1. Understand the events involved in NetSim DES (Discrete Event Simulator) in simulating flow of one packet from a Wired node to a Wireless node (Level 2)

1.1 Theory

NetSim's Network Stack forms the core of NetSim, and its architectural aspects are diagrammatically explained below. Network Stack accepts inputs from the end-user in the form of Configuration file and the data flows as packets from one layer to another layer in the Network Stack. All packets, when transferred between devices move up and down the stack, and all events in NetSim fall under one of these ten categories of events, namely, **Physical IN, Data Link IN, Network IN, Transport IN, Application IN, Application Out, Transport OUT, Network OUT, Data Link OUT and Physical OUT.** The IN events occur when the packets are entering a device while the OUT events occur while the packet is leaving a device.

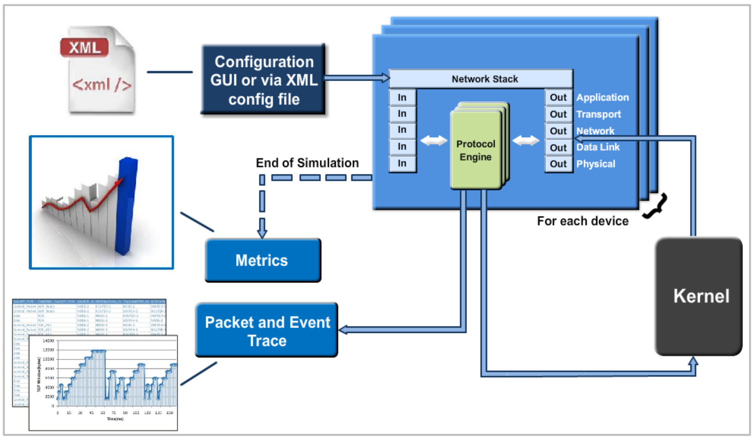

Figure 1-1: Flow of one packet from a Wired node to a Wireless node

Every device in NetSim has an instance of the Network Stack shown above. Switches & Access points have a 2-layer stack, while routers have a 3-layer stack. End-nodes have a 5-layer stack.

The protocol engines are called based on the layer at which the protocols operate. For example, TCP is called during execution of Transport IN or Transport OUT events, while 802.11b WLAN is called during execution of MAC IN, MAC OUT, PHY IN and PHY OUT events.

When these protocols are in operation, they in turn generate events for NetSim's discrete event engine to process. These are known as SUB EVENTS. All SUB EVENTS, fall into one of the above 10 types of EVENTS.

Each event gets added in the Simulation kernel by the protocol operating at the particular layer of the Network Stack. The required sub events are passed into the Simulation kernel. These sub events are then fetched by the Network Stack in order to execute the functionality of each protocol. At the end of Simulation, Network Stack writes trace files and the Metrics files that assist the user in analyzing the performance metrics and statistical analysis.

Event Trace

The event trace records every single event along with associated information such as time stamp, event ID, event type etc. in a text file or .csv file which can be stored at a user defined location.

1.2 Network Setup

Open NetSim and click on **Experiments> Internetworks> Network Performance> Advanced Simulation events in NetSim for transmitting one packet** then click on the tile in the middle panel to load the example as shown in below Figure 1-2.

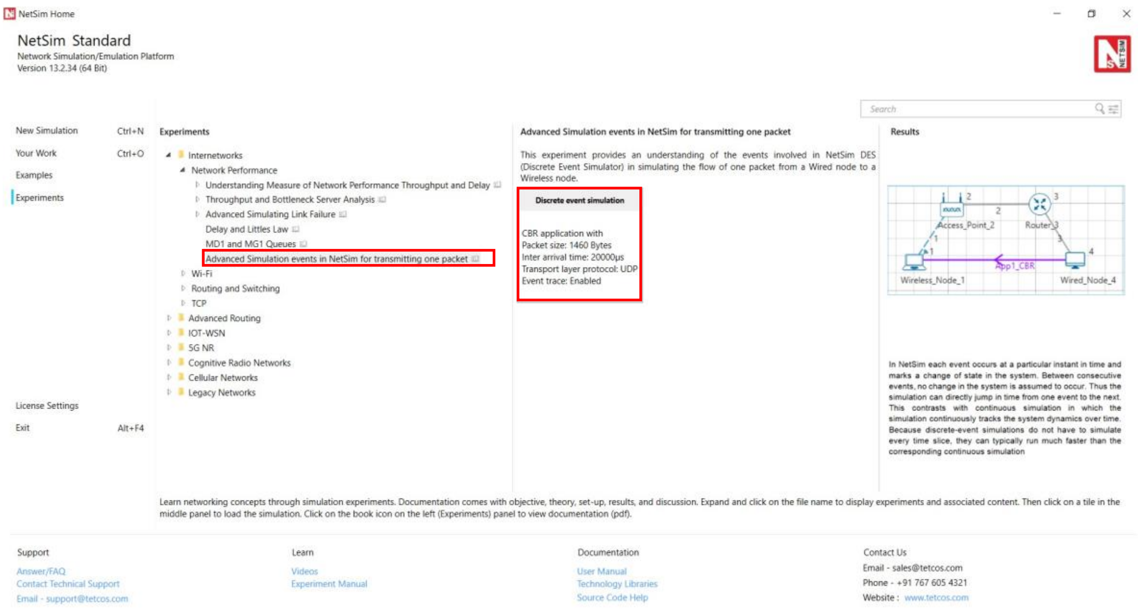

Figure 1-2: List of scenarios for the example of Advanced Simulation events in NetSim for transmitting one packet

NetSim UI displays the configuration file corresponding to this experiment as shown below Figure 1-3.

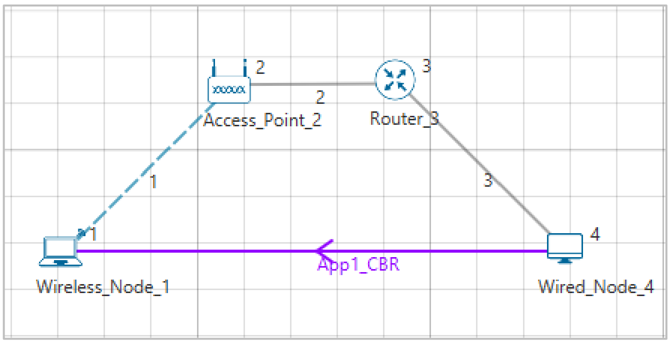

Figure 1-3: Network set up for studying the Advanced Simulation events in NetSim for transmitting one packet

1.3 Procedure

The following set of procedures were done to generate this sample:

Step 1: A network scenario is designed in NetSim GUI comprising of 1 Wired Node, 1 Wireless Node, 1 Router, and 1 Access Point in the **"Internetworks"** Network Library.

Step 2: The device positions are set as per the below Table 1-1.

Device Positions				
	Access Point 2	**Wired Node 4**	**Wireless Node 1**	**Router 3**
X / Lon	150	250	100	200
Y / Lat	50	100	100	50

Table 1-1: Devices Positions

Step 3: Right-click the link ID (of the wireless link) and select Properties to access the link's properties. The **"Channel Characteristics"** is set to NO PATHLOSS.

Step 4: Right click on the Application Flow **App1 CBR** and select Properties or click on the Application icon present in the top ribbon/toolbar.

A CBR Application is generated from Wired Node 4 i.e., Source to Wireless Node 1 i.e., Destination with Packet Size remaining 1460 Bytes and Inter Arrival Time remaining 20000µs. Transport Protocol is set to UDP instead of TCP.

Step 5: Event Trace is enabled from the options in NetSim GUI.

Step 6: Run the simulation for 10 secs. At the end of the simulation, a very large .csv file contains all the UDP IN and OUT EVENTS is available for the users. Plots are enabled in NetSim GUI.

NOTE: Event trace is available only in NetSim Standard and Pro versions.

1.4 Output

Once the simulation is complete, go to the Results Dashboard and in the left-hand-side of the window, click on the **"Open Event Trace"** Option. An Event trace file like the following opens in Excel as shown below:

Event_Id	Event_Type	Event_Time(US)	Device_Type	Device_Id	Interface_Id	Application_Id	Packet_Id	Segment_Id	Protocol_Name	Subevent_Type	Packet_Size(Bytes)	Prev_Event_Id
1	TIMER_EVENT	0	NODE	1	0	0	0	0	IPV4	IP_INIT_TABLE	0	0
2	TIMER_EVENT	0	ROUTER	3	0	0	0	0	IPV4	IP_INIT_TABLE	0	0
3	TIMER_EVENT	0	NODE	4	0	0	0	0	IPV4	IP_INIT_TABLE	0	0
4	TIMER_EVENT	0	ACCESSPOINT	2	2	0	0	0	ETHERNET	ETH_IF_UP	0	0
5	TIMER_EVENT	0	ROUTER	3	1	0	0	0	ETHERNET	ETH_IF_UP	0	0
6	TIMER_EVENT	0	ROUTER	3	2	0	0	0	ETHERNET	ETH_IF_UP	0	0
7	TIMER_EVENT	0	NODE	4	1	0	0	0	ETHERNET	ETH_IF_UP	0	0
8	TIMER_EVENT	0	NODE	4	0	1	1	0	APPLICATION	APP_START	1460	0
9	APPLICATION_	0	NODE	4	0	1	1	0	APPLICATION	0	1460	8
11	TRANSPORT_O	0	NODE	4	0	1	2	0	UDP	0	1460	9
12	NETWORK_OU	0	NODE	4	0	1	1	0	IPV4	0	1468	11
13	TIMER_EVENT	0	NODE	4	1	1	1	0	IPV4	IP_PROCESSING_DELAY	1488	12
14	MAC_OUT	0	NODE	4	1	1	1	0	ETHERNET	0	1488	13
15	PHYSICAL_OUT	0	NODE	4	1	1	1	0	ETHERNET	0	1514	14
16	PHYSICAL_IN	127.08	ROUTER	3	2	1	1	0	ETHERNET	0	1514	15
17	MAC_IN	127.08	ROUTER	3	2	1	1	0	ETHERNET	0	1514	16
18	NETWORK_IN	127.08	ROUTER	3	2	1	1	0	IPV4	0	1488	17
19	NETWORK_OU	127.08	ROUTER	3	2	1	1	0	IPV4	0	1468	18
20	TIMER_EVENT	127.08	ROUTER	3	1	1	1	0	IPV4	IP_PROCESSING_DELAY	1488	19
21	MAC_OUT	127.08	ROUTER	3	1	1	1	0	ETHERNET	0	1488	20
22	PHYSICAL_OUT	127.08	ROUTER	3	1	1	1	0	ETHERNET	0	1514	21
23	PHYSICAL_IN	253.2	ACCESSPOINT	2	2	1	1	0	ETHERNET	0	1514	22
24	MAC_IN	253.2	ACCESSPOINT	2	2	1	1	0	ETHERNET	0	1514	23
25	MAC_OUT	253.2	ACCESSPOINT	2	1	1	1	0	WLAN	0	1488	24
26	MAC_OUT	253.2	ACCESSPOINT	2	1	1	1	0	WLAN	CS	1488	25
27	MAC_OUT	303.2	ACCESSPOINT	2	1	1	1	0	WLAN	IEEE802_11_EVENT_DIFS_EN	1488	26
28	MAC_OUT	323.2	ACCESSPOINT	2	1	1	1	0	WLAN	IEEE802_11_EVENT_BACKOF	1488	27
29	MAC_OUT	343.2	ACCESSPOINT	2	1	1	1	0	WLAN	IEEE802_11_EVENT_BACKOF	1488	28
30	MAC_OUT	363.2	ACCESSPOINT	2	1	1	1	0	WLAN	IEEE802_11_EVENT_BACKOF	1488	29

Figure 1-4: Event trace

We start from the **APPLICATION_OUT** event of the first packet, which happens in the Wired Node and end with the **MAC_IN** event of the **WLAN_ACK** packet which reaches the Wired Node. Events in the event trace are logged with respect to the time of occurrence due to which, event id may not be in order.

1.4.1 Events Involved

Events are listed in the following format:

[EVENT_TYPE,	EVENT_TIME,	PROTOCOL,	EVENT_NO,	SUBEVENT_TYPE]
[APP_OUT,	20000,	APP,	6,	-]
[TRNS_OUT,	20000,	UDP,	7,	-]
[NW_OUT,	20000,	IPV4,	9,	-]
[MAC_OUT,	20000,	ETH,	10,	-]
[MAC_OUT,	20000,	ETH,	11,	CS]
[MAC_OUT,	20000.96,	ETH,	12,	IFG]
[PHY_OUT,	20000.96,	ETH,	13,	-]

```
[PHY_OUT,        20122.08,   ETH,    14,   PHY_SENSE]
[PHY_IN,         20127.08,   ETH,    15,   -]
[MAC_IN,         20127.08,   ETH,    16,   -]
[NW_IN,          20127.08,   IPV4,   17,   -]
[NW_OUT,         20127.08,   IPV4,   18,   -]
[MAC_OUT,        20127.08,   ETH,    19,   -]
[MAC_OUT,        20127.08,   ETH,    20,   CS]
[MAC_OUT,        20128.04,   ETH,    21,   IFG]
[PHY_OUT,        20128.04,   ETH,    22,   -]
[PHY_OUT,        20249.16,   ETH,    23,   PHY_SENSE]
[PHY_IN,         20254.16,   ETH,    24,   -]
[MAC_IN,         20254.16,   ETH,    25,   -]
[MAC_OUT,        20254.16,   WLAN,   26,   -]
[MAC_OUT,        20254.16,   WLAN,   27,   DIFS_END]
[MAC_OUT,        20304.16,   WLAN,   28,   BACKOFF]
[MAC_OUT,        20324.16,   WLAN,   29,   BACKOFF]
[MAC_OUT,        20344.16,   WLAN,   30,   BACKOFF]
[MAC_OUT,        20364.16,   WLAN,   31,   BACKOFF]
[PHY_OUT,        20364.16,   WLAN,   32,   -]
[TIMER_EVENT,    21668.16,   WLAN,   35,   UPDATE_DEVICE_STATUS]
[PHY_IN,         21668.4,    WLAN,   33,   -]
[MAC_IN,         21668.4,    WLAN,   36,   RECEIVE_MPDU]
[NW_IN,          21668.4,    IPV4,   37,   -]
```

[MAC_OUT,	21668.4,	WLAN,	38,	SEND_ACK]
[TRNS_IN,	21668.4,	UDP,	39,	-]
[APP_IN,	21668.4,	APP,	41,	-]
[PHY_OUT,	21678.4,	WLAN,	40,	-]
[TIMER_EVENT,	21982.4,	WLAN,	43,	UPDATE_DEVICE]
[PHY_IN,	21982.63,	WLAN,	42,	-]
[MAC_IN,	21982.63,	WLAN,	44,	RECEIVE_ACK]
[TIMER_EVENT,	21985,	WLAN,	34,	ACK_TIMEOUT]

Event Flow Diagram for one packet from Wired Node to Wireless Node

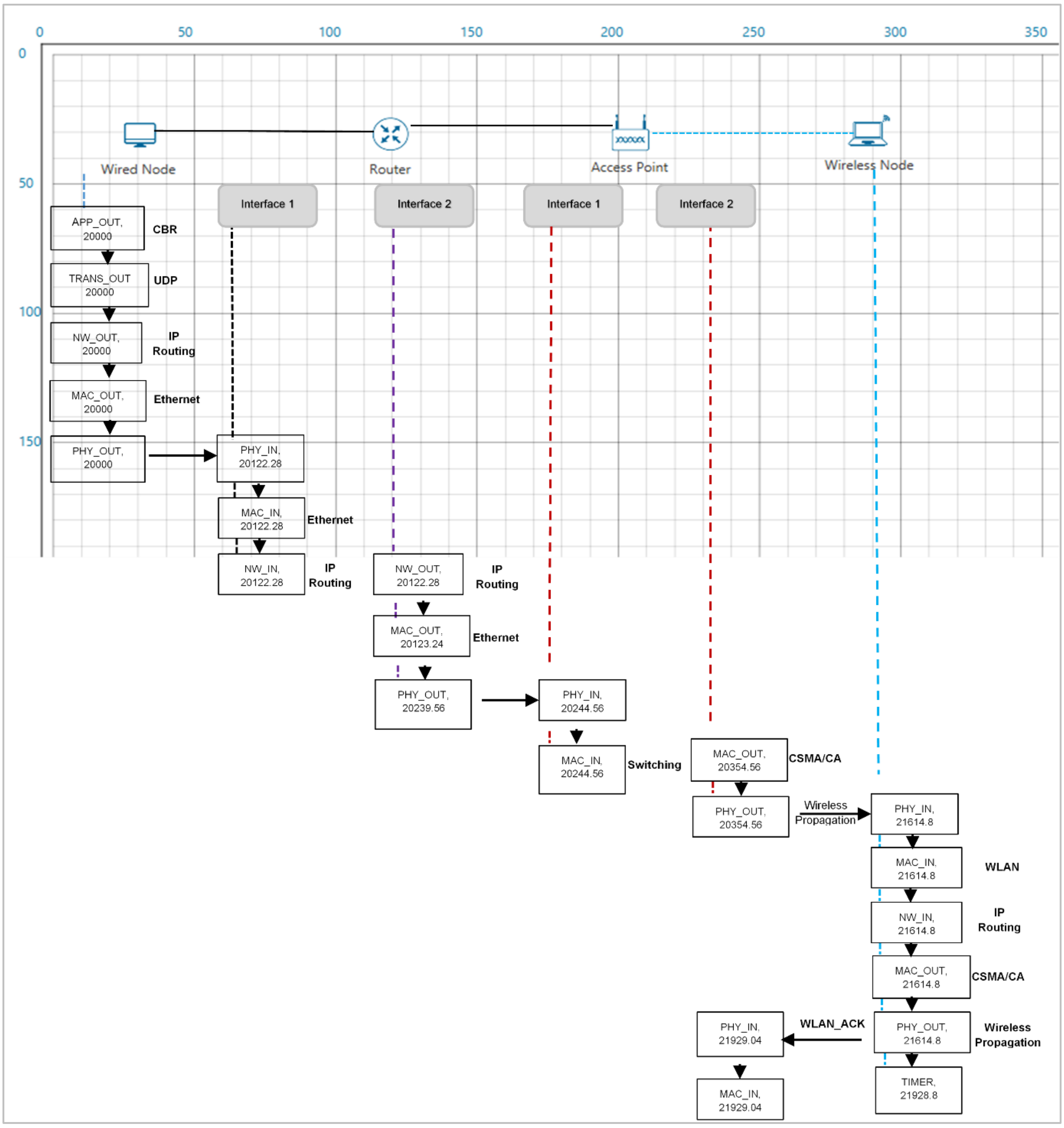

Figure 1-5: Event Flow Diagram for one packet from Wired Node to Wireless Node

For Example:

MAC_OUT in the Access Point involves sub events like CS, IEEE802.11_EVENT_DIFS_END and IEEE802.11_EVENT_BACKOFF. As you can see in the trace file shown below, CS happens at event time 20252.24. Adding DIFS time of 50µs to this will give IEEE802.11_EVENT_DIFS_END sub event at 20302.24. Further it is followed by three backoffs each of 20 µs, at event time 20322.24, 20342.24, 020362.24 respectively.

Event_Id	Event_Type	Event_Time(US)	Device_Type	Device_Id	Interface_Id	Applicatio	Packet_Id	Segment_Id	Protocol_Name	Subevent_Type	Packet_Size	Prev_Event_Id
53	TIMER_EVENT	20126.12	ROUTER	3	1	1	2	0	IPV4	IP_PROCESSING_DELAY	1488	52
54	MAC_OUT	20126.12	ROUTER	3	1	1	2	0	ETHERNET	0	1488	53
55	PHYSICAL_OUT	20126.12	ROUTER	3	1	1	2	0	ETHERNET	0	1514	54
56	PHYSICAL_IN	20252.24	ACCESSPOINT	2	2	1	2	0	ETHERNET	0	1514	55
57	MAC_IN	20252.24	ACCESSPOINT	2	2	1	2	0	ETHERNET	0	1514	56
58	MAC_OUT	20252.24	ACCESSPOINT	2	1	1	2	0	WLAN	0	1488	57
59	MAC_OUT	20252.24	ACCESSPOINT	2	1	1	2	0	WLAN	CS	1488	58
60	MAC_OUT	20302.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_DIFS_END	1488	59
61	MAC_OUT	20322.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	60
62	MAC_OUT	20342.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	61
63	MAC_OUT	20362.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	62
64	MAC_OUT	20382.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	63
65	MAC_OUT	20402.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	64
66	MAC_OUT	20422.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	65
67	MAC_OUT	20442.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	66
68	MAC_OUT	20462.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	67
69	MAC_OUT	20482.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	68
70	MAC_OUT	20502.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	69
71	MAC_OUT	20522.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	70
72	MAC_OUT	20542.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	71
73	MAC_OUT	20562.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	72
74	MAC_OUT	20582.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_EVENT_BACKOFF	1488	73
75	PHYSICAL_OUT	20582.24	ACCESSPOINT	2	1	1	2	0	WLAN	IEEE802_11_PHY_TXSTART_REQUEST	1528	74
76	PHYSICAL_IN	21886.25	NODE	1	1	1	2	0	WLAN	0	1528	75
78	MAC_IN	21886.25	NODE	1	1	1	2	0	WLAN	RECEIVE_MPDU	1528	76
79	NETWORK_IN	21886.25	NODE	1	1	1	2	0	IPV4	0	1528	78
80	MAC_OUT	21886.25	NODE	1	1	1	2	0	WLAN	SEND_ACK	1528	78
81	TRANSPORT_IN	21886.25	NODE	1	1	1	2	0	UDP	0	1468	79
83	APPLICATION_IN	21886.25	NODE	1	1	1	2	0	APPLICATION	0	1460	81

Figure 1-6: Sub events like CS, IEEE802.11_EVENT_DIFS_END and IEEE802.11_EVENT_BACKOFF event times

In this manner the event trace can be used to understand the flow of events in NetSim Discrete Event Simulator.

1.5 Discussion

In NetSim each event occurs at a particular instant in time and marks a change of state in the system. Between consecutive events, no change in the system is assumed to occur. Thus the simulation can directly jump in time from one event to the next.

This contrasts with continuous simulation in which the simulation continuously tracks the system dynamics over time. Because discrete-event simulations do not have to simulate every time slice, they can typically run much faster than the corresponding continuous simulation.

Understanding NetSim's Event trace and its flow is very much helpful especially when customizing existing code and debugging to verify the correctness the modified code. The event IDs provided in the event trace can be used to go to a specific event while debugging.