Infrastructure mode support for IEEE 802.11 implementation in NS-2

Ilango Purushothaman, Sumit Roy
{ilangop, sroy}@u.washington.edu
Department of EE, University of Washington,
Seattle, WA 98195-2500

The existing 802.11 implementation in ns-2.32 doesn't support infrastructure mode simulations. Beacon frame, Scanning, Authentication and Association functions have not been implemented.

This report outlines the changes which were made to the existing ns-2.32 802.11 implementation, to enable infrastructure mode support.

Stage 1 - Beacon Frames and Passive Scanning

- A node can be configured as an Access Point (AP) by the following command.
  $mac_(ap_node) **ap** [$mac_(ap_node) id]
  Once an AP is set up, it will start transmitting beacons.
- A new timer, **BeaconTimer**, was added to facilitate periodic transmission of beacons (set for BeaconInterval).
- The beacon packet is built in the **sendBEACON() function** and transmitted (depending on channel status) in **chk_pktBEACON()**. According to the IEEE 802.11 standard, AP should send the beacon at every expiry of BeaconInterval. If the medium is found to be busy, AP should defer using basic CSMA deferral procedure.
- Currently, Beacon frames have the following fields.
  - Address and BSSID information.
  - Timestamp
  - Beacon Interval
- Multiple APs can be setup and beacons can be successfully sent by all the APs and received. Beacons from different APs do not collide, due to CSMA/CA rules.
- The beacon reception function **recvBEACON()** is called, which basically stores the all the relevant information that can be obtained the beacon frame.

- **Passive Scanning** - If **ScanType** is configured as PASSIVE, the RSS (Received signal strength) of all beacons received within **ChannelTime** are stored and the "best AP", based on RSS measurements, is determined using **passive_scan()**. The list of APs and beacon powers is stored in a linked list.
- Now that passive scanning is over, STAs can now move on to authentication and association.

Stage 2 – Probe frames and Active Scanning

- **Active Scanning** – A STA with ScanType as ACTIVE, sends out a broadcast Probe Request, evoking all APs within range, to respond. The functions in charge of probe request transmission are, **sendPROBEREQ()** and **check_pktPROBEREQ().**

- All APs within range, receive the probe request using **recvPROBEREQ()** and schedule unicast Probe Responses, using **sendPROBEREP()** and c**heck_pktPROBEREP()**
- Probe Response, has essentially the same information as a Beacon frame.
- The probe response reception function **recvPROBEREP()** is called, which basically stores the all the information in the probe frame.
- **Probe Timer** – A new timer was created to handle active scanning. It can be run for two different time intervals.
    - Set to **MinChannelTime** – Time to declare a channel empty. If CCA busy is not indicated within this time, channel is declarer, devoid of APs.
    - Set to **MaxChannelTime** – Time to collect all Probe Responses. If CCA busy was indicated within MinChannelTime, Probe Timer is continued for MaxChannelTime.
- Once Probe Timer expires with MaxChannelTime, the STA processes all the probe responses using **active_scan()**, and selects the "strongest" AP based on RSS measurements.

Stage 3 - Authentication

- Once the scanning is over and the best AP has been determined, each STA can start the authentication process. This is accomplished by checking the **checkAuthAssocStatus()** function.
- Each STA sends out an Authentication frame, with sequence number 1, to its strongest AP, containing the specific BSSID. The function **sendAUTHENTICATE()** builds the Authentication frame and the actual transmission function is **check_pktAUTHENTICATE().**
- The AP, on receiving the authentication frame, using **recvAUTHENTICATE()**, AP respond with an authentication frame of sequence number 2, informing the client of the Authentication key. On reception of the second authentication frame, each STA sets its own "**authenticated**" flag and sends an acknowledgement to the AP.
- Currently, a two frame authentication exchange sequence, with **Open-key authentication** is provided.
- On successful authentication, the client sets the AP's MAC address as its **BSSID** and is ready for association.
- On receiving the ACK from the authenticating node, the AP sets the **authentication state** of the STA, in a linked list.

Stage 4 - Association

- Once authentication is completed, each STA starts the association process, by sending the Association Request frame, using **sendASSOCREQ()** and **check_pktASSOCREQ**().
- On receiving the Association request frame using **recvASSOCREQ()**, the AP sends out the Association response, containing the Association ID. The functions used are **sendASSOCREP() and check_pktASSOCREP()**.

- On reception of Association response frames, using **recvASSOCREP(),** each STA sets its own "**associated**" flag and sends an acknowledgement to the AP. It can now transfer data packets in the BSS.
- On receiving the ACK, the AP updates its client list, setting the **association state** of the STA, in question.
- On successful association, the STA is now eligible to transfer data in the BSS through the AP.
- **Packet filtering** is done, using this client state table. If a client doesn't belong to the BSS, the AP discards the packet

Inter-AP communication:

- Clients of the same BSS/different BSSs can exchange data, and the APs of the BSSs perform forwarding. Even if a connection (UDP, TCP etc) is setup between two clients of the same BSS, data will be forwarded through the APs only.
- Inter-AP communication can take place on a wired Distribution System or a Wireless distribution medium.
- Due to lack of multi-channel support and incomplete support for wired-cum-wireless scenarios, Inter-AP communication has been implemented on the same wireless medium, as used by the clients.
- **Multicast** - Data transfer across multiple BSSs is facilitated by the two participating APs, using a multicast method. This is accomplished by the **ToDs and FromDs** bits and use of a fourth address field (as per the IEEE 802.11 standard).

Mobility Support:

- A Mobile client can move from one BSS into another BSS, thereby initiating a handoff.
- Handoff Detection – Handoff is detected usually by a decrease in RSS below a handoff power threshold.
- Method Used – Three consecutive retransmissions indicate a mobile, moving out of range of an AP. The other reasons for dropped packets, - Collisions and fading are ruled out (Idea: Velayos et al. [2]).
- Once handoff is detected, Active Scanning is started and a new AP (if any) is selected and (re)association takes place.
- The new and the old APs update their client tables accordingly, after a successful handoff.

Limitations:

- Multiple BSSs can be set up in a single channel. Multiple BSSs can also be setup on different channels (using different channel objects) but communication across different channels is not possible for now. Due to lack of multiple channel support, a STA cannot scan all the channels used in the ESS. Hence, for now, scanning (active and passive) is limited to one channel only.
- Inter-AP communication, using a separate distribution medium, has not been implemented yet. Inter-AP communication, for now, is achieved by using the same

wireless channel used by the clients, but special address filtering makes sure that only APs can receive these “distribution frames”.

Example Script:

An example script **“infra.tcl”,** of two BSSs operating in a single channel is provided in **/tcl/ex/80211**. This sets up two APs in a single channel and STAs are configured with either Passive or Active ScanType. One STA is made to move from one BSS into another, triggering a handoff

The following can be observed from this scenario.

- ➢ Passive and Active Scanning
- ➢ Authentication and Association
- ➢ Due to mobility, handoff can be observed.
- ➢ Inter-AP forwarding of packets can be observed here.

User configurable TCL Hooks:

1. ScanType
2. BeaconInterval
3. ProbeDelay
4. MinChannelTime
5. MaxChannelTime
6. ChannelTime

References:

- IEEE Std. 802.11-1999, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, *IEEE Standard 802.11, 1999.*
- H. Velayos and G. Karlsson, “Techniques to Reduce IEEE 802.11b MAC Layer Handover Time,” Proceedings of *IEEE ICC, vol. 7, pp. 3844-3848, June 2004.*