

Rebroadcasting packet in NetSim MANET/VANETs

Software: NetSim Standard v13.0 (32bit/ 64bit), Microsoft Visual Studio2019

Project Download Link:

https://github.com/NetSim-TETCOS/Probability-based-rebroadcast_v13.0/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>

Broadcasting

Broadcasting is the process of sending a message from one node to all other nodes in an ad-hoc network. It is a fundamental operation for communication in ad-hoc networks as it allows for the update of network information and route discovery at every node.

Rebroadcasting

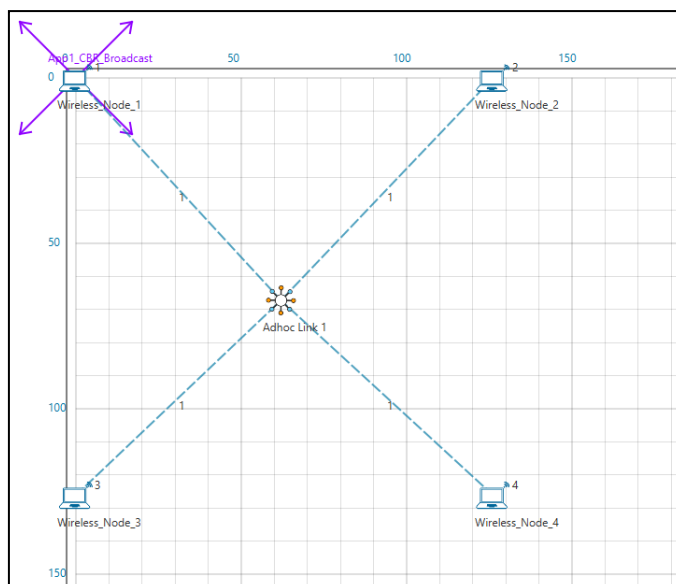
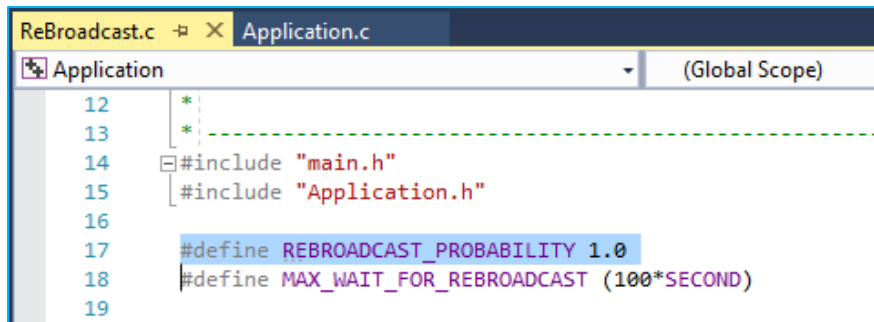


Figure 1: Network Scenario created in MANET

Wireless Node 1 initiates a broadcast message, and the message is received by nodes 2, 3 and 4. 2, 3 and 4 rebroadcast the message if they have not broadcasted that before. Furthermore, this implementation involves a `Rebroadcast_Probability` based on which the nodes resend the packets.

Probability-based rebroadcasting - The decision of rebroadcasting is based upon a random probability. This probability may be as simple as flipping a coin or it may be very complex involving probabilities which include parameters such as node density, duplicate packets received, battery power or a particular nodes participation within the network etc. Users can change the `Rebroadcast_Probability` macros present in `Rebroadcast.c` file as shown below:



```

12  *
13  *
14  #include "main.h"
15  #include "Application.h"
16
17  #define REBROADCAST_PROBABILITY 1.0
18  #define MAX_WAIT_FOR_REBROADCAST (100*SECOND)
19

```

Figure 2: Rebroadcast Probability

Rebroadcasting in NetSim

To implement this project in NetSim, we have created an additional Rebroadcast.c file inside Application project. The file contains the following functions:

- void rebroadcast_packet(); //This function is used to rebroadcast the packet.
- static bool isRebroadcastAllowed(); //This function is used to check whether rebroadcasting is allowed or not.
- void rebroadcast_add_packet_to_info(); //This function is used to add the packet to rebroadcast list.
- static void cleanup_broadcast_info();//This function is used to clean the broadcast information.

Steps to simulate

- Open the Source codes in Visual Studio by going to Your work-> Workspace Options and Clicking on Open code button in NetSim Home Screen window.
- Right click on Solution in Solution Explorer and select 'Rebuild solution.

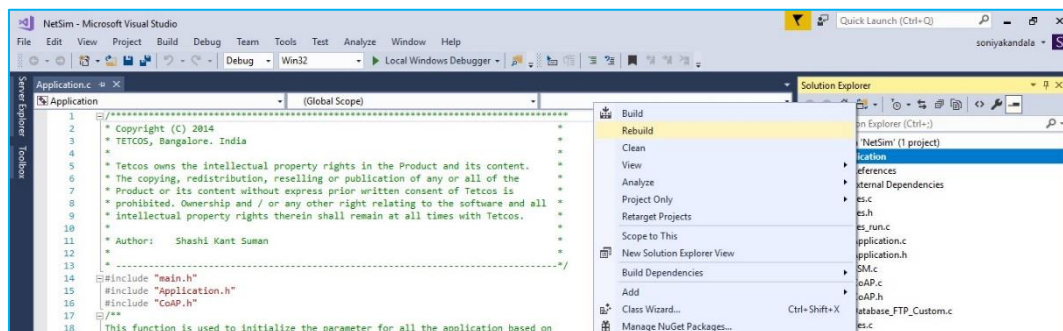


Figure 3: Screen shot of NetSim project source code in Visual Studio

- Upon rebuilding, **libApplication.dll** will automatically get updated in the respective bin folder of the current workspace.

Example

1. The Workspapce_Rebroadcasting comes with a sample network configuration that are already saved. To open this example, go to Your work in the Home screen of NetSim and click on the **Rebroadcasting_VANET_Example/Rebroadcasting_MANET_Example** from the list of experiments.
2. Run the simulation for 100 seconds.

VANET SCENARIO

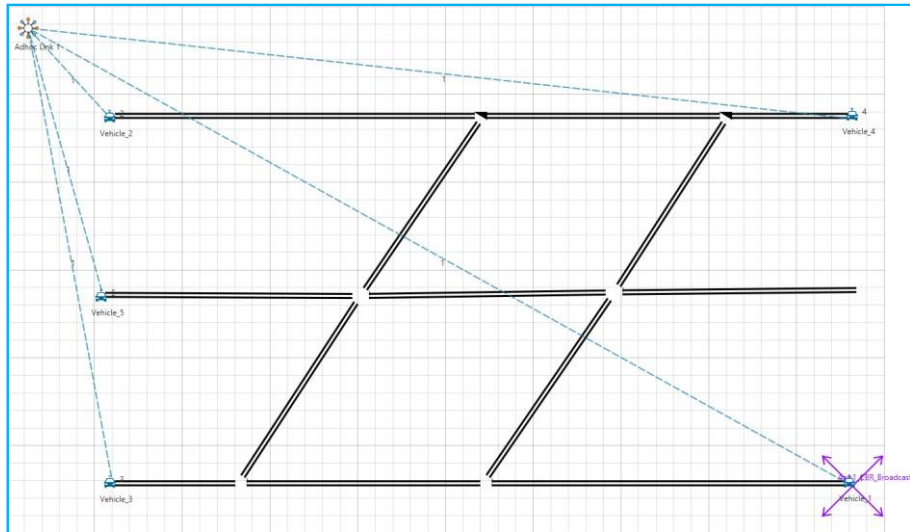


Figure 4: Network Scenario created in VANET

Results and discussion

- In the above scenario, Vehicle-1 is broadcasting the packet and it is received by the Vehicles 2, 3 and 4. Then Vehicles 2, 3, and 4 will rebroadcast the same packet based on the probability value in Rebroadcast.c file.
- After simulation, open Packet Trace and filter Packet_Id to '1' or any other id and observe that the nodes other than source are rebroadcasting the same packet.

	A	B	C	D	E	F	G	H
1	PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID
2	1	0	CBR	App1_CBR	NODE-1	Broadcast-0	NODE-1	NODE-2
3	1	0	CBR	App1_CBR	NODE-1	Broadcast-0	NODE-1	NODE-3
4	1	0	CBR	App1_CBR	NODE-1	Broadcast-0	NODE-1	NODE-4
5	1	0	CBR	App1_CBR	NODE-2	Broadcast-0	NODE-2	NODE-1
6	1	0	CBR	App1_CBR	NODE-2	Broadcast-0	NODE-2	NODE-3
7	1	0	CBR	App1_CBR	NODE-2	Broadcast-0	NODE-2	NODE-4
8	1	0	CBR	App1_CBR	NODE-3	Broadcast-0	NODE-3	NODE-1
9	1	0	CBR	App1_CBR	NODE-3	Broadcast-0	NODE-3	NODE-2
10	1	0	CBR	App1_CBR	NODE-3	Broadcast-0	NODE-3	NODE-4
20	1	0	CBR	App1_CBR	NODE-4	Broadcast-0	NODE-4	NODE-1
21	1	0	CBR	App1_CBR	NODE-4	Broadcast-0	NODE-4	NODE-2
22	1	0	CBR	App1_CBR	NODE-4	Broadcast-0	NODE-4	NODE-3

Figure 5: NetSim Packet Trace

- Note that Users SHOULD NOT use the performance metrics provided at the end of simulation but should rather calculate the network performance metrics from the packet trace.
- Users can also create their own network scenarios in **Single MANET/VANET** and run the simulation.

Appendix: NetSim source code modifications

Changes to int fn_NetSim_Application_Run() function in the APPLICATION_IN_EVENT, in Application.c file, within Application project

```
/*This is used to generate next broadcast packet if the current device is present in the source list*/

_declspec (dllexport) int fn_NetSim_Application_Run()
{
switch(pstruEventDetails->nEventType)
{
case APPLICATION_OUT_EVENT:
handle_app_out();
break;
case APPLICATION_IN_EVENT:
{
NetSim_PACKET* pstruPacket=pstruEventDetails->pPacket;
if(pstruPacket->nPacketType != PacketType_Control && pstruPacket->pstruAppData-
>nApplicationId &&
pstruPacket->nControlDataType/100 != PROTOCOL_APPLICATION)
{
ptrAPPLICATION_INFO pstruappinfo;
fnValidatePacket(pstruPacket);
pstruappinfo=applicationInfo[pstruPacket->pstruAppData->nApplicationId-1];
pstruPacket->pstruAppData->dEndTime = pstruEventDetails->dEventTime;
fn_NetSim_Application_Plot(pstruPacket);
#ifdef REBROADCAST
if (pstruappinfo->sourceList[0] == pstruPacket->nSourceId)
#endif
appmetrics_dest_add(pstruappinfo, pstruPacket, pstruEventDetails->nDeviceId);
if(pstruappinfo->nAppType==TRAFFIC_PEER_TO_PEER && pstruPacket->pstruAppData-
>nAppEndFlag==1)
{
fn_NetSim_Application_P2P_MarkReceivedPacket(pstruappinfo,pstruPacket);
fn_NetSim_Application_P2P_SendNextPiece(pstruappinfo,get_first_dest_from_packet(pstruPacket
),pstruEventDetails->dEventTime);
}
}
}
}
```

Changes to handle_app_out() function, in APP_OUT.c file, within Application project

```
/*The code checks if the destination is '0' i.e., Broadcast packet, then it adds the packet to
rebroadcast list*/
```

```
//Fragment the packet
```

```
int nSegmentCount = 0;
```

```
double segmentsize = fn_NetSim_Stack_GetMSS(pstruPacket);
```

```
nSegmentCount = fn_NetSim_Stack_FragmentPacket(pstruPacket,
(int)fn_NetSim_Stack_GetMSS(pstruPacket));
```

```

//add rebroadcast
#ifdef REBROADCAST
if (applInfo->sourceList[0] == pstruEventDetails->nDeviceId)
#endif

set_app_end_and_generate_next_packet(pstruPacket, otherDetails, destCount, dest);

//Add the dummy payload to packet
fn_NetSim_Add_DummyPayload(pstruPacket, applInfo);
#ifdef REBROADCAST
if (applInfo->sourceList[0] == pstruEventDetails->nDeviceId)
#endif

appmetrics_src_add(applInfo, pstruPacket);
appout_send_packet(s, applInfo, pstruPacket, nDeviceId);
#ifdef REBROADCAST
if (!dest[0])
rebroadcast_add_packet_to_info(pstruPacket, pstruEventDetails->dEventTime);
#endif // REBROADCAST
}

```

Changes to int fn_NetSim_Application_Run()function in the APPLICATION_IN_EVENT, in Application.c file, within Application project

/* It checks whether the destination is '0' or not. If it is '0', then it rebroadcasts the packet or else deletes the packet.*/

```

#ifdef REBROADCAST
if (pstruappinfo->sourceList[0] == pstruPacket->nSourceId)
#endif
appmetrics_dest_add(pstruappinfo, pstruPacket, pstruEventDetails->nDeviceId);
if(pstruappinfo->nAppType==TRAFFIC_PEER_TO_PEER && pstruPacket->pstruAppData->nAppEndFlag==1)
{
fn_NetSim_Application_P2P_MarkReceivedPacket(pstruappinfo,pstruPacket);
fn_NetSim_Application_P2P_SendNextPiece(pstruappinfo,get_first_dest_from_packet(pstruPacket),pstruEventDetails->dEventTime);
}
if(pstruappinfo->nAppType == TRAFFIC_EMULATION && pstruPacket->szPayload)
{
fn_NetSim_Dispatch_to_emulator(pstruPacket);
}
if (pstruappinfo->nAppType == TRAFFIC_BSM_APP)
{

```

```

process_saej2735_packet(pstruPacket);
}
#ifdef REBROADCAST
UINT destCount;
NETSIM_ID* dest = get_dest_from_packet(pstruPacket, &destCount);
if (!dest[0])
{
rebroadcast_packet(pstruPacket,
pstruEventDetails->nDeviceId,
pstruEventDetails->dEventTime);
}
else
{
#ifdef REBROADCAST
//Delete the packet
fn_NetSim_Packet_FreePacket(pstruPacket);
//add
#endif
}
#endif
}

```

Added the following function declarations in Application.h file, within Application project

```

int fn_NetSim_Add_DummyPayload(Netsim_PACKET* packet, ptrAPPLICATION_INFO);

//Encryption
char xor_encrypt(char ch, long key);
int aes256(char* str, int* len);
int des(char* buf, int* len);

//Application event handler
void handle_app_out();
#define REBROADCAST
void rebroadcast_add_packet_to_info(Netsim_PACKET* packet, double time);
void rebroadcast_packet(Netsim_PACKET* packet, NETSIM_ID devId, double time);
#endif

```