Wireless Ad Hoc Networks
Lab 6

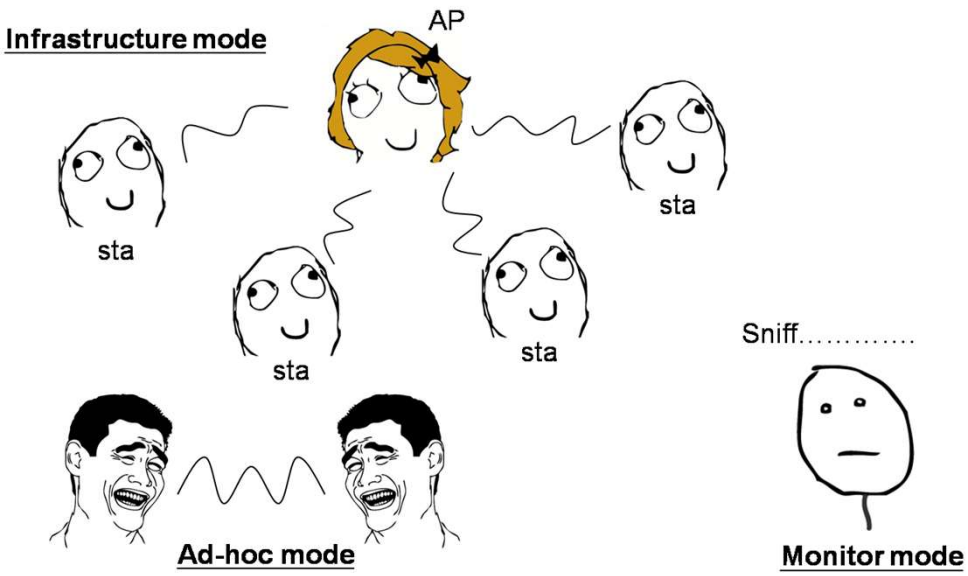
Ad Hoc Network

Wireless Operating Modes

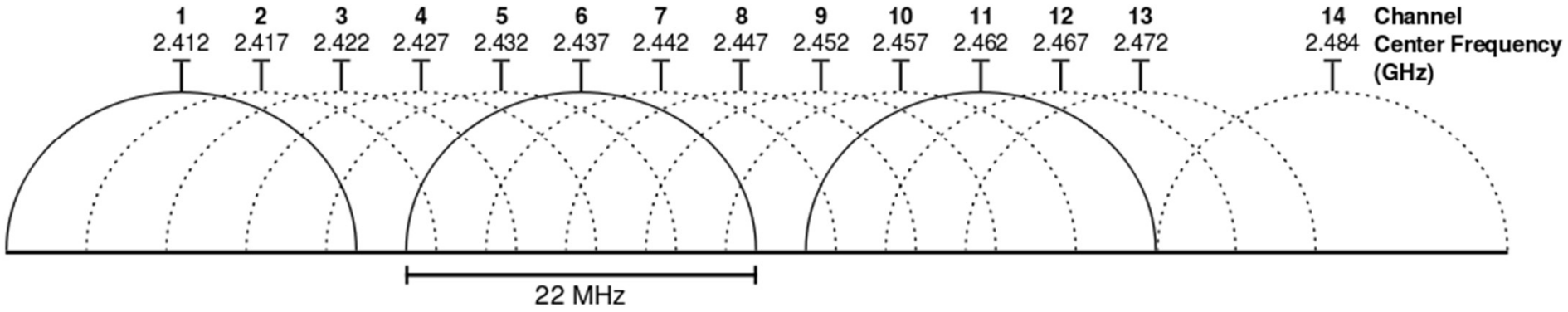
Wi-Fi modes of operation (802.11 or Wi-Fi)

- Station (STA) infrastructure mode
  - This mode is also called **“ Managed “**
- AccessPoint (AP) infrastructure mode
- Monitor (MON) mode (i.e, Sniff mode)

  Don’t need to connect any AP
- **Ad-Hoc (IBSS) mode**



Ad-Hoc mode – Wireless Channel



Before setting

先安裝下面兩個再開始實驗

sudo apt-get install traceroute

sudo apt-get install babeld

Wireless NIC on Ad-Hoc mode

- An **IBSS (Independent Basic Service Set)** network,
  - often called an **ad-hoc** network
  - a way to have a group of devices talk to each other wirelessly, without a central controller
  - All devices talk directly to each other, with no inherent relaying
- How create a new interface (Linux iw command)
  - sudo iw phy phy0 interface add adhoc0 **type ibss**
- How to join an adhoc network (Linux iwconfig command)
  - sudo iwconfig adhoc0 **mode ad-hoc** essid **bun-mesh-x** channel **x**

Set wireless NIC on Ad-Hoc mode (1)

Before set NIC to adhoc mode, remember to stop/disconnect all the wireless network connection

- turn off the interface
  - iw dev
  - sudo ifconfig wlan# down
  - sudo iw dev wlan# del
- create a new interface
  - sudo iw phy phy# interface add adhoc0 type ibss

```
adhoc@adhoc:~$ iw dev
phy#0
        Interface wlan8
                ifindex 3
                type managed
```

Set wireless NIC on Ad-Hoc mode (2)

- join the adhoc network
  - sudo ifconfig adhoc0 down (need to turn off before change mode!)
  - sudo iwconfig adhoc0 mode ad-hoc essid bun-mesh-x channel x
- configure the IP address
  - sudo ip -6 addr add FE80::42:42:xx/128 dev adhoc0 (for IPv6)
  - sudo ip addr add 192.168.10.xx/32 dev adhoc0 (for IPv4)
- turn on the interface
  - sudo ifconfig adhoc0 up
  - ifconfig (check if your ip addr is the same as your setting)
- check the wireless interface status
  - sudo iwconfig

```
adhoc0    IEEE 802.11bgn  ESSID:"bun-mesh-1"
          Mode:Ad-Hoc  Frequency:2.432 GHz  Cell: 0A:04:C6:36:94:B1
          Tx-Power=20 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Power Management:off
```

```
adhoc0    Link encap:Ethernet  HWaddr f4:f2:6d:17:74:bc
          inet6 addr: fe80::42:42:82/128 Scope:Link
          inet6 addr: fe80::f6f2:6dff:fe17:74bc/64 Scope:Link
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:430 errors:0 dropped:1 overruns:0 frame:0
          TX packets:292 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:50890 (50.8 KB)  TX bytes:41168 (41.1 KB)
```

Babel Routing Protocol (1)

- **Babel** – a loop-avoiding distance-vector routing protocol
  - based on ideas in DSDV, AODV, and Cisco's EIGRP
  - designed to work well not only in wired networks but also in wireless mesh networks
  - is in the process of becoming an IETF Standard

- Resources:
  - https://www.irif.fr/~jch/software/babel/
  - https://tools.ietf.org/html/rfc6126 (RFC Draft)
  - https://github.com/jech/babeld (Source Code)

Babel Routing Protocol (2)

- Loop-avoiding
  - Uses distributed Bellman-Ford
  - Feasibility condition guarantees good transient behavior
- Metrics
  - Hop-count on wired links
  - ETX (expected transmission count) on wireless links
  - Babel-Z3 (refines ETX by taking radio interference into account)
  - Babel-RTT (uses delay as component of routing metric)
- Route Selection
  - Choose route with smallest metric
  - Prefer stable routes

Babel Routing Protocol (3)

- Packets are sent in the body of a UDP datagram
- Hello Message
  - For Neighbor discovery
  - Broadcast periodically with a seqno (sequence number)
- IHU (I Heard You)
  - To confirm bidirectional reachability
  - Sent less often than Hellos
  - Carry the link's rxcost (reception cost)
- Route Request
  - Prompts receiver to send an update for a given prefix

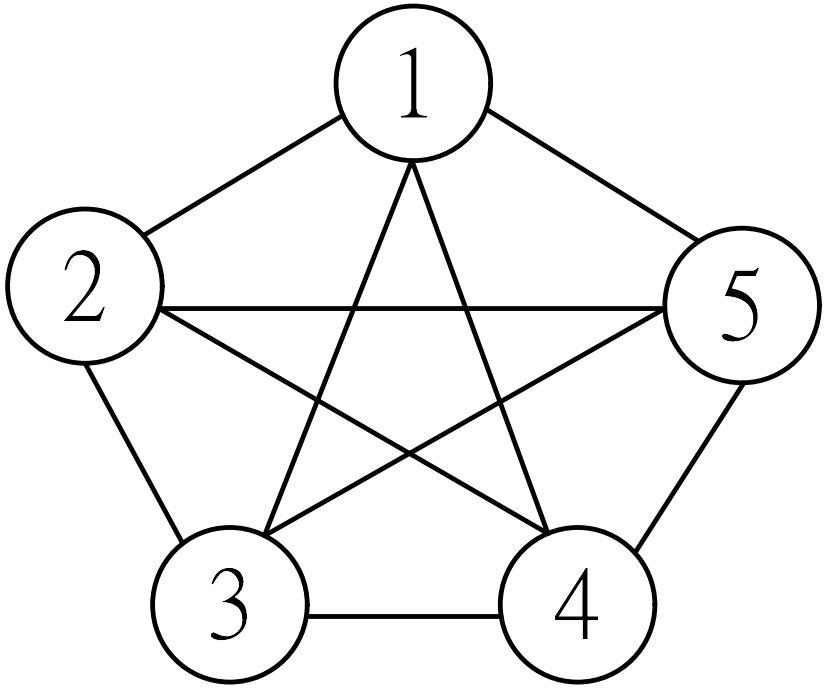
Install Babel

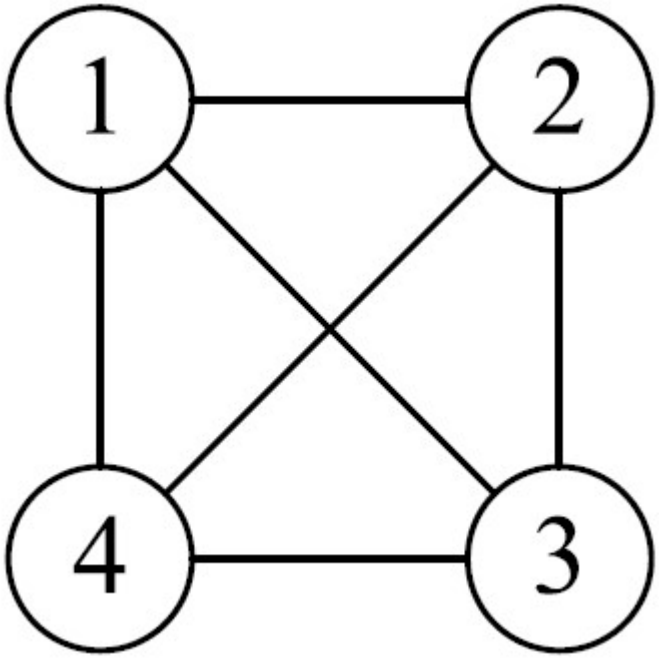
- From source
  - git clone git://github.com/jech/babeld.git
- Install directly from Ubuntu packages
  - sudo apt-get install babeld

- run babel routing daemon
  - sudo babeld adhoc0
- Or, run babel routing daemon with debugging
  - sudo babeld –g 33123 adhoc0
  - telnet ::1 33123 (in another window/tab to observe)

Create topology

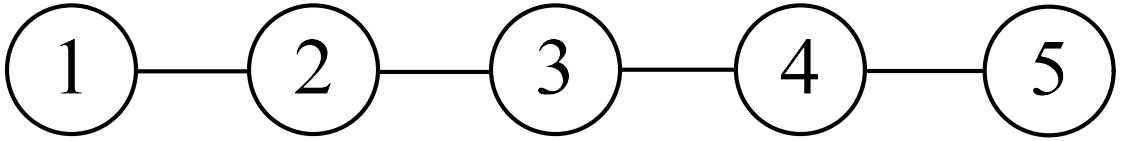
- When every node can connect to each other, the topology is a star-shaped topology.





Create topology

- How can we create the desirous topology in a small space?
  - (ex: lab, classroom, office)



Experiment Part 1

Part 1

1. set NIC to adhoc mode
2. open Wireshark
3. Run babeld
4. Observe the mesh network
5.結報放三種封包(Hello ,IHU, Route Request)截圖

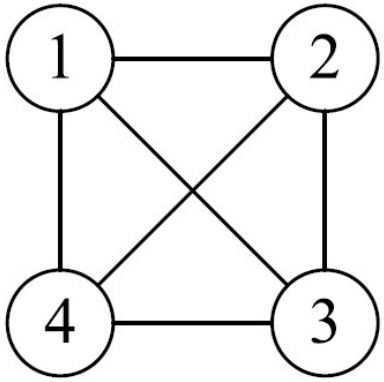
Hint :

Use wireshark to observe babel packets

Experiment Part 2

Part 2 四人一組

1. Change the topology
2. Observe the change
3. 結報參考範例



Hint :

Use traceroute to observe routing path

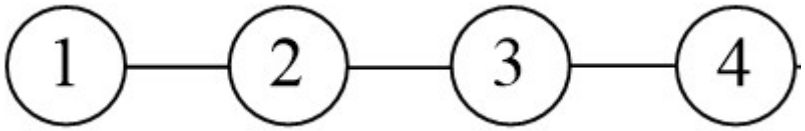
Use ping and ping6 to send packets to neighboring nodes

- ping6 -I adhoc0 fe80:42:42:XX
- sudo traceroute -i adhoc0 fe80::42:42:XX

Experiment Part 3 (optional)

Part 3 四或三人一組

1. Change the topology (set topology to a line)
2. Observe the change



Hint :

sudo ip6tables -t filter -A INPUT -s xx::xx:xx:xx -j DROP (set filter rule)

sudo ip6tables -L (list all filter rules)

sudo ip6tables -F (delete all filter rules)

Use ping & traceroute to observe the **multihop** traffic path