

Sink Hole Attack in MANET using DSR.

Software: NetSim Standard v13.3, Visual Studio 2022

Project Download Link:

<https://github.com/NetSim-TETCOS/Sinkhole-Attack-in-DSR-v13.3/archive/refs/heads/main.zip>

Follow the instructions specified in the following link to download and setup the Project in NetSim:

<https://support.tetcos.com/en/support/solutions/articles/14000128666-downloading-and-setting-up-netsim-file-exchange-projects>

Introduction:

Sinkhole attack is one of the most severe attacks in wireless Ad hoc networks. In sinkhole Attack, a compromised node or malicious node advertises wrong routing information to pretend itself as a specific node and receives whole network traffic. After receiving the whole network traffic, it can either modify the packet information or drop them to make the network complicated. Sinkhole attacks affect the performance of Ad hoc network protocols such as the DSR protocol.

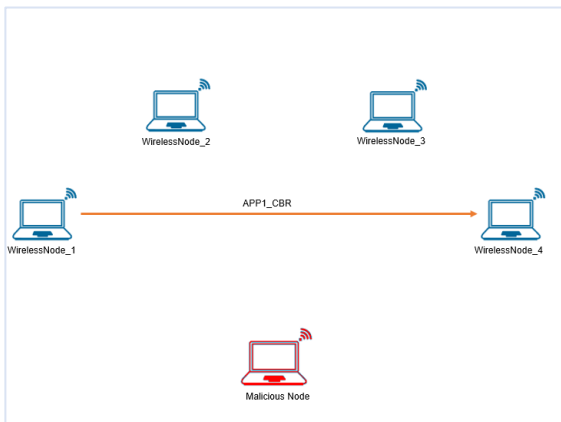


Figure 1: network configuration of how the traffic flow is configured.

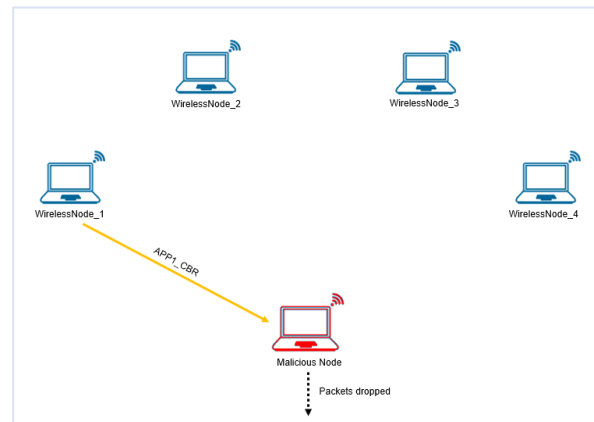


Figure 2: Network configuration of actual traffic flow along with the working of malicious node

Implementation in DSR:

- In DSR the source broadcasts the RREQ packet during Route Discovery.
- The destination on receiving the RREQ packet replies with an RREP packet containing the route to reach the destination.
- But Intermediate nodes can also send RREP packets to the source if they have a route to the destination in their route cache.

- Using this as an advantage the malicious node adds a fake route entry into its route cache with the destination node as its next hop.
- On receiving the RREQ packet from the source the malicious node sends a fake RREP packet with the fake route.
- The source node on receiving this packet observes this as a better route to the destination.
- All the Network Traffic is attracted towards the Sinkhole (Malicious Node) and it can either modify the packet Information or simply drop the packet.

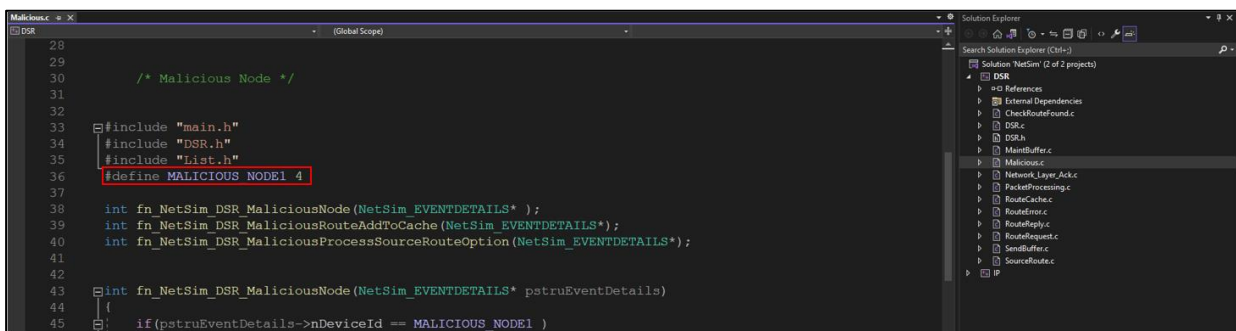
A file **malicious.c** is added to the DSR project which contains the following functions:

- **fn_NetSim_DSR_MaliciousNode();** //This function is used to identify whether a current device is malicious or not in-order to establish malicious behavior.
- **fn_NetSim_DSR_MaliciousRouteAddToCache();** //This function is used to add a fake route entry into the route cache of the malicious device with its next hop as the destination.
- **fn_NetSim_DSR_MaliciousProcessSourceRouteOption();** //This function is used to drop the received packets if the device is malicious, instead of forwarding the packet to the next hop.

You can set any device as malicious, and you can have more than one malicious node in a scenario. Device IDs of malicious nodes can be set inside the **fn_NetSim_DSR_MaliciousNode()** function.

Steps to simulate:

1. Open the Source codes in Visual Studio by going to Your work -> Source Code and click on the Open code in the NetSim Home Screen window.
2. Expand the DSR project and open the Malicious.c file and set the malicious node id.



```

28
29
30      /* Malicious Node */
31
32
33  #include "main.h"
34  #include "DSR.h"
35  #include "List.h"
36  #define MALICIOUS_NODE1 4
37
38  int fn_NetSim_DSR_MaliciousNode(NetSim_EVENTDETAILS* );
39  int fn_NetSim_DSR_MaliciousRouteAddToCache(NetSim_EVENTDETAILS*);
40  int fn_NetSim_DSR_MaliciousProcessSourceRouteOption(NetSim_EVENTDETAILS*);
41
42
43  int fn_NetSim_DSR_MaliciousNode(NetSim_EVENTDETAILS* pstruEventDetails)
44  {
45      if(pstruEventDetails->nDeviceId == MALICIOUS_NODE1 )
  
```

Figure 3: Set Malicious Node in malicious .c file

3. Now right-click on the DSR project and rebuild it.

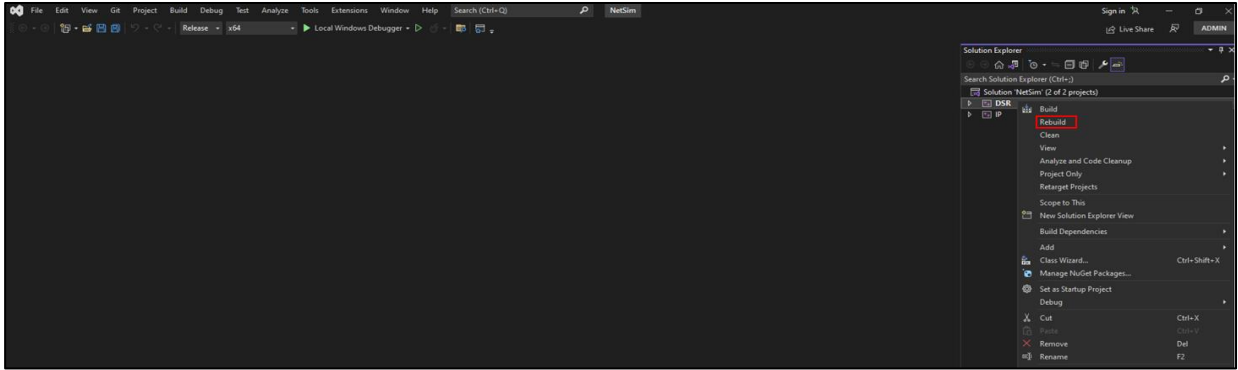


Figure 4: Screenshot of NetSim project source code in Visual Studio

4. Upon rebuilding, libDSR.dll will automatically get updated in the respective bin folder of the current workspace.

Example

1. The **SINK_HOLE_ATTACK_DSR_WorkSpace** comes with a sample network configuration that is already saved. To open this example, go to Your work in the home screen of NetSim and click on the **SINK_HOLE_ATTACK_DSR_Example** from the list of experiments.
2. The network consists of 6 Wired nodes with properties configured as shown below:

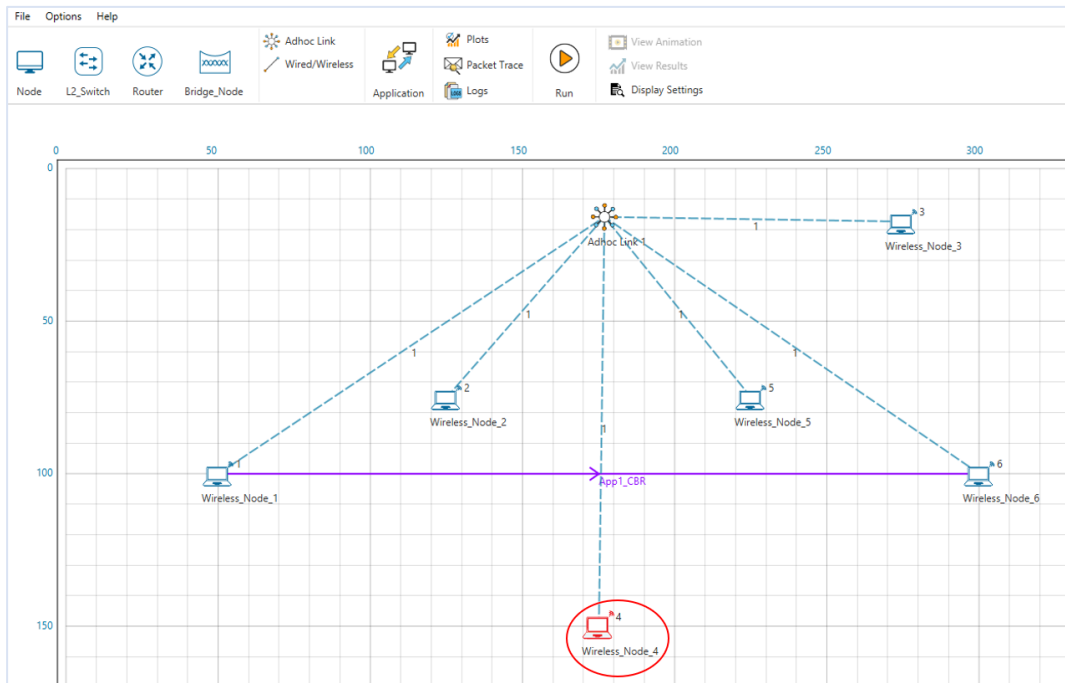


Figure 5: Network Topology

3. Application properties

Application Properties	
Source ID	1
Destination ID	6

Table 1: Application Properties

4. In Ad-hoc link Set the Channel Characteristics: **Pathloss only**, Path Loss Model: **Log Distance**, Path Loss Exponent: **3**

5. Run the Simulation for 100 seconds.

Results and discussion:

1. View the packet animation. You will find that the malicious node (Device id 4) gives Route Reply on receiving Route Request and attracts packets towards it. You will also find that the malicious node does not forward the packets that it receives.

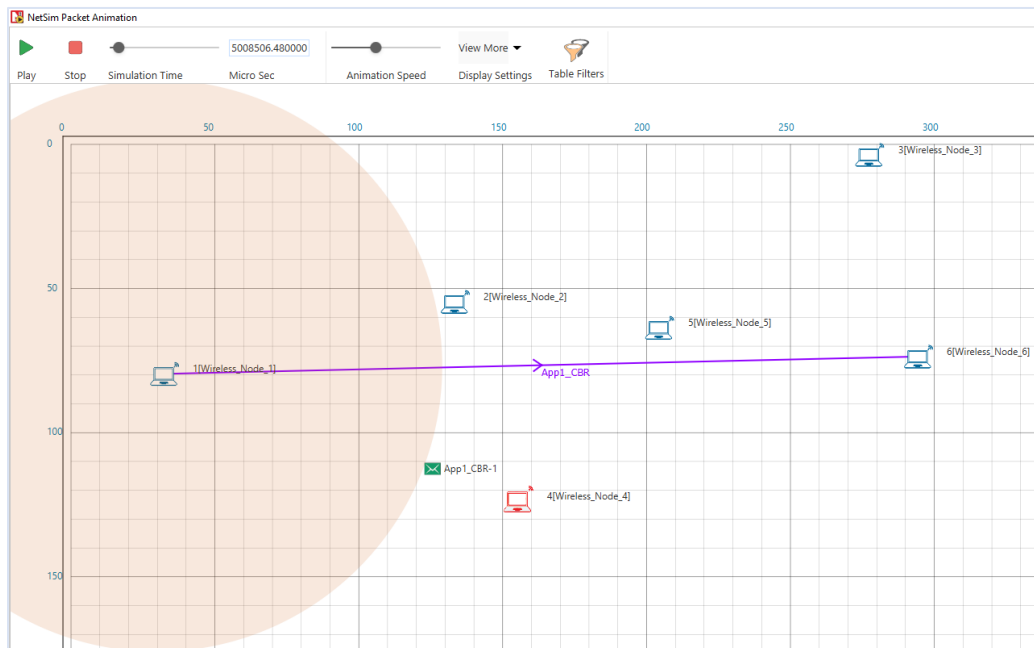


Figure 6: Animation Window

2. This will have a direct impact on the Application Throughput which can be observed in the Application Metrics table present in the NetSim Simulation Results window.

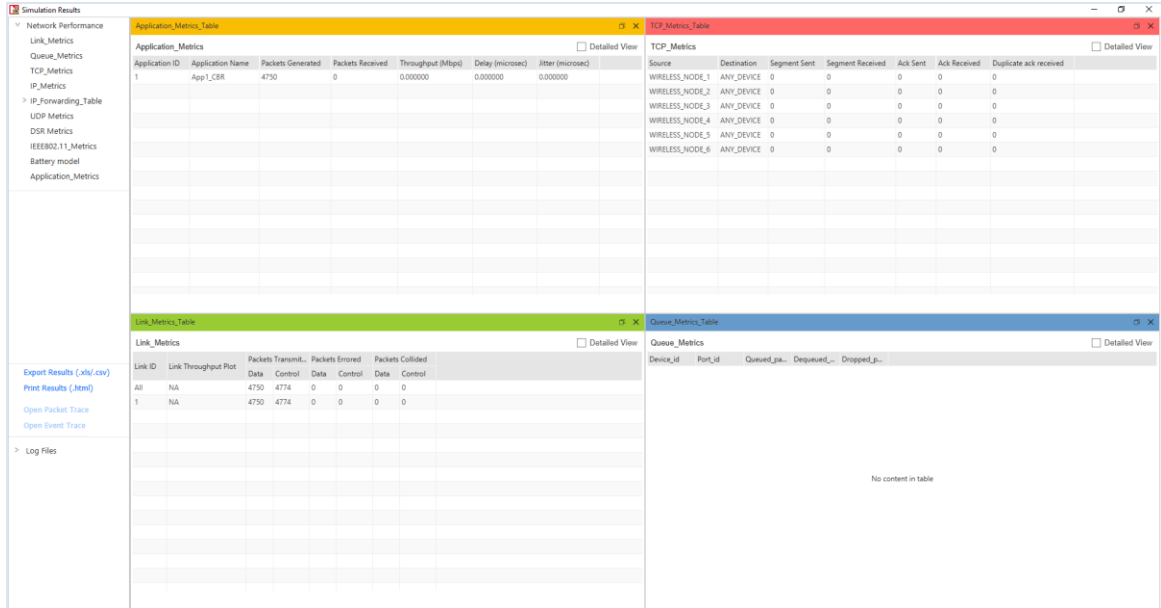


Figure 7: Result Dashboard