

#

# For more details refer to:

# http://wireless.kernel.org/en/users/Documentation/acs

#

#CONFIG\_ACS=y

# Multiband Operation support

# These extentions facilitate efficient use of multiple frequency bands

# available to the AP and the devices that may associate with it.

#CONFIG\_MBO=y

# Client Taxonomy

# Has the AP retain the Probe Request and (Re)Association Request frames from

# a client, from which a signature can be produced which can identify the model

# of client device like "Nexus 6P" or "iPhone 5s".

#CONFIG\_TAXONOMY=y

重

重

重

重