1.编译ubuntu内核**(没用)**

(1)执行命令uname -a查看ubuntu当前的版本

Linux gaorui-virtual-machine 3.5.0-23-generic

(2)到

http://packages.ubuntu.com/precise/linux-headers-3.5.0-23-generic

下载ubuntu 3.5.0的内核源代码

(3)解压源代码  tar -xzvf 源代码压缩包

(4)在/usr/src下创建文件夹kernel

mkdir /usr/src/kernel

将源代码的整个文件拷贝到/usr/src/kernel

cp -r 源文件目录/usr/src/kernel

(5)进入该目录/usr/src/kernel/linux-3.5，

将ubuntu本身原有的配置文件拷贝过来

cp /boot/config-3.5.0-23-generic .config

(6)执行命令 make oldconfig，编译原有内核模式，

过程中的选择，全选 y

(7)执行命令make，同时编译内核及内核模块

(8)编译结束后，生成bzlmage文件，位于./arch/i386/boot

(9)执行命令 make modules\_install，安装编译好的内核模块

(10)此时，自动创建了/lib/modules/3.5.0目录，该目录存放编译好的内核文件.(该目录看一下就好，不需要进入)

(11)还在/usr/src/kernel/linux-3.5下，执行make install。内核编译结束

依赖该内核编译出来的iaodv并不能加载到ubuntu中，版本不对，后来发现 目录/usr/src/linux-headers-3.5.0-23-generic存放的就是ubuntu编译好的代码，直接用这个就好

2.编译iaodv(有用)

(1)在终端执行命令，uname -a查看内核版本以及为32位还是64位，确定内核版本后，去/usr/src找到对应的源码

(1)进入iaodv目录，打开makefile文件

添加ubuntu\_KDIR := /usr/src/linux-headers-3.5.0-23-generic

(/usr/src/linux-headers-3.5.0-23-generic为ubuntu内核版本的源 码，源码版本一定要与ubuntu的内核版本一致)

添加 $(MAKE) -C $(ubuntu\_KDIR) SUBDIRS=$(PWD) obj-m=fbaodv\_ubuntu.o ARCH=x86 modules

rm \*.o

rm \*.mod.c

(2)进入终端，进入iaodv的目录下，执行

export ARCH=x86(内核为64位时，执行)

或export ARCH=i386(32位)

make

(3)执行完命令后，会看到fbaodv\_ubuntu.ko，这就是我们所需要的内核模块

3.加载fbaodv\_ubuntu.ko

(1)执行脚本aodv.sh，fbaodv\_ubuntu.ko与该脚本在同一个目录下

./aodv 192.168.5.11

(2)aodv脚本内容：

|  |
| --- |
| #!/bin/bash  usage(){  echo "usage: `basename $0` ip-addrs (aodv-blacklist) (dtn-blacklist)"  }  if [ $# -lt 1 ]; then  usage  exit 1  fi  IP=$1  aodvblacklist=""  dtnblacklist=""  if [ $2 ];then  aodvblacklist=$2  fi  if [ $3 ];then  dtnblacklist=$3  fi  del\_adhoc0=0  del\_wlan0=0  add\_adhoc0=0  adhoc0\_up=0  adhoc0\_join=0  stop network-manager  ifconfig adhoc0 $IP  if [ $? -eq 0 ];then  iw dev adhoc0 del  if [ $? -eq 0 ];then  del\_adhoc0=1  echo "adhoc0 del done!"  else  exit 1;  fi  fi  ifconfig wlan0 $IP  if [ $? -eq 0 ];then  iw dev wlan0 del  if [ $? -eq 0 ];then  del\_wlan0=1  echo "wlan0 del done!"  else  exit 1;  fi  fi  iw phy phy0 interface add adhoc0 type ibss  if [ $? -eq 0 ];then  add\_adhoc0=1  echo "iw phy phy0 interface add adhoc0 done!"  else  exit 1;  fi  if [ $add\_adhoc0 -eq 1 ];then  ifconfig adhoc0 up  if [ $? -eq 0 ];then  adhoc0\_up=1  echo "ifconfig adhoc0 up done!"  else  exit 1;  fi  fi  if [ $adhoc0\_up -eq 1 ];then  iw dev adhoc0 ibss join wujingbang 2412  if [ $? -eq 0 ];then  adhoc0\_join=1  echo "iw dev adhoc0 ibss join done!"  else  exit 1;  fi  fi  ifconfig adhoc0 $IP  ifconfig adhoc0  route add default gw 192.168.5.1  route -n  #fbaodv config  MESH\_DEV="mesh\_dev=adhoc0"  AODV\_NET="network\_ip=$IP/255.255.255.0"  GATEWAY="aodv\_gateway=0"  #METRIC="routing\_metric=ETT"  METRIC="routing\_metric=HOPS"  RATE="nominal\_rate=60"  COM\_RADIUS="com\_radius=700"  echo "1" > /proc/sys/net/ipv4/ip\_forward  echo "Running AODV-MCC"  #rmmod fbaodv\_ubuntu  #echo "clean log!"  #echo > /system/log  insmod ./fbaodv\_ubuntu.ko $MESH\_DEV $AODV\_NET $METRIC $RATE $GATEWAY $COM\_RADIUS aodv\_blacklist="$aodvblacklist" dtn\_blacklist="$dtnblacklist" |

3.一些细节：

(1)加载aodv后，使用命令dmesg > log.txt，查看是否加载aodv正常

因为当我在笔记本用自己的内置网卡设置的adhoc，aodv加载异 常。换用虚拟机下的ubuntu12.04和芯片为RT3070的usb无线网 卡即可成功加载AODV

(2)加载成功aodv后，aodv与模拟器交互正常，却ping不通。是因为 aodv发送给模拟器的数据流是由五个成员变量组成的，而模拟器 中只返回了四个，漏掉的那一个变量在aodv中是用来判断是否做 距离判断的，很重要。修改模拟器后，aodv正常使用

(3)即使可以相互ping通，DTN2中还是会出现Network is unreachble 的问题。

这是因为网关的原因

在控制台下执行

route add default gw 192.168.5.1

route -n

即可。

改命令已经放入脚本中

(4)在配好adhoc模式后，网络一会又自动变成wifi模式，

此时，需要在控制台下执行

stop network-manager

该命令已经放入脚本