CSC/ECE 570 – COMPUTER NETWORKS

FINAL OPNET PROJECT

Submission date: 10-December-2013

Submitted by:

Truptesh Malangi Nagesh
Unity ID: tmalang
Student ID#: 200021946

Bharath Venkatesh
Unity ID: bharat
Student ID: 001037220

1. Abstract

Wireless networks represent an emerging network architecture which has been actively studied and standardized for the last several years. However wireless networks are prone to errors in the physical media and the problem of hidden and exposed nodes in the network. In this project we evaluate the performance of one of the mechanism introduced to solve the hidden terminal problem - RTS/CTS, on 802.11n Wireless networks. We then streamline our study on how priority and packet size in combination can impact the throughput of the network. We further introduce multiple types of Jammer nodes into the network and evaluate how the network responds to these jamming attacks. Finally we modify the wlan_mac process model to simulate misbehavior of the participating nodes and study the resultant effect on throughput in comparison to the normal setup.

2. Introduction

The IEEE 802.11n standard offers several technical benefits over previous technology generations, which result in improved throughput to 802.11n-based clients, as well as greater reliability for legacy 802.11a/b/g clients. 802.11n based networks are now common place. With increasing use of mobile devices in the recent past, 802.11n enabled Wi-fi on mobile devices have gained a widespread usability in the mobile domain. Wireless LAN unlike Ethernet uses air as the channel for communication and this is a fading channel with interference and thus more prone to errors. Also in a wireless network, it is likely that a node at the far end of the access point's range can see the access point, but it cannot see a node on the opposite end of the access point's range. To counter the problem mentioned above, RTS/CTS mechanism was introduced. Our first objective is to measure its effectiveness in 802.11n Networks.

Request to Send/ Clear to Send

Consider a scenario as shown in Figure 1. Station B wants to send data to Station C. The medium appears to be the medium appears used only for the stations located within the range of station B. Other stations cannot sense a signal and consider the medium idle. Therefore station D might start a transmission since it does not notice B’s ongoing transmission. However since the receiving station C is within range of B and D and thus receives two signals at once it will not be able to clearly understand either of the two transmission (whether destined for it or not). This collision however cannot be detected at the sending station B unless it notices the lacking acknowledgment from station C after a certain time-out.

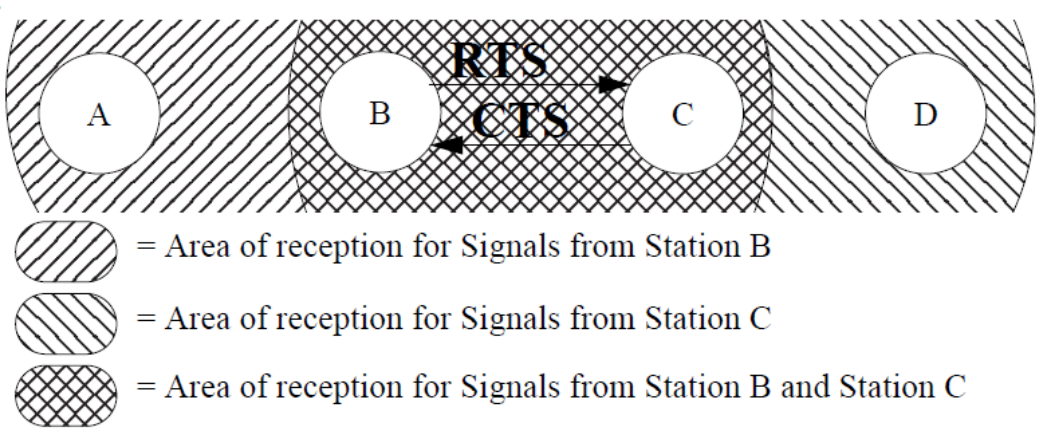


Figure 1: Hidden Node Problem in Wireless Networks

To solve this problem the RTS/CTS mechanism was introduced. With RTS/CTS, Each station seeking access to the medium selects a random slot within the contention window. The station with the earliest slot now has the permission to start transmitting. Stations with a later slot hear the other station transmitting and refrain from sending. When the RTS/CTS mechanism is applied, the station does not send data packets right away but sends a RTS packet to the receiving station that responds with a CTS packet. If a station captures a RTS packet from another station and it is not the destination of the RTS packet it reads the intended transmission duration from the RTS packet and stays silent for that time. The same happens if only a CTS packet is received by a station outside of the transmission range of the sender but within the range of the receiver. This guarantees that all stations within range of either sender or receiver have knowledge of the transmission and its duration.

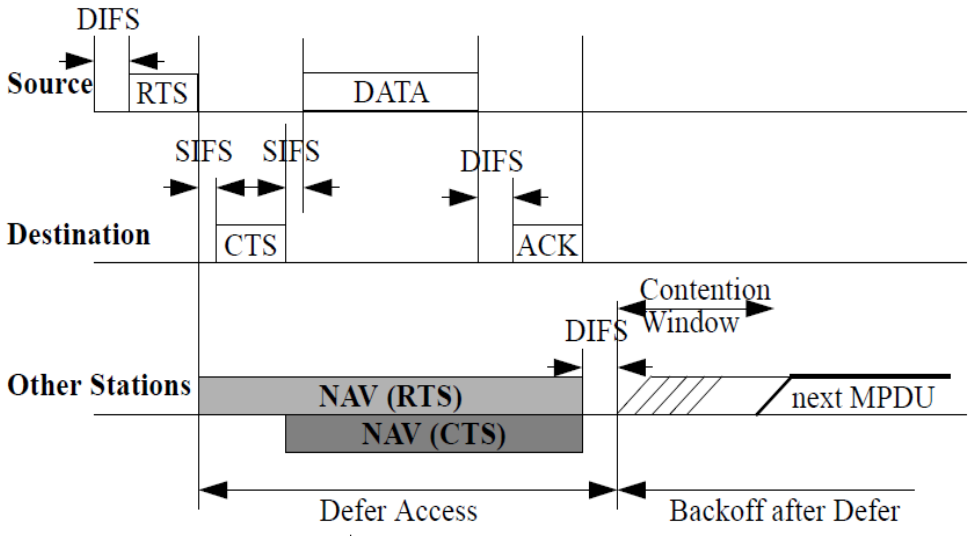


Figure 2: DCF mechanism

Priority in 802.11n Networks

To support more efficient QoS in 802.11n networks the concept of EDCF was introduced. In EDCF Packets are assigned to one of four different access categories (AC); they are Background, Best Effort, Video, and Voice. This mechanism works by manipulating the idle time that exists between frames (i.e., the inter-frame spaces and the collision window (CW)). The arbitration inter-frame space (AIFS), whose length varies depending on the priority of the frame being transmitted, replaces the distributed inter frame space (DIFS). Higher priority frames have a smaller AIFS and CW than do lower priority frames.
The AIFS for each of the traffic is as below

Type of Traffic	AIFS
Background	7
Best Effort	3
Video	2
Voice	2

Jammers in 802.11n Networks

Security is a major concern when deploying any Wireless networks. Jamming is one of most widespread attacks in recent days especially in military base environments. Jamming is any attack to deny service to legitimate users by generating noise or fake protocol packets or legitimate packets but with spurious timing. The objectives of this project is to simulate a 802.11n network with single band and pulse jammers; launch jamming attacks in order to test how much influence different jammers have in WLAN communications.

Drawbacks of Conventional Jamming and Means of Intelligent Jamming

The most trivial way of disrupting a wireless network is by generating a continuous high power noise across the entire bandwidth near the transmitting and receiving nodes. But these kinds of jamming attacks are very easily detectable and are power consuming. An interesting aspect of this would be, if instead of a dedicated jammer in the network, a random participating node tries to misbehave and gain undue advantage. This would cause detrimental results on the network. Our final objective is to simulate a misbehaving network and analyze its impact on throughput.

3. Methodology

Since our aim was to study a network wherein there is a fair distribution of all types of traffic we chose a base scenario that reflects a fair mix of all four access categories.

- Background Traffic : 16.67% - 2 nodes
- Best effort traffic : 33.33% - 4 nodes
- Voice Traffic – 25% - 3 nodes
- Interactive Multimedia – 25% - 3 nodes

OPNET modeler was used to simulate and analyze four scenarios.

- The first scenario was simulated based on the wireless LAN models supported by OPNET modeler library to test the efficiency of RTS/CTS in improving the performance of the network wherein hidden nodes are present. By varying the power of the nodes in the network we were able to simulate the hidden terminal scenario explained above. Further RTS/CTS was enabled on all the nodes to evaluate its impact on the performance. This is the base scenario on which multiple variations described in scenarios 2 to 4 are built.
- The second scenario was simulated to analyze the ramifications of jammers in 802.11n networks and the difference in results for multiple jammers used. Here we first introduce a single band jammer on the same channel that all the nodes are transmitting on and observe its impact. Further we replace the single band jammer with a pulse band jammer. By varying the pulse width of the jammer we were able to observe varying throughputs.
- The third scenario was simulated with an aim to observe how packet size and various access category priorities in combination can impact the performance of 802.11n networks. In this scenario we aim to study priority inversion as one of the means to jam the network. For this we decreased the inter-arrival time of the Background traffic with a considerable reduction in packet size. It is to be noted that Background traffic has the least priority. We measure its impact on the throughput of the network.
- The fourth scenario aims at simulating misbehaviors in the network. Every node in the network follows an underlying MAC protocol that defines the SIFS, DIFS and a backoff mechanism. However an interesting observation can be made when nodes try to misbehave by deciding not to backoff. Since there exists no inbuilt feature that employs such a feature we modified the wireless lan mac node model code to induce the effect of zero backoff in the network. Observations are explained in detail in the forthcoming section.

Node Characteristics/Model Specification

The first scenario consisted of 12 nodes each of type wlan_station_adv with an Access Point in the center. All nodes are arranged in a circular fashion as shown in the figure below. Mobile nodes 9 and 10 generate background traffic, Nodes 11, 12 1 and 2 represent best effort service, nodes 3,4 and 5 belong to interactive multimedia and the rest nodes 6,7 and 8 generate interactive voice traffic. This represents the base scenario.

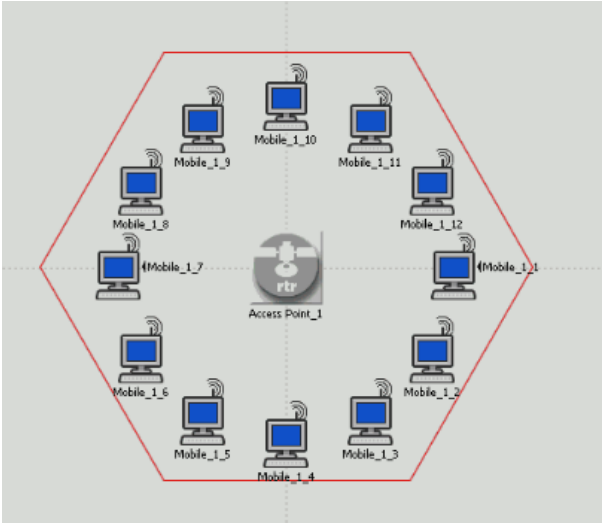


Figure3: Base scenario

Scenario 2 uses a single band and pulse jammer to disrupt the network. We make sure to place the jammer close to the Access point with sufficient power to disrupt communication in the network as shown in the figure below. Rest of the network remains the same as in the base scenario.

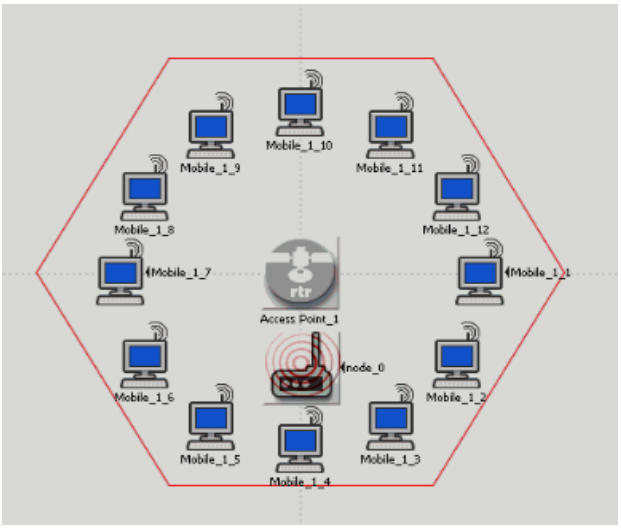


Figure4: Scenario with jammer

Here node_0 represents both single band and pulse jammers in consecutive runs respectively.

Node Attributes

Most of the nodes in the network follow the normal node characteristics as depicted in the figure below.

Each node is of type 802.11n with a data rate of 26 Mbps/240Mbps with a Transmit Power of 0.012W.

An important thing to note is that for all nodes and the Access Point are running on Channel 3 with a frequency of 2422 MHz.

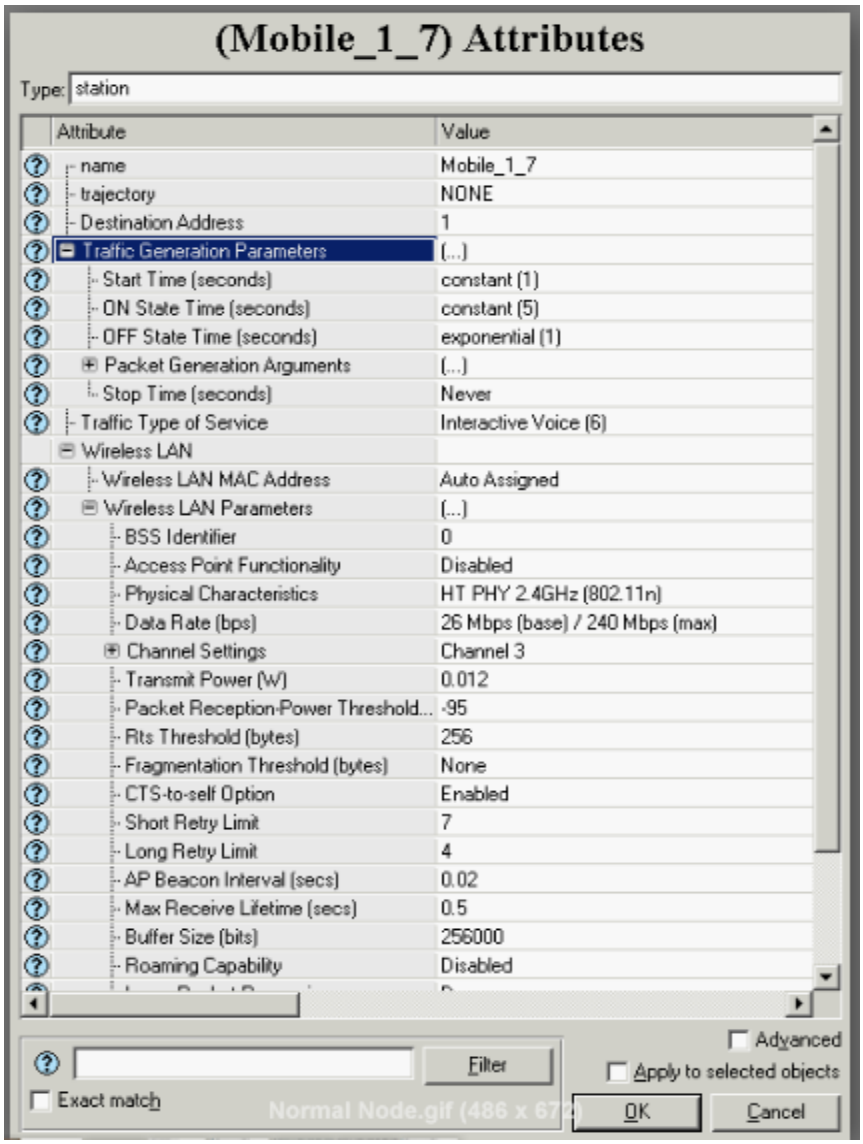


Figure 5: Node characteristics

Some of the nodes are set to a transmitted power lesser than what is shown in the figure to create hidden terminals. The Access Point is also of type 802.11n also operating on channel 3 as shown in the attribute diagram below.

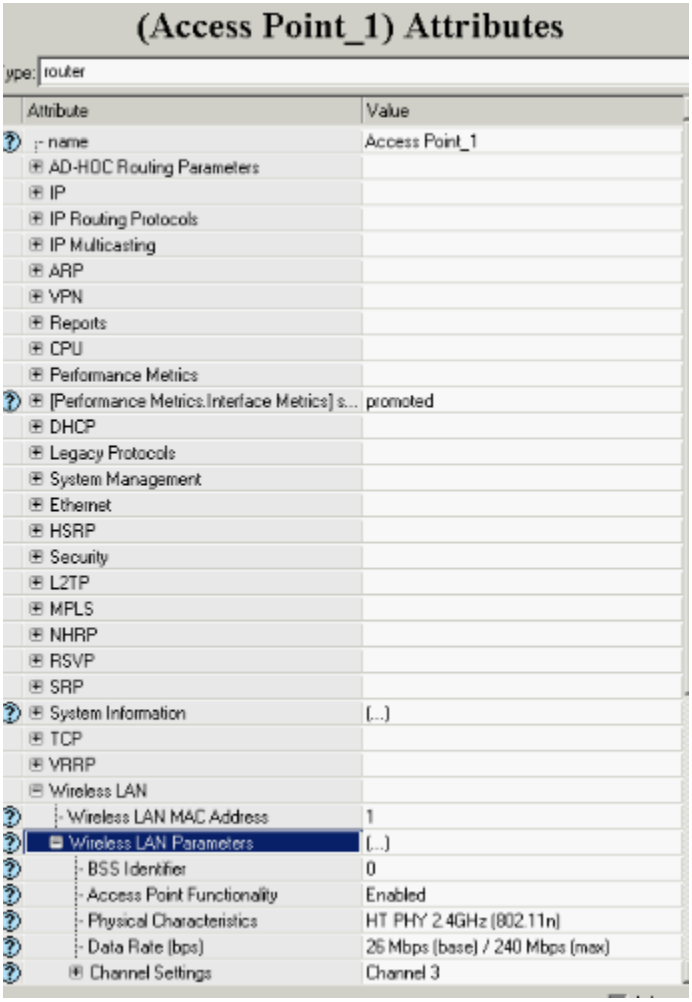


Figure 6: Access point characteristics

4. Simulation Results and Analysis

For all the scenarios, packet sizes of 100, 200, 500, 1000 are used and the parametric results are presented to give an idea of effect of the simulated case of various packet sizes. Packet sizes and inter arrival rates are chosen so as to not exceed the maximum capacity of the 802.11n networks i.e 24 Mbps.

Scenario 1: Effect of RTS/CTS

In this scenario we have 12 nodes transmitting packets at an inter-arrival rate of 0.008 and Maximum Packet Size of 1000 bytes.

The Maximum Load on the system is given by

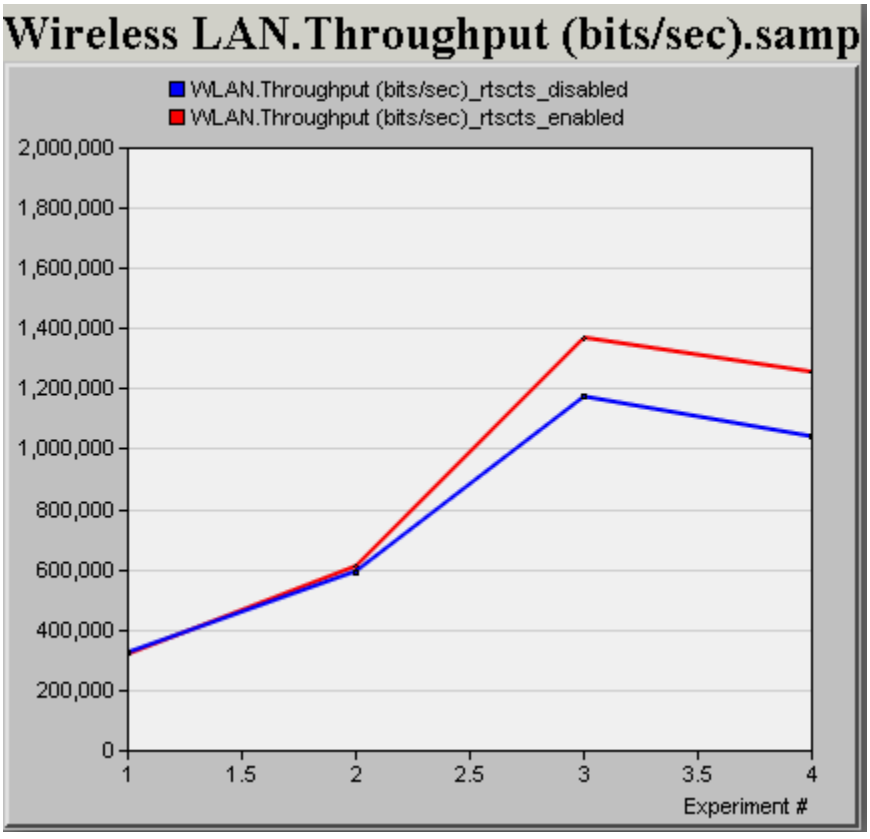
Load = ((Mean ON Time) / (Mean ON Time + Mean OFF Time)) * (Packet size in bits) * (Number of nodes generating traffic) / (Mean Inter-arrival Time)

Load = (5/5) * 1000 * 8 * 12/ 0.008

= 12 Mbps

This load is well within the acceptable limits.

The throughput obtained with and without RTS/CTS is as depicted by the graph below.



Analysis:

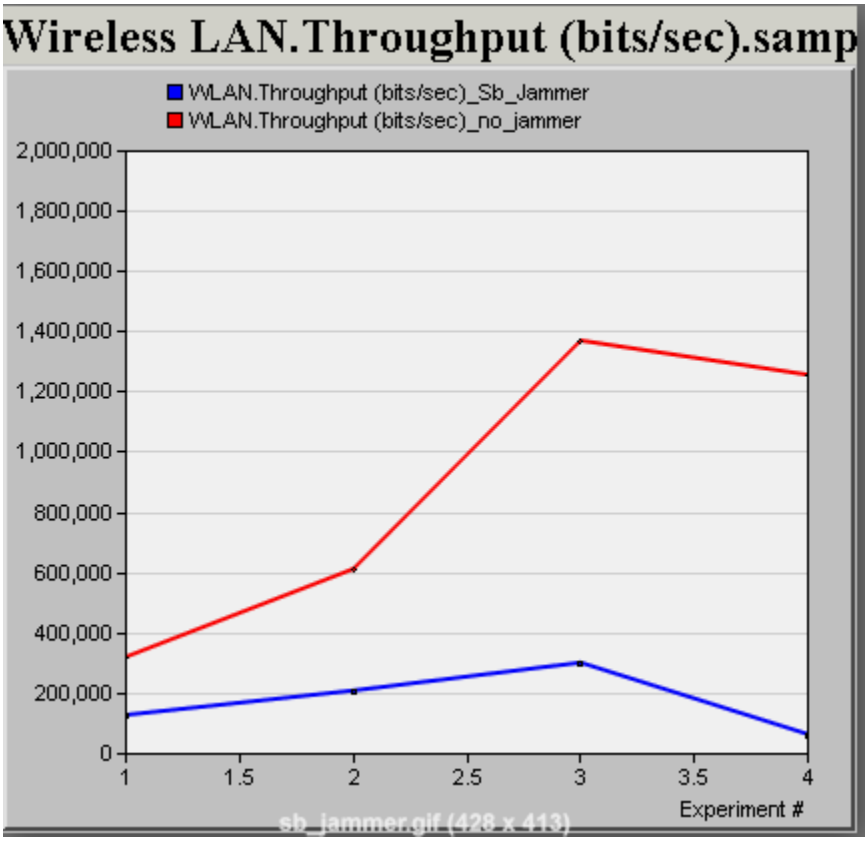
We can see a marked increase in the throughput of RTS/CTS enabled scenario in comparison to the disabled scenario. This difference can be accounted to the fact that there are hidden nodes in the network that transmit at a power that does not reach the other end of the network and thus cause the Hidden Terminal Problem explained before.

With RTS/CTS mechanism we see that *each node transmits an RTS packet before transmission and only after a CTS is received it will start transmission and thus the problem of hidden node* will be resolved which accounts for higher throughput. Another key observation to make from the graph is that the throughput doesn't show a marked increase until experiment 2. This is due to a reason that the packet size is varied between 100, 200, 500 and1000. The RTS/CTS mechanism is an advantage only *when packet size is sufficiently high*. It can thus be inferred that RTS/CTS will give tangible increase only after a packet size of more than 200 for the given power.

Scenario 2: Effect of Jammers

i. Single Band Jammer

Firstly we introduced a single band jammer in the network at the same frequency where the nodes were transmitting. This scenario was aimed at measuring the efficiency of jammer in disrupting the network. The Load on the system is the same as that of the first scenario. The throughput comparison of jammer enabled vs no jammer networks is as shown below. For this use case we have taken a comparison of the RTS/CTS enabled scenario as it is more commonplace in actual networks.

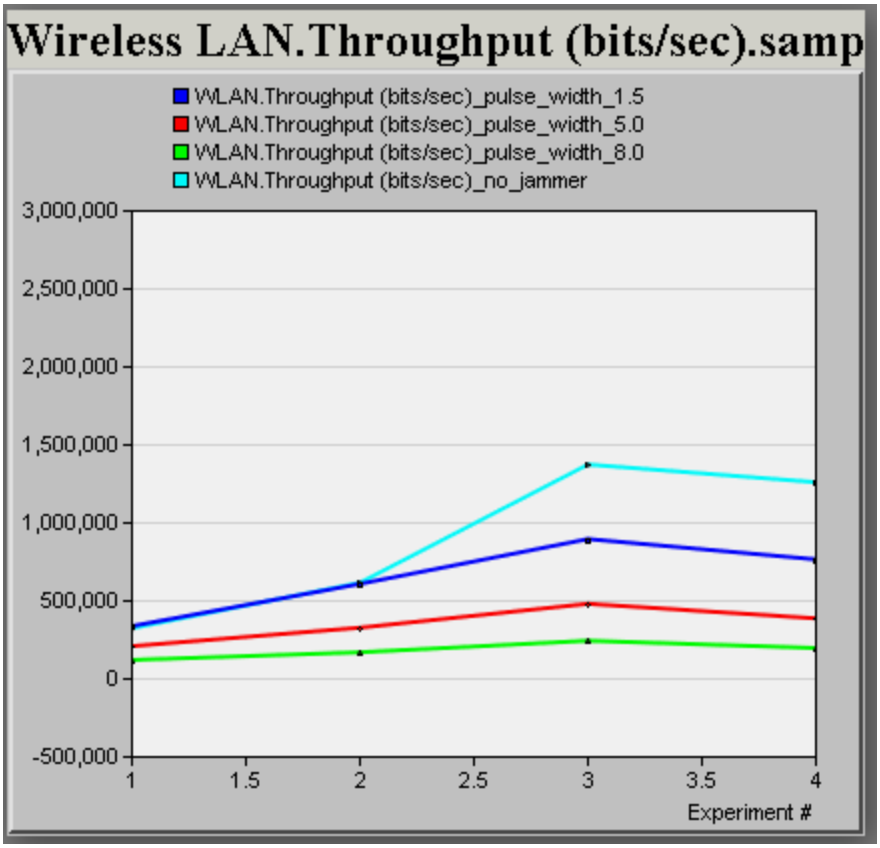


Analysis:

The graph shows a massive decrease in the throughput from 1.4 Mbps to very minimal throughput. This is because of the fact that the jammer is deployed at the same channel as each of the node in the network. The jammer transmits signals at very high inter arrival rate and high power thus effectively decimating all communication in the network. An easy way out to better the throughput of this scenario is to change the channel at which the nodes are transmitting. Thus Single band jammers though highly effective, have limited use and also they consume very high power.

ii. Pulse Jammer

In a single band jammer, the jammer is continuously transmitting packets and thus consumes very high power. An alternative to that is to send signals in pulses of On and Off. Such jammers are termed as pulse jammers. They consume considerably less power than the Single band jammers. But as seen in the throughput graph shown below the effect of jamming is expectedly not as good as of single band jammers.



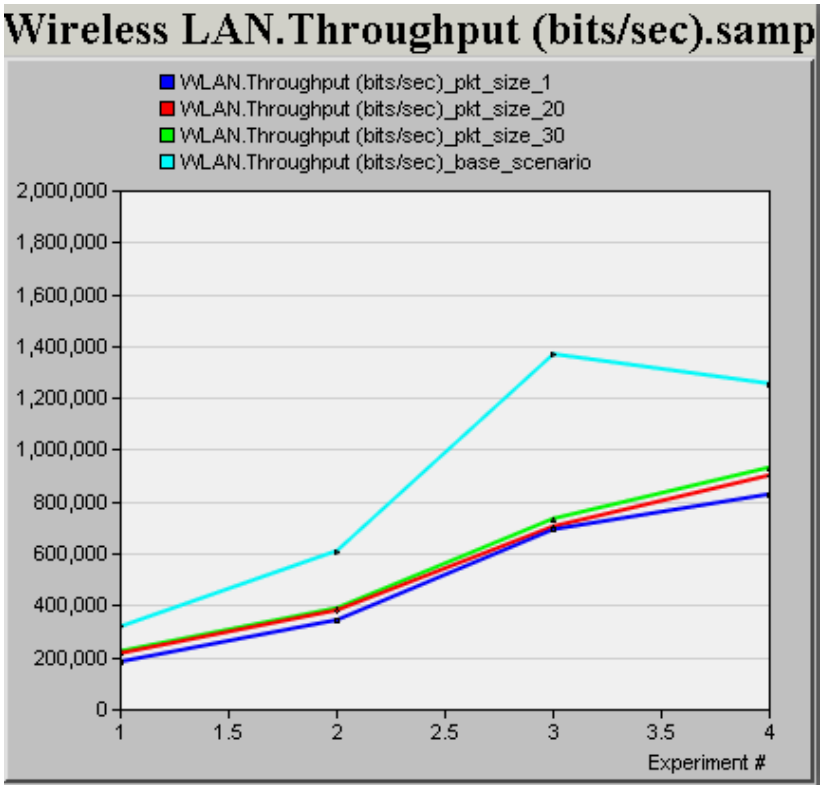
Analysis:

The graph shows a comparative study of throughput at different pulse widths. Pulse width represents the ON time of the jammer. We observe that as the pulse width increases the throughput reaches a near zero. With a very low pulse width of 1.5 we see that the throughput is not considerably different but however we do see a marked difference once the packet size increases. With higher pulse widths the jammer effectively kills most of the communication in the network.

Intelligent Jamming Approaches

Scenario 3: Priority Inversion as means of Jamming the network

As described in [4] we know that when a node transmits at a very small packets of lowe priority and very high inter arrival rate then the throughput of the low priority node is more in comparison to that of the higher frequency nodes thus effectively leading to Priority Inversion. However another aspect of such priority inversion is that the throughput of the network also reduces even when the load remains the same in the network. This can be used effectively as means of jamming the network without actually having a jammer node in the network. We achieve this by reducing the packet size and increasing the inter-arrival size of one of the background node. The result obtained is as follows.



Analysis:

Effective Load on Base Network: 12 Mbps

Load on priority inverted network:

(5/5) * 1000 * 8 * 11/0.008 + (5/5) * 1 * 8 * 1/0.00001

= 11 Mbps + 0.8 Mbps = **11.8 Mbps**

The throughput depicted above clearly proves the cause explained. We see that very low packet size for lower priority node [background traffic node] effectively reduces the throughput to half its value in scenario1. This is due to the fact that, although the load on the system is effectively the same, lower priority packets fill up the buffer at the access point at a rate higher than the higher priority packets . As and when the buffer empties very minimally, the higher priority packets dont have the available memory to accommodate get into the buffer and thus yield to the low priority packets. The low priority node eventually takes over the network transmitting most of its own packets and starving the higher priority packets. Since packets from the rest 11 nodes wait a long time in the queue we see that the throughput of the overall network also reduces considerably. The simulation also shows the relationship between packet size and the throughput. At the same inter-arrival rate, a 1 byte Packet jams the network more effectively than a 30 byte packet indicating that lower the packet size, higher the jamming capability.

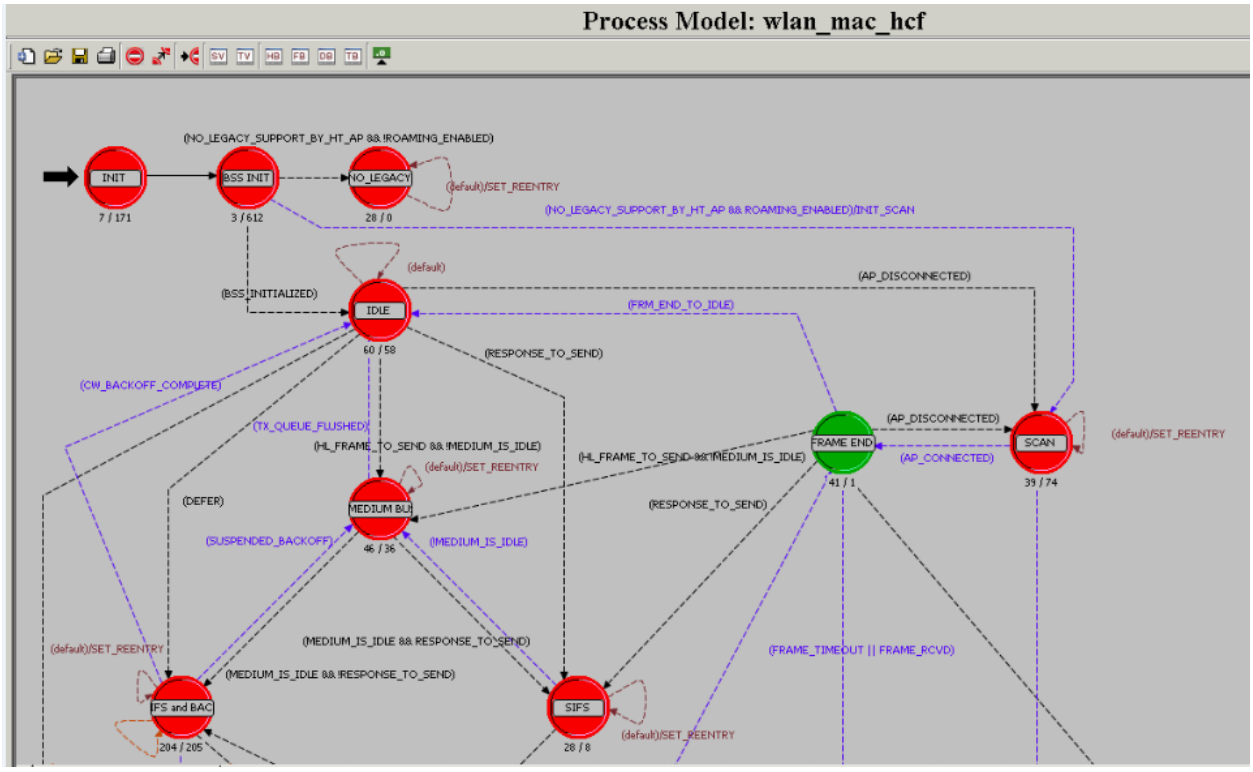
Scenario 4:

The jamming scenarios explained before require an extra jamming node to be placed in the network. [3] gives various methods of intelligent jamming like CTS corrupt jamming, ACK corrupt jamming, data corrupt jamming that jam the network with minimal ways to get detected. In this scenario we try to exploit the MAC protocol and cause network performance degradation using one or more legitimate nodes of the network. The DCF of MAC layer mandates that the nodes should backoff a random number of slots, after the DIFS time before transmitting a packet. The nodes use binary exponential backoff algorithm to determine the number of slots it has to backoff before transmitting. The number of backoff slots increase with the increasing number of collisions that the node experiences. If a node becomes greedy and decides not to adhere to this rule, we might expect that this node gets preferential treatment. We simulate this behavior by making the node transmit immediately after the DIFS period without going through the backoff phase.

To incorporate this, we created a new attribute to the wireless LAN node called as “greedy”. This is a boolean that can be set to specify if the node is behaving greedily or not. This is shown in the figure below.

Attribute	Value
name	Mobile_1_10
trajectory	NONE
Destination Address	1
Traffic Generation Parameters	(...)
Traffic Type of Service	Background (1)
Wireless LAN	
wireless_lan_mac.Greedy	Yes
Traffic Generation Parameters [0].Packet...	promoted

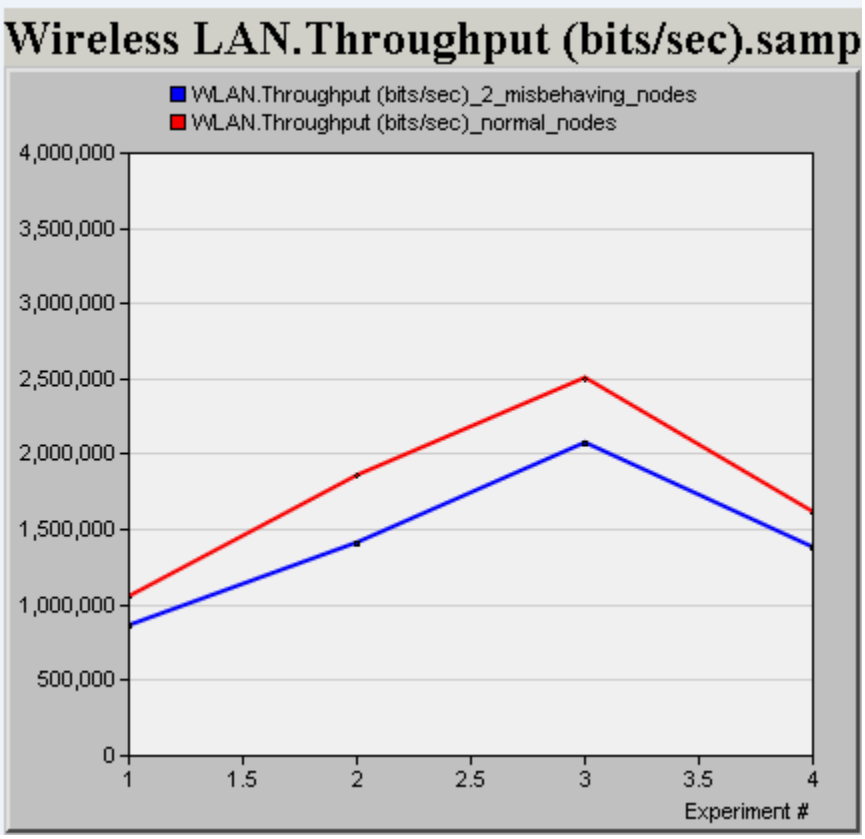
To change the behavior of the MAC layer, we changed the process model of wlan_hcf. By looking at the wlan_dispatch process model, it was observed that the wlan_mac_hcf is invoked in our network as we have all 802.11n nodes with HCF enabled. The wlan_mac_hcf process model is as shown below:



The IFS and BACKOFF enter executives have the calculations for the backoff slots to be used by the node. Minor changes was made in this enter executive to read the “greedy” attribute of the node and if it is set, the backoff calculations are nullified by setting the backoff slots directly to zero. The code of the IFS and BACKOFF enter executive is the appendix.

With this change to the process model, two nodes in the network are made to misbehave by setting the “greedy” attribute. Also to make the effect of the misbehavior apparent, the network is made fully visible by increasing the transmitted power of all the nodes.

The traffic generation parameters are unchanged from the base scenario. The effect on the throughput of the network can be seen in the figure below:



Analysis:

The degradation in throughput is because of the misbehaving node. The misbehaving nodes by aggressively transmitting after the DIFS time make other nodes backoff more and more, effectively reducing the throughput. Such kind of nodes can be used to intentionally degrade the network performance just as jammers do. The Load produced by each misbehaving node is exactly the same as any normal node and thus detection of such attacks is also a hard problem making the jamming technique more effective.

Conclusion and Future work:

We have successfully demonstrated that the use of RTS/CTS improves the overall throughput of the network in the presence of hidden nodes. By analyzing the effect of different types of jammers on the throughput, we see that single band jammer is more effective than pulse band jammes. But pulse band jammers give a decent jamming effect with lesser power consumed than the single band jammers as they have an off period between pulses. The network performance degradation can also be achieved by having nodes transmit smaller packets of lower priority at a high frequency. This effect is because of the “priority inversion” phenomenon that occurs in 802.11 networks as explained in [4]. Misbehaving nodes not following the MAC protocol can also cause performance degradation by refusing to backoff after the DIFS period. Any node not

following any of the MAC specific rules will cause network degradation. As a future work, we can explore this by modifying the MAC process model to change the DIFS, SIFS, AIFS times and cause performance degradation from within the network.

References:

[1] Shaiful Alam Chowdhury, Mohamamd Tauhldul lslam, Fariha Tasmin Jaigirdar, Md. Rokan Uddin Faruqui, Shahid AI Noor, "Performance study and simulation analysis of CSMA and IEEE 802.11 in Wireless Sensor Networks and limitations of IEEE 802.11", InternationalConference on Computer and Information Technology,December 2009

[2] Acharya, M., and D. Thuente, "Intelligent Jamming Attacks, Counterattacks and (Counter)2 Attacks in 802.11b Wireless Networks",OPNETWORK 2005, September 2005.

[3] D.J. Thuente and M. Acharya "Intelligent Jamming in Wireless Networks with Applications to 802.11b and Other Networks," in IEEE MILCOM 2006.

[4] Formyduval, W. L., & Thuente, D. J. (2013). Priority inversion and queue management for 802.11 priority WLANs. (*2013 IEEE Consumer Communications and Networking Conference (CCNC),*) (pp. 565-573).

[5] Thuente D.J, Newlin Benjamin, Acharya M, "Jamming Vulnerabilites in 802.11e" , Military Conmmunications Conference 2007

[6] Kyasanur, P., N. Vaidya, "Detection and Handling of MAC LayerMisbehavior in Wireless Networks", DSN 2003.

[7] Tiang Fu, "Modeling and Simulation of Jamming attacks in WLAN",Thesis Work, Department of Technology Systems, East Carolina University, April 2012.

[8] Jost Weinmiller*, Hagen Woesner*, Jean-Pierre Ebert*, Adam Wolisz, Analyzing the RTS/CTS Mechanism in the DFWMAC Media Access Protocol for Wireless LANs, IFIP TC6 Workshop Personal Wireless Comm.

Appendix

Modifications made to code are indicated in red.

```
/* We enter into this state when we have a high layer or management      */
/* frame in at least one access category to send and/or we have a   */
/* contention window backoff (backoff after a successful                 */
/* transmission) to perform, and the medium is idle. We will leave  */
/* this state when our deference (inter-frame spacing and backoff)  */
/* is complete or if the medium becomes busy.                            */

/* Make sure this is not a reentry into this state.                      */
op_ima_obj_attr_get (my_objid, "Greedy", &greedy);

//greedy = OPC_BOOLINT_ENABLED;
if (state_reentered == OPC_FALSE)
       {
       /* Determine the AIFS and backoff durations and schedule an         */
       /* interrupt for the end of it. If there are multiple access */
       /* categories contending then schedule the interrupt only for*/
       /* the AC that will complete its deference first.                   */

       /* Reset the ht_protection_state to Unprotected. As we will  */
       /* might enter this state immediately after receiving a             */
       /* response frame after a successful transmission.                  */
       hcf_protection_state = WlanC_HCF_Unprotected;

       /* We mark that this state is unsafe to process the inactivity      */
       /* timeout process at the recipient end of a block ACK              */
       /* agreement with delayed BA policy                                                */
       wlan_flags->delay_inact_tout_proc = OPC_TRUE;

       /* Initialize the temporary variables used to find out the          */
       /* access category with the shortest backoff period.                */
       min_backoff_slots = OPC_INT_INFINITY;
       max_cw_slots      = 0;

       /* Adjust the medium idle time, which is the current time, if*/
       /* the EIFS flag is set (section 9.2.3.4).                                 */
       if (wlan_flags->wait_eifs_dur == OPC_TRUE)
              {
              /* Check if we have entered this state when performing a     */
              /* backoff due to busy secondary channel .                          */
              if (backoff_on_secondary_busy == OPC_FALSE)
                     {
                     /* When we are performing a EIFS wait due to a              */
                     /* reception activity during our last CTS-to-self           */
                     /* transmission, then we consider the entire EIFS time      */
                     /* while calculating the medium idle time for backoff.      */
                     /* Otherwise we discount the "EIFS - DIFS" while            */
                     /* calculating medium idle time for backoff.                */
                     if(wlan_flags->bad_cts_dropped == OPC_FALSE)
                            /* Adjust the idle time by "EIFS - DIFS" based       */
                            /* on receiver idle time.                                          */
                            bk_medium_idle_time   =   rcv_idle_time   +   eifs_time   -
difs_time;
                     else
```

```
                        bk_medium_idle_time = rcv_idle_time + eifs_time;

                /* Make sure that the EIFS flag is recent.                          */
                if (bk_medium_idle_time < MEDIUM_IDLE_TIME)
                        bk_medium_idle_time = MEDIUM_IDLE_TIME;
                }
        else
                /* If we are trying to send a 40 MHz PPDU and found    */
                /* that the secondary channel was busy, we may                */
                /* perform a backoff for transmitting the 40 MHz PPDU.        */
                /* In such a case perform a backoff operation starting        */
                /* from the current time.                                                   */
                bk_medium_idle_time = current_time;

        /* Reset the EIFS flag.
        wlan_flags->wait_eifs_dur = OPC_FALSE;
        }
    else
        {
        /* If we are trying to send a 40 MHz PPDU and found that      */
        /* the secondary channel was busy, we may optionally          */
        /* perform a backoff for transmitting the 40 MHz PPDU. In     */
        /* such a case perform a backoff operation starting from      */
        /* the current time.                                                        */
        bk_medium_idle_time   =   (backoff_on_secondary_busy   ==   OPC_FALSE)   ?
MEDIUM_IDLE_TIME : current_time;
        }

    /* Go over the list of the active access categories.                */
    for (i = 0; ac_queue_status_arr [i] != WlanC_AC_None; i++)
        {
        /* Store the current AC in a temporary variable.                    */
        ac = ac_queue_status_arr [i];

        /* Is this a new backoff or a resumed backoff?                      */
        if (deference_info_arr [ac].backoff_slots == BACKOFF_SLOTS_UNSET)
            {
            /* If we are performing a backoff due to a busy                 */
            /* secondary channel, then do not increament the                */
            /* retry counters and the contention window.                    */
            if (backoff_on_secondary_busy == OPC_FALSE)
                {
                /* A new backoff. Is it a retransmission?                           */
                if (src_arr [ac] + lrc_arr [ac] > 0)
                    {
                    /* This is a retransmission. Increase the CW    */
                    /* size unless it already reached its maximum   */
                    /* size.
                    if (cw_arr [ac] < cwmax_arr [ac])
                        cw_arr [ac] = cw_arr [ac] * 2 + 1;
                    }
                else
                    /* First trial. Hence, CW size equals to                    */
                    /* CWmin.
                    cw_arr [ac] = cwmin_arr [ac];
                }

            /* Pick a random number of backoff slots from CW.               */


            deference_info_arr     [ac].backoff_slots     =     (int)     floor
(op_dist_uniform (cw_arr [ac] + 1));
            if(greedy == OPC_BOOLINT_ENABLED)
```

```
                        deference_info_arr [ac].backoff_slots = 0;

                    /* Update the dimensioned backoff slot statistic for  */
                    /* this ac.
                    op_stat_write    (ac_backoff_slots_shndl_arr    [ac],    (double)
deference_info_arr [ac].backoff_slots);

                    /* Write an ODB trace message if enabled.                          */
                    if (wlan_trace_active)
                          {
                          sprintf (msg1, "Initiating a backoff period for AC %s for
%d slots (CW size = %d slots aifsn_arr [ac] = %d , ac is %d).",
                                                WLANC_AC_NAME_ARRAY [ac], deference_info_arr
[ac].backoff_slots, cw_arr [ac], aifsn_arr [ac], ac);
                          op_prg_odb_print_major (msg1, OPC_NIL);
                          }

                    }

             /* We are resuming an already scheduled backoff. Write an    */
             /* ODB trace message if enabled.                                          */
             else if (wlan_trace_active)
                    {
                    sprintf  (msg1,  "Resuming  backoff  period  for  AC  %s  for  the
remaining %d slots aifsn_arr [ac] = %d , ac is %d",
                                                  WLANC_AC_NAME_ARRAY              [ac],
deference_info_arr [ac].backoff_slots, aifsn_arr [ac], ac);
                    op_prg_odb_print_major (msg1, OPC_NIL);
                    }

             /* Is the queue of the access category empty? If it is then  */
             /* this backoff is just to complete a successful                    */
             /* transmission.
             deference_info_arr [ac].type = (op_prg_list_size (hlpk_lptr_arr [ac]) ==
0) ? WlanC_CW_Elapsed : WlanC_Backoff_Elapsed;

             /* Find out the total number of slots in the deference       */
             /* period.
             ac_total_slots = aifsn_arr [ac] + deference_info_arr [ac].backoff_slots;

             /* Update the lowest/highest number of slots across the ACs  */
             /* if necessary.
             if (deference_info_arr [ac].type == WlanC_Backoff_Elapsed)
                    {
                    if (ac_total_slots < min_backoff_slots)
                          {
                          min_backoff_slots = ac_total_slots;
                          min_backoff_ac    = ac;
                          }
                    }
             else
                    {
                    if (ac_total_slots > max_cw_slots)
                          {
                          max_cw_slots = ac_total_slots;
                          max_cw_ac    = ac;
                          }
                    }
             }

       /* Schedule an end of deference interrupt for the access            */
       /* category that has the shortest time when its AIFSN and           */
       /* random backoff slots are added. In case of a tie, then pick      */
```

```
	/* the AC with higher priority. We will schedule an interrupt */
	/* for a CW-backoff (backoff for ACs with empty queue, which is     */
	/* performed due to the recent successful transmission) only if     */
	/* there is not an AC performing a regular backoff. In that         */
	/* case, schedule the interrupt for the longest CW-backoff so */
	/* that when the interrupt delivered, all the CW-backoff will */
	/* be complete, which will enable us to go back to the IDLE         */
	/* state, unless we receive a packet from the higher layer          */
	/* during the backoff period.                                                    */
	if (min_backoff_slots != OPC_INT_INFINITY)
		{
		backoff_ac  = min_backoff_ac;
		intrpt_time   =   bk_medium_idle_time   +   sifs_time   +   slot_time   *
min_backoff_slots;
		deference_info_arr            [backoff_ac].deference_end_evh            =
op_intrpt_schedule_self (intrpt_time, WlanC_Backoff_Elapsed);
		}
	else
		{
		backoff_ac  = max_cw_ac;
		intrpt_time = bk_medium_idle_time + sifs_time + slot_time * max_cw_slots;
		deference_info_arr            [backoff_ac].deference_end_evh            =
op_intrpt_schedule_self (intrpt_time, WlanC_CW_Elapsed);
		}

	/* Record the state name for debugging purposes.                            */
	if (debug_mode)
		{
		strcpy (current_state_name, "IFS and BACKOFF");
		}
	}

#if defined (OPD_PARALLEL)
/* Unlock the mutex that serializes accessing the roaming related   */
/* information of this MAC.                                                      */
op_prg_mt_mutex_unlock (roam_state_ptr->roam_info_mutex);
#endif
```