Congestion Control AODV (CC-AODV)

Software Recommended: NetSim Standard v12.1 (64 bit), Visual Studio 2019

Reference: Y. Mai, F. M. Rodriguez and N. Wang, "CC-ADOV: An effective multiple paths congestion control AODV," 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, 2018, pp. 1000-1004.

Follow the instructions specified in the following link to clone/download the project folder from GitHub using Visual Studio:
https://tetcos.freshdesk.com/support/solutions/articles/14000099351-how-to-clone-netsim-fileexchange-project-repositories-from-github-

Other tools such as GitHub Desktop, SVN Client, Sourcetree, Git from the command line, or any client you like to clone the Git repository.

Note: It is recommended not to download the project as an archive (compressed zip) to avoid incompatibility while importing workspaces into NetSim.

Secure URL for the GitHub repository:
https://github.com/NetSim-TETCOS/CC_AODV_Project_v12.1.git

Introduction

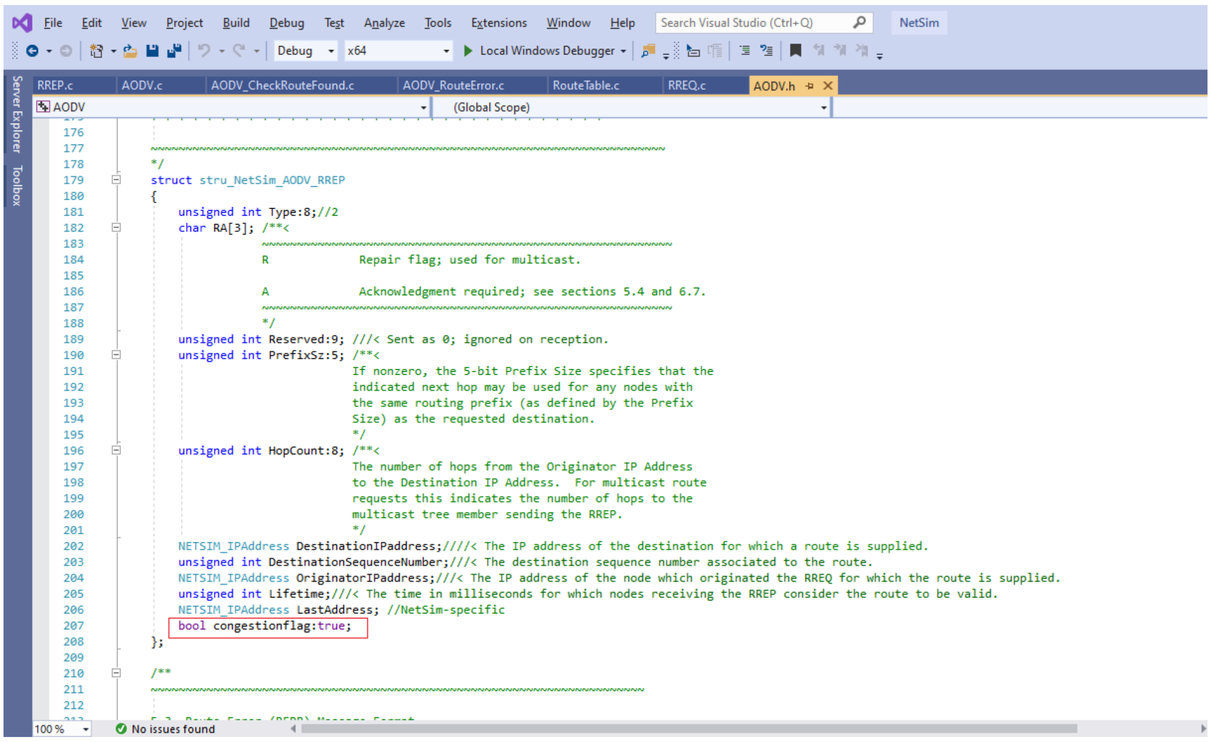
Ad hoc On-Demand Distance Vector (AODV) routing is one of the famous routing algorithms. Tremendous amounts of research on this protocol have been done to improve the performance. In this paper, a new control scheme, named congestion control AODV (CC-AODV), is proposed to manage the described routing condition. With this table entry, the package delivery rates are significantly increased while the package drop rate is decreased, however its implementation causes package overhead.

CC-ADOV aims to lower the performance degradation caused by the packets congestion while the data is delivered using AODV. Furthermore, CC-AODV determines a path for the data by using the congestion counter label. This is achieved by checking how stressed the current node is in a table, and once the RREP package is generated and transmitted through the nodes, the congestion counter adds one to the counter. The process of CC-AODV explains how to establish the route. First, the source node performs a flooding broadcast RREQ package in the entire network. When RREQ package arrives to the intermediate node, the router checks the congestion counter whether it is less than a certain predetermined value. If the comparison yields less than the counter, the routing table updates and forwarding to next router; otherwise, the router drops the RREQ package. Once the RREQ arrives to the corresponding destination, the RREP is generated by the router. In CC-AODV, the congestion flag is added to the RREP header. There are two cases of which a RREP is generated corresponding to a RREQ. One is from the source node to establish the route and the other is from the neighbour nodes to maintain the route. When the destination node receives the RREQ from the source node, it generates the RREP with the congestion flag set to true. While the RREP unicast back to the corresponding source node, passing by the intermediate node, the router checks the congestion flag. If it is true, the counter increases; otherwise, the counter keeps the same. Then, the router updates the routing information.

Procedure to implement CC-AODV in NetSim:

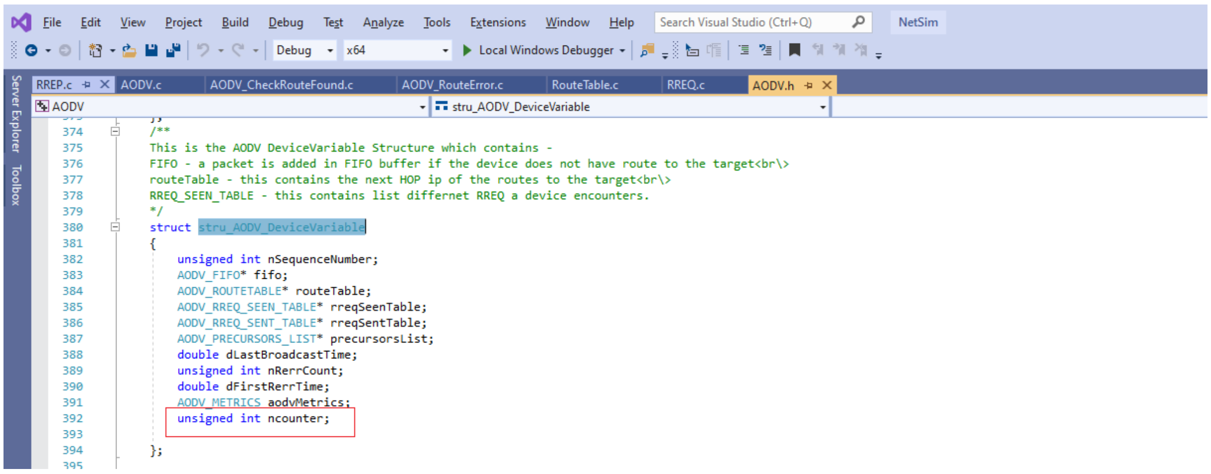
In order to implement CC-AODV following code modification done in AODV Protocol

1. The RREP structure stru_NetSim_AODV_RREP is defined in AODV.h has been modified to include a Congestion flag for implementing CC-AODV



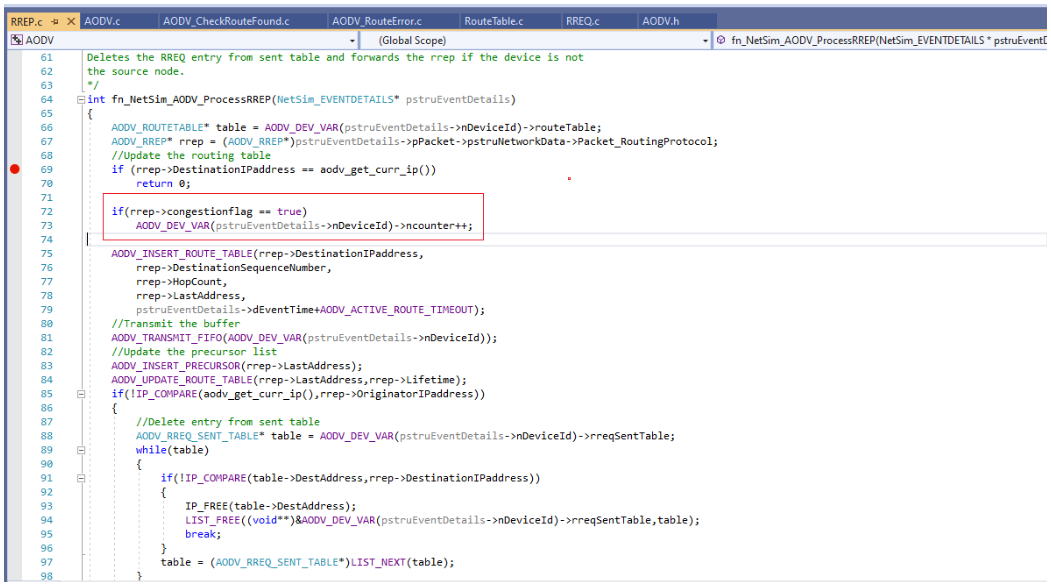
```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
*/
struct stru_NetSim_AODV_RREP
{
    unsigned int Type:8;//2
    char RA[3]; /**<
                ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                R             Repair flag; used for multicast.

                A             Acknowledgment required; see sections 5.4 and 6.7.
                ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                */
    unsigned int Reserved:9; ///< Sent as 0; ignored on reception.
    unsigned int PrefixSz:5; /**<
                             If nonzero, the 5-bit Prefix Size specifies that the
                             indicated next hop may be used for any nodes with
                             the same routing prefix (as defined by the Prefix
                             Size) as the requested destination.
                             */
    unsigned int HopCount:8; /**<
                             The number of hops from the Originator IP Address
                             to the Destination IP Address.  For multicast route
                             requests this indicates the number of hops to the
                             multicast tree member sending the RREP.
                             */
    NETSIM_IPAddress DestinationIPaddress;////< The IP address of the destination for which a route is supplied.
    unsigned int DestinationSequenceNumber;///< The destination sequence number associated to the route.
    NETSIM_IPAddress OriginatorIPaddress;///< The IP address of the node which originated the RREQ for which the route is supplied.
    unsigned int Lifetime;///< The time in milliseconds for which nodes receiving the RREP consider the route to be valid.
    NETSIM_IPAddress LastAddress; //NetSim-specific
    bool congestionflag:true;
};

/**
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

2. The DeviceVariable Structure stru_AODV_DeviceVariable is defined in AODV.h file has been modified to include a congestion counter for implementing CC-AODV



```
/**
This is the AODV DeviceVariable Structure which contains -
FIFO - a packet is added in FIFO buffer if the device does not have route to the target<br\>
routeTable - this contains the next HOP ip of the routes to the target<br\>
RREQ_SEEN_TABLE - this contains list differnet RREQ a device encounters.
*/
struct stru_AODV_DeviceVariable
{
    unsigned int nSequenceNumber;
    AODV_FIFO* fifo;
    AODV_ROUTETABLE* routeTable;
    AODV_RREQ_SEEN_TABLE* rreqSeenTable;
    AODV_RREQ_SENT_TABLE* rreqSentTable;
    AODV_PRECURSORS_LIST* precursorsList;
    double dLastBroadcastTime;
    unsigned int nRerrCount;
    double dFirstRerrTime;
    AODV_METRICS aodvMetrics;
    unsigned int ncounter;

};
```

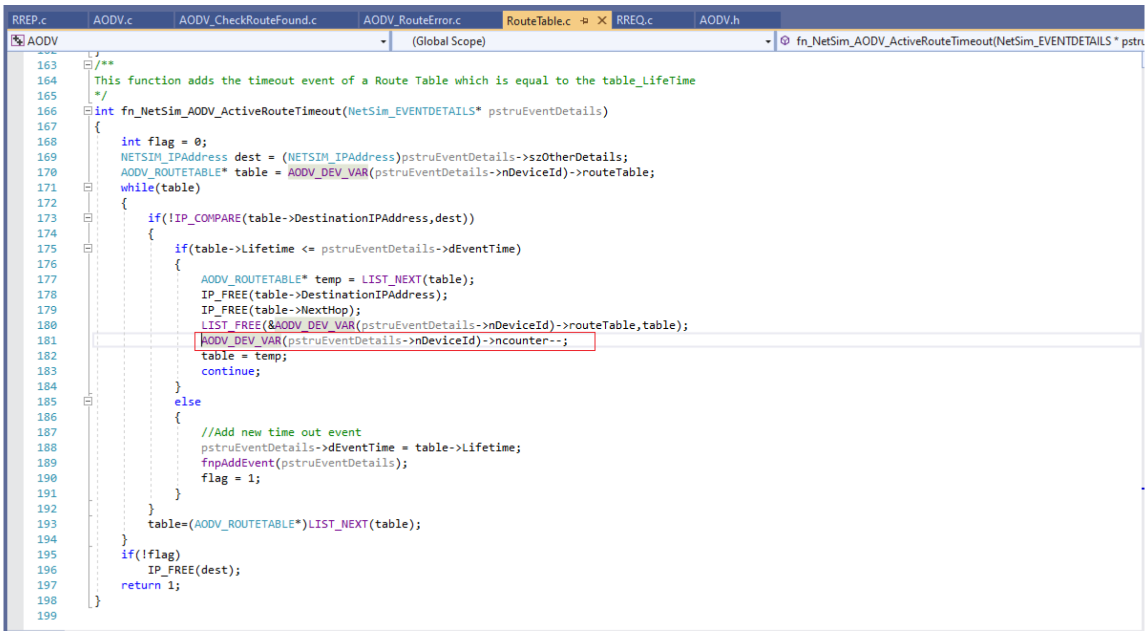
3. The source codes of functions in **RREP.c**, **RouteTable.c** and **AODV_RouteError.c** has been modified suitably to Increment, Decrement the congestion counter accordingly



RREP.c | AODV.c | AODV_CheckRouteFound.c | AODV_RouteError.c | RouteTable.c | RREQ.c | AODV.h

AODV | (Global Scope) | fn_NetSim_AODV_ProcessRREP(NetSim_EVENTDETAILS * pstruEventD

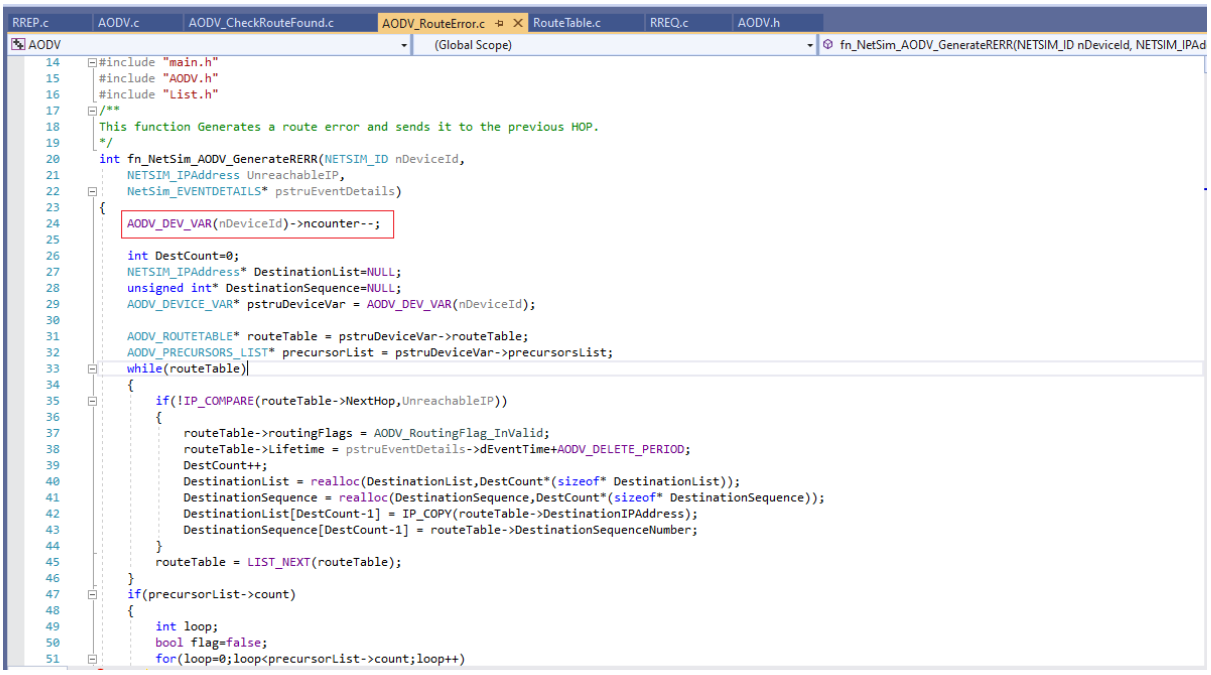
```c
Deletes the RREQ entry from sent table and forwards the rrep if the device is not
the source node.
*/
int fn_NetSim_AODV_ProcessRREP(NetSim_EVENTDETAILS* pstruEventDetails)
{
    AODV_ROUTETABLE* table = AODV_DEV_VAR(pstruEventDetails->nDeviceId)->routeTable;
    AODV_RREP* rrep = (AODV_RREP*)pstruEventDetails->pPacket->pstruNetworkData->Packet_RoutingProtocol;
    //Update the routing table
    if (rrep->DestinationIPaddress == aodv_get_curr_ip())
        return 0;

    if(rrep->congestionflag == true)
        AODV_DEV_VAR(pstruEventDetails->nDeviceId)->ncounter++;

    AODV_INSERT_ROUTE_TABLE(rrep->DestinationIPaddress,
        rrep->DestinationSequenceNumber,
        rrep->HopCount,
        rrep->LastAddress,
        pstruEventDetails->dEventTime+AODV_ACTIVE_ROUTE_TIMEOUT);
    //Transmit the buffer
    AODV_TRANSMIT_FIFO(AODV_DEV_VAR(pstruEventDetails->nDeviceId));
    //Update the precursor list
    AODV_INSERT_PRECURSOR(rrep->LastAddress);
    AODV_UPDATE_ROUTE_TABLE(rrep->LastAddress,rrep->Lifetime);
    if(!IP_COMPARE(aodv_get_curr_ip(),rrep->OriginatorIPaddress))
    {
        //Delete entry from sent table
        AODV_RREQ_SENT_TABLE* table = AODV_DEV_VAR(pstruEventDetails->nDeviceId)->rreqSentTable;
        while(table)
        {
            if(!IP_COMPARE(table->DestAddress,rrep->DestinationIPaddress))
            {
                IP_FREE(table->DestAddress);
                LIST_FREE((void**)&AODV_DEV_VAR(pstruEventDetails->nDeviceId)->rreqSentTable,table);
                break;
            }
            table = (AODV_RREQ_SENT_TABLE*)LIST_NEXT(table);
        }
```



RREP.c | AODV.c | AODV_CheckRouteFound.c | AODV_RouteError.c | RouteTable.c | RREQ.c | AODV.h

AODV | (Global Scope) | fn_NetSim_AODV_ActiveRouteTimeout(NetSim_EVENTDETAILS * pstru

```c
/**
This function adds the timeout event of a Route Table which is equal to the table_LifeTime
*/
int fn_NetSim_AODV_ActiveRouteTimeout(NetSim_EVENTDETAILS* pstruEventDetails)
{
    int flag = 0;
    NETSIM_IPAddress dest = (NETSIM_IPAddress)pstruEventDetails->szOtherDetails;
    AODV_ROUTETABLE* table = AODV_DEV_VAR(pstruEventDetails->nDeviceId)->routeTable;
    while(table)
    {
        if(!IP_COMPARE(table->DestinationIPAddress,dest))
        {
            if(table->Lifetime <= pstruEventDetails->dEventTime)
            {
                AODV_ROUTETABLE* temp = LIST_NEXT(table);
                IP_FREE(table->DestinationIPAddress);
                IP_FREE(table->NextHop);
                LIST_FREE(&AODV_DEV_VAR(pstruEventDetails->nDeviceId)->routeTable,table);
                AODV_DEV_VAR(pstruEventDetails->nDeviceId)->ncounter--;
                table = temp;
                continue;
            }
            else
            {
                //Add new time out event
                pstruEventDetails->dEventTime = table->Lifetime;
                fnpAddEvent(pstruEventDetails);
                flag = 1;
            }
        }
        table=(AODV_ROUTETABLE*)LIST_NEXT(table);
    }
    if(!flag)
        IP_FREE(dest);
    return 1;
}
```

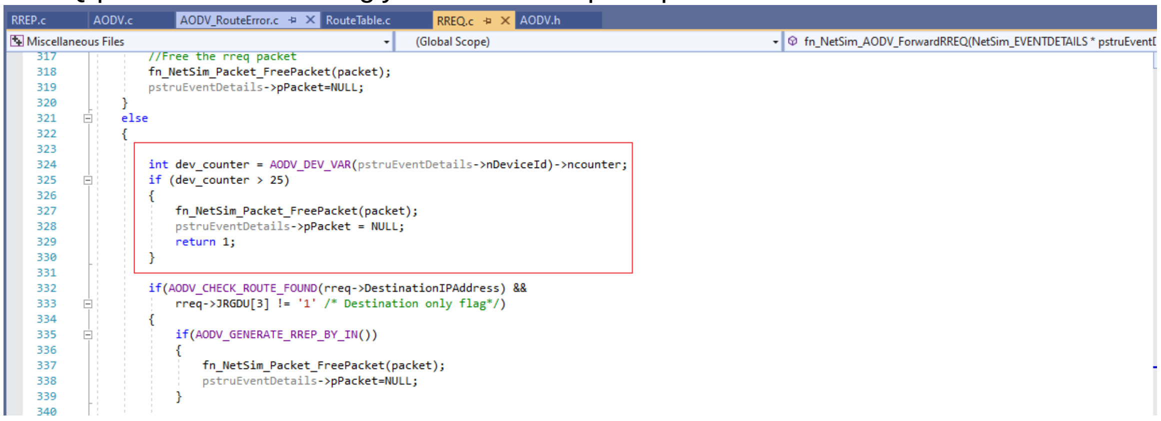


RREP.c | AODV.c | AODV_CheckRouteFound.c | AODV_RouteError.c | RouteTable.c | RREQ.c | AODV.h

AODV | (Global Scope) | fn_NetSim_AODV_GenerateRERR(NETSIM_ID nDeviceId, NETSIM_IPAd

```c
#include "main.h"
#include "AODV.h"
#include "List.h"
/**
This function Generates a route error and sends it to the previous HOP.
*/
int fn_NetSim_AODV_GenerateRERR(NETSIM_ID nDeviceId,
    NETSIM_IPAddress UnreachableIP,
    NetSim_EVENTDETAILS* pstruEventDetails)
{
    AODV_DEV_VAR(nDeviceId)->ncounter--;

    int DestCount=0;
    NETSIM_IPAddress* DestinationList=NULL;
    unsigned int* DestinationSequence=NULL;
    AODV_DEVICE_VAR* pstruDeviceVar = AODV_DEV_VAR(nDeviceId);

    AODV_ROUTETABLE* routeTable = pstruDeviceVar->routeTable;
    AODV_PRECURSORS_LIST* precursorList = pstruDeviceVar->precursorsList;
    while(routeTable)
    {
        if(!IP_COMPARE(routeTable->NextHop,UnreachableIP))
        {
            routeTable->routingFlags = AODV_RoutingFlag_InValid;
            routeTable->Lifetime = pstruEventDetails->dEventTime+AODV_DELETE_PERIOD;
            DestCount++;
            DestinationList = realloc(DestinationList,DestCount*(sizeof* DestinationList));
            DestinationSequence = realloc(DestinationSequence,DestCount*(sizeof* DestinationSequence));
            DestinationList[DestCount-1] = IP_COPY(routeTable->DestinationIPAddress);
            DestinationSequence[DestCount-1] = routeTable->DestinationSequenceNumber;
        }
        routeTable = LIST_NEXT(routeTable);
    }
    if(precursorList->count)
    {
        int loop;
        bool flag=false;
        for(loop=0;loop<precursorList->count;loop++)
```

4. The source codes and functions related to Route request are defined in the file **RREQ.c.** The **fn_NetSim_AODV_ProcessRREQ()** function that is part of this file has been modified suitably to check the value of the congestion counter in the received RREQ packet and accordingly forward or drop the packet



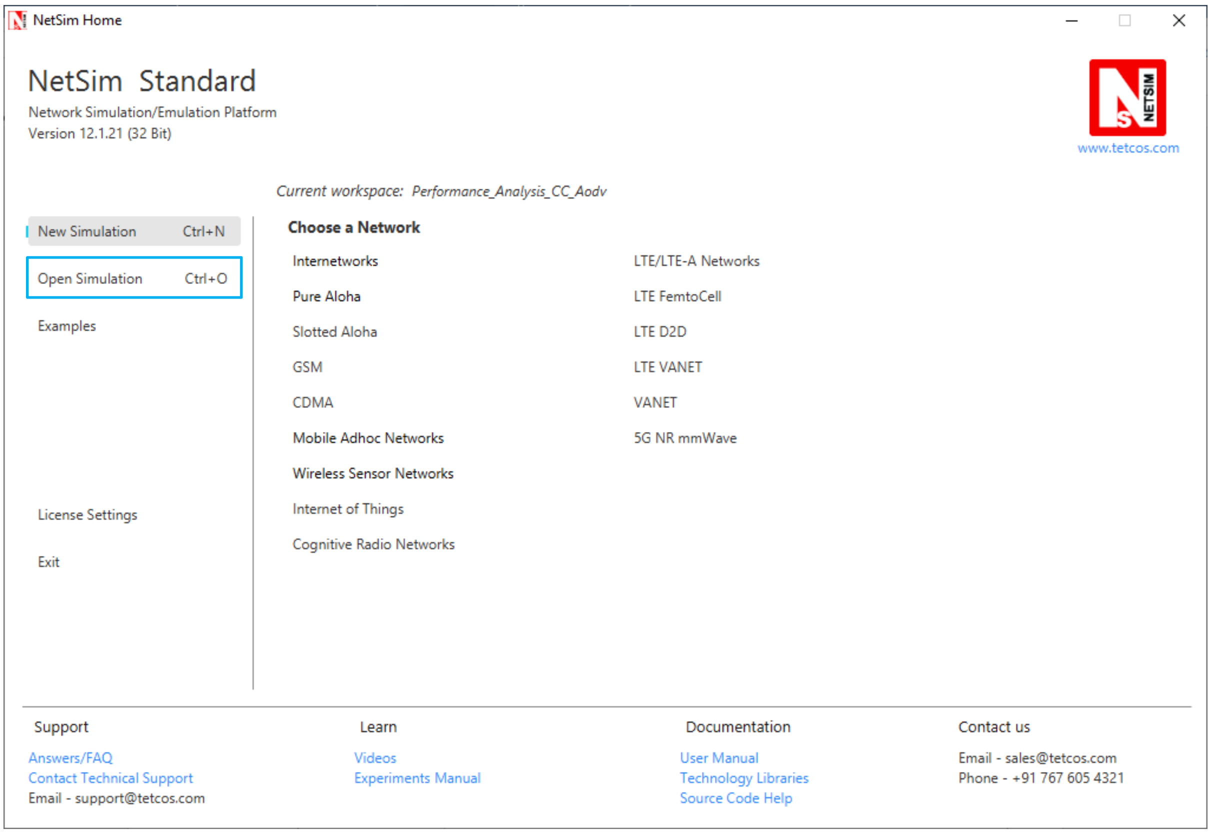
RREP.c | AODV.c | AODV_RouteError.c | RouteTable.c | RREQ.c | AODV.h

Miscellaneous Files | (Global Scope) | fn_NetSim_AODV_ForwardRREQ(NetSim_EVENTDETAILS * pstruEventD

```c
                //Free the rreq packet
                fn_NetSim_Packet_FreePacket(packet);
                pstruEventDetails->pPacket=NULL;
            }
            else
            {

                int dev_counter = AODV_DEV_VAR(pstruEventDetails->nDeviceId)->ncounter;
                if (dev_counter > 25)
                {
                    fn_NetSim_Packet_FreePacket(packet);
                    pstruEventDetails->pPacket = NULL;
                    return 1;
                }

                if(AODV_CHECK_ROUTE_FOUND(rreq->DestinationIPAddress) &&
                    rreq->JRGDU[3] != '1' /* Destination only flag*/)
                {
                    if(AODV_GENERATE_RREP_BY_IN())
                    {
                        fn_NetSim_Packet_FreePacket(packet);
                        pstruEventDetails->pPacket=NULL;
                    }

```

Steps:

1. After you unzip the downloaded project folder, Open NetSim Home Page click on **Open Simulation**


NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 12.1.21 (32 Bit)

www.tetcos.com

Current workspace: Performance_Analysis_CC_Aodv

New Simulation Ctrl+N
Open Simulation Ctrl+O
Examples
License Settings
Exit

Choose a Network

Internetworks	LTE/LTE-A Networks
Pure Aloha	LTE FemtoCell
Slotted Aloha	LTE D2D
GSM	LTE VANET
CDMA	VANET
Mobile Adhoc Networks	5G NR mmWave
Wireless Sensor Networks	
Internet of Things	
Cognitive Radio Networks	

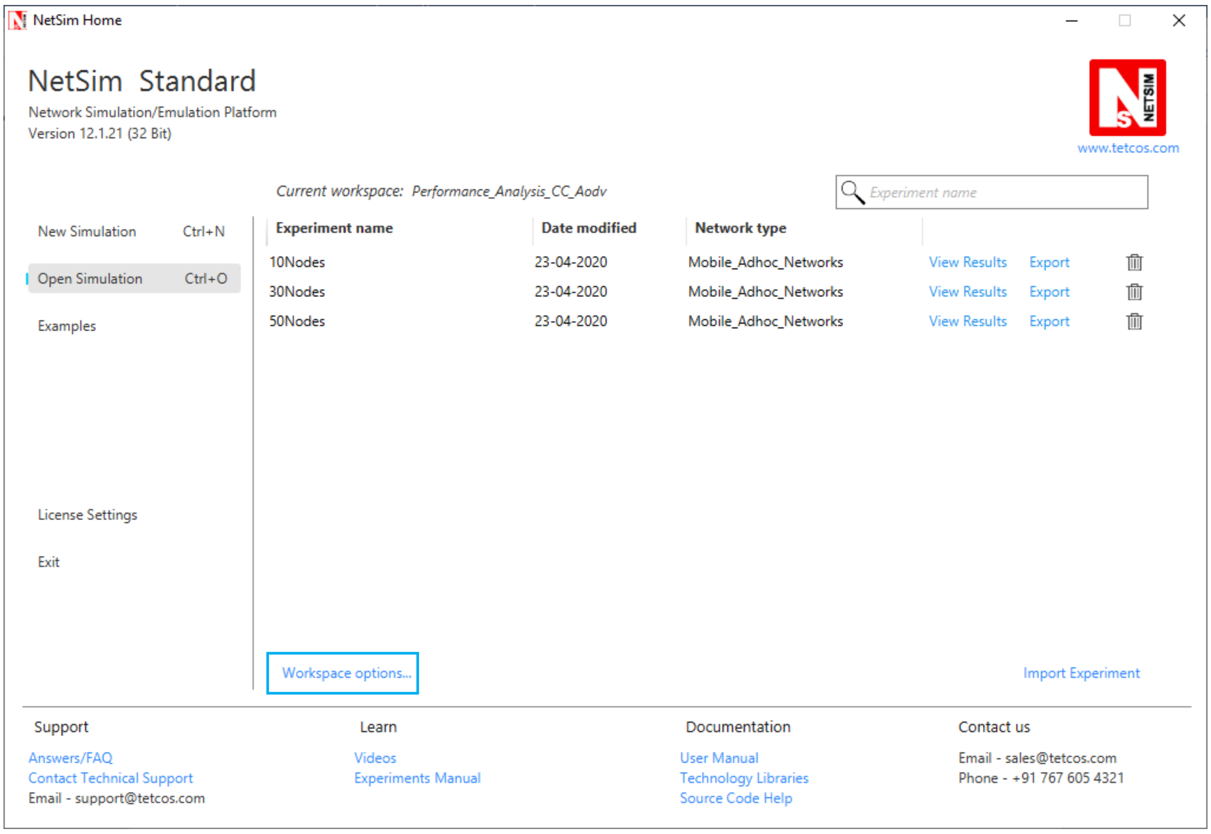
Support
Answers/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiments Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact us
Email - sales@tetcos.com
Phone - +91 767 605 4321

2. Click on **Workspace options**


NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 12.1.21 (32 Bit)

www.tetcos.com

Current workspace: Performance_Analysis_CC_Aodv

Experiment name

New Simulation Ctrl+N
Open Simulation Ctrl+O
Examples
License Settings
Exit

Experiment name	Date modified	Network type			
10Nodes	23-04-2020	Mobile_Adhoc_Networks	View Results	Export	🗑
30Nodes	23-04-2020	Mobile_Adhoc_Networks	View Results	Export	🗑
50Nodes	23-04-2020	Mobile_Adhoc_Networks	View Results	Export	🗑

Workspace options...
Import Experiment

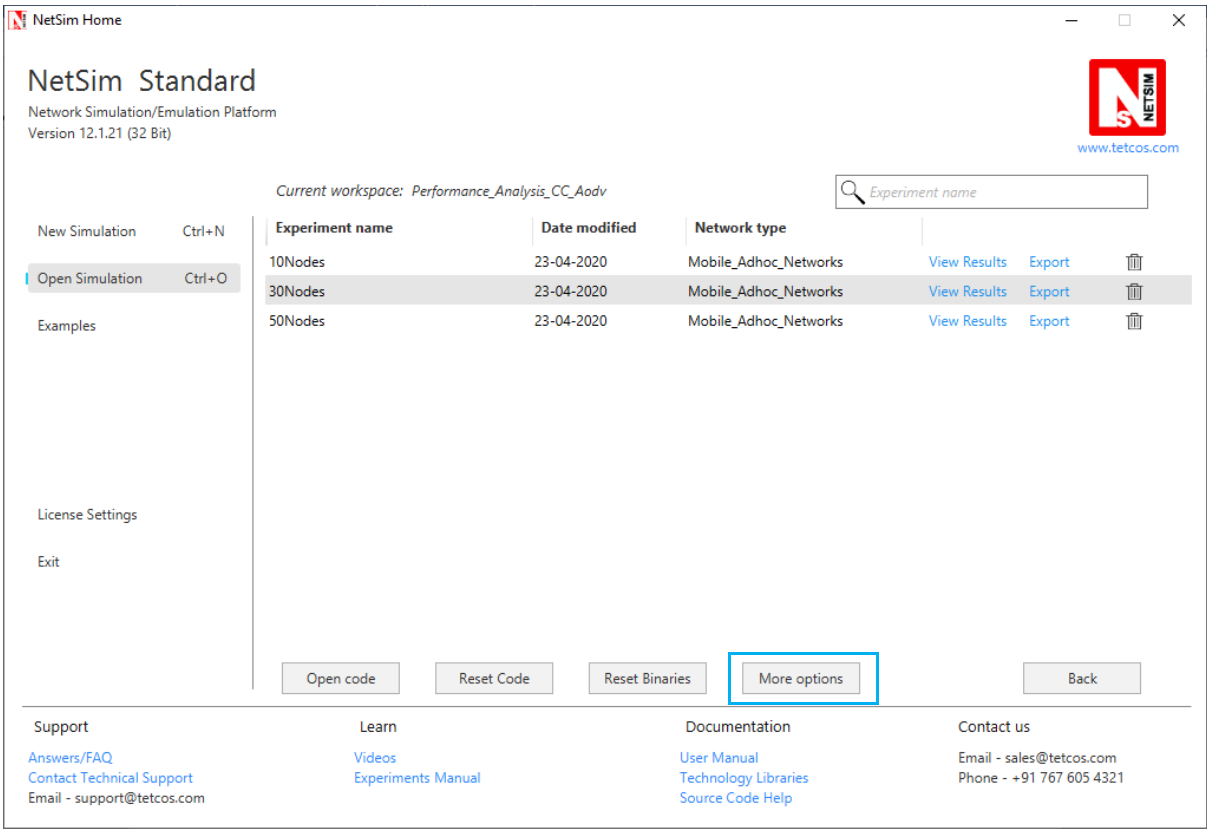
Support
Answers/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiments Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact us
Email - sales@tetcos.com
Phone - +91 767 605 4321

3. Click on **More Options,**


NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 12.1.21 (32 Bit)

www.tetcos.com

Current workspace: Performance_Analysis_CC_Aodv

Experiment name

Experiment name	Date modified	Network type			
10Nodes	23-04-2020	Mobile_Adhoc_Networks	View Results	Export	🗑
30Nodes	23-04-2020	Mobile_Adhoc_Networks	View Results	Export	🗑
50Nodes	23-04-2020	Mobile_Adhoc_Networks	View Results	Export	🗑

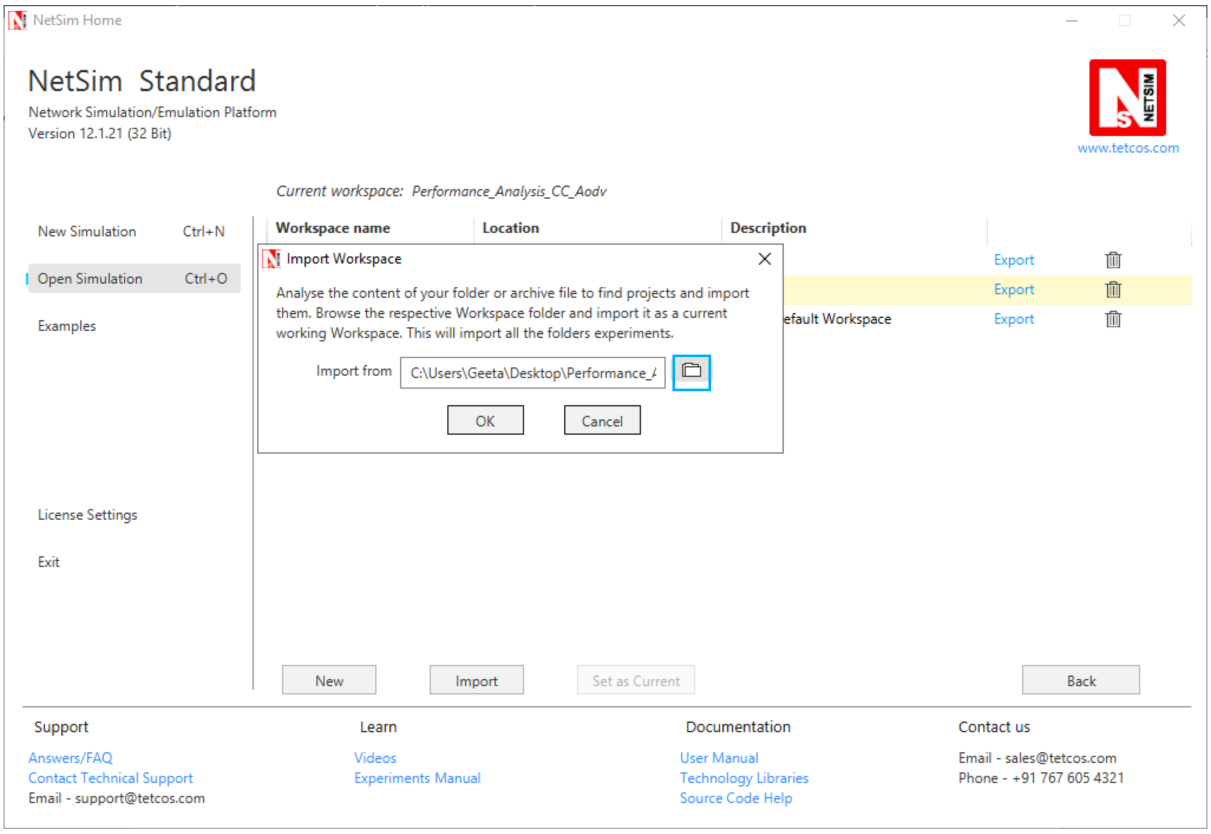
Support
Answers/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiments Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact us
Email - sales@tetcos.com
Phone - +91 767 605 4321

4. Click on **Import**, browse the extracted folder and go into the WorkSpace_Performance_Analysis_CC_AODV directory, click on select folder and then click on OK.


NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 12.1.21 (32 Bit)

www.tetcos.com

Current workspace: Performance_Analysis_CC_Aodv

Workspace name	Location	Description		
			Export	🗑
			Export	🗑
		efault Workspace	Export	🗑

Import Workspace

Analyse the content of your folder or archive file to find projects and import them. Browse the respective Workspace folder and import it as a current working Workspace. This will import all the folders experiments.

Import from C:\Users\Geeta\Desktop\Performance_A

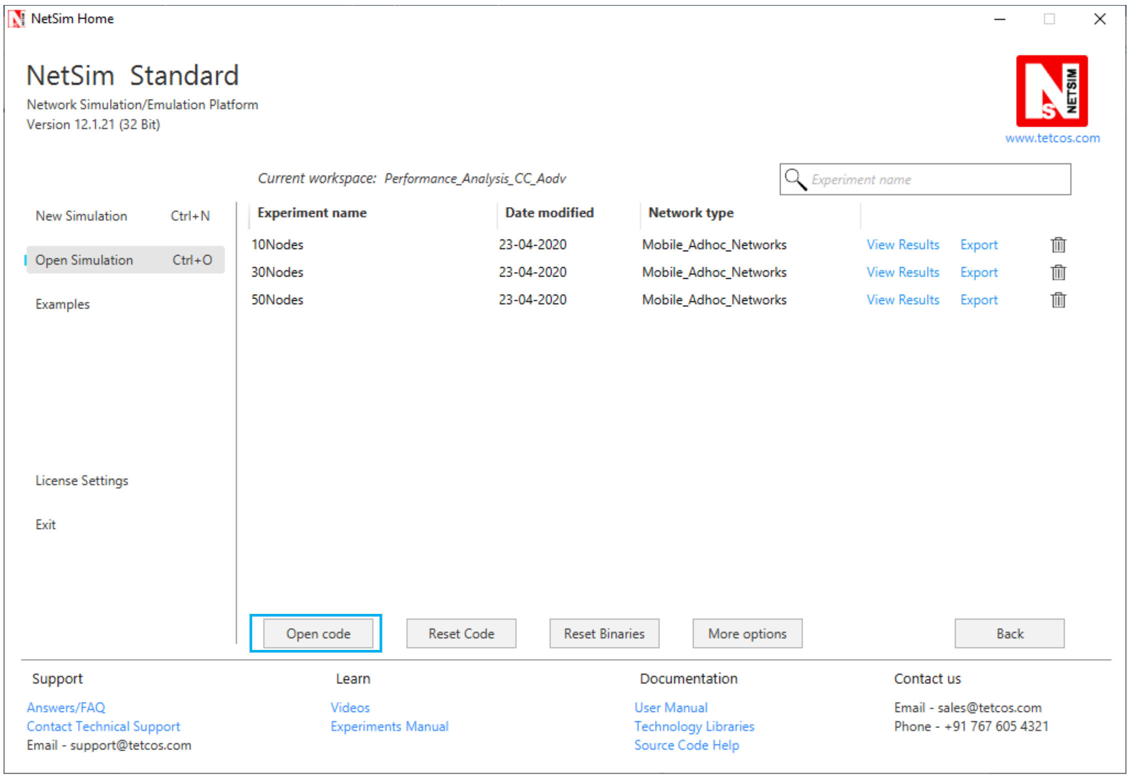
Support
Answers/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiments Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact us
Email - sales@tetcos.com
Phone - +91 767 605 4321

5. Go to home page, Click on **Open Simulation → Workspace options → Open code**



Current workspace: *Performance_Analysis_CC_Aodv*

Experiment name	Date modified	Network type		
10Nodes	23-04-2020	Mobile_Adhoc_Networks	View Results	Export
30Nodes	23-04-2020	Mobile_Adhoc_Networks	View Results	Export
50Nodes	23-04-2020	Mobile_Adhoc_Networks	View Results	Export

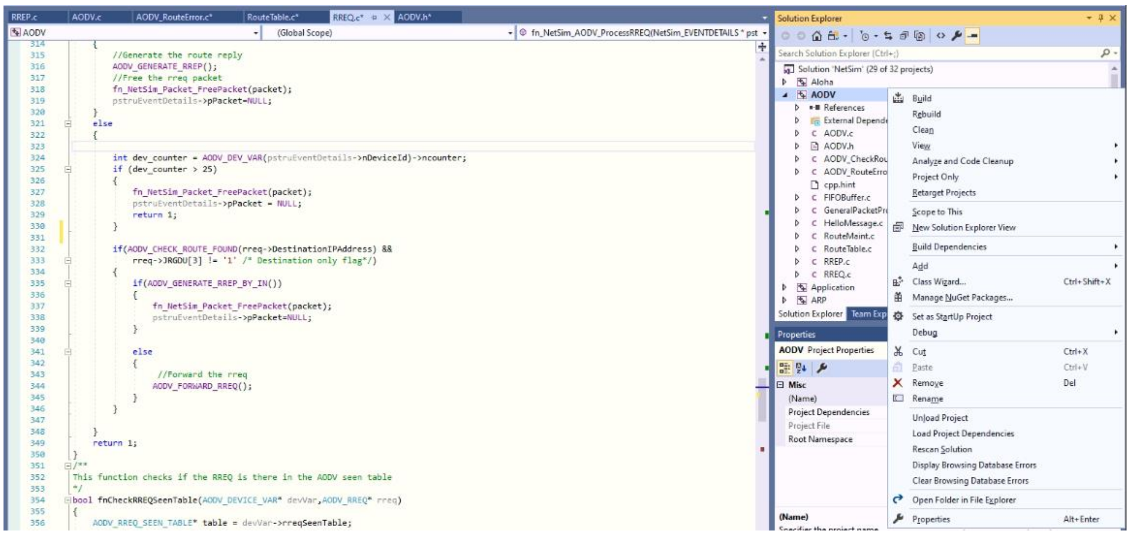
Support
Answers/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiments Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact us
Email - sales@tetcos.com
Phone - +91 767 605 4321

6. Right click on the AODV Project and select rebuild.

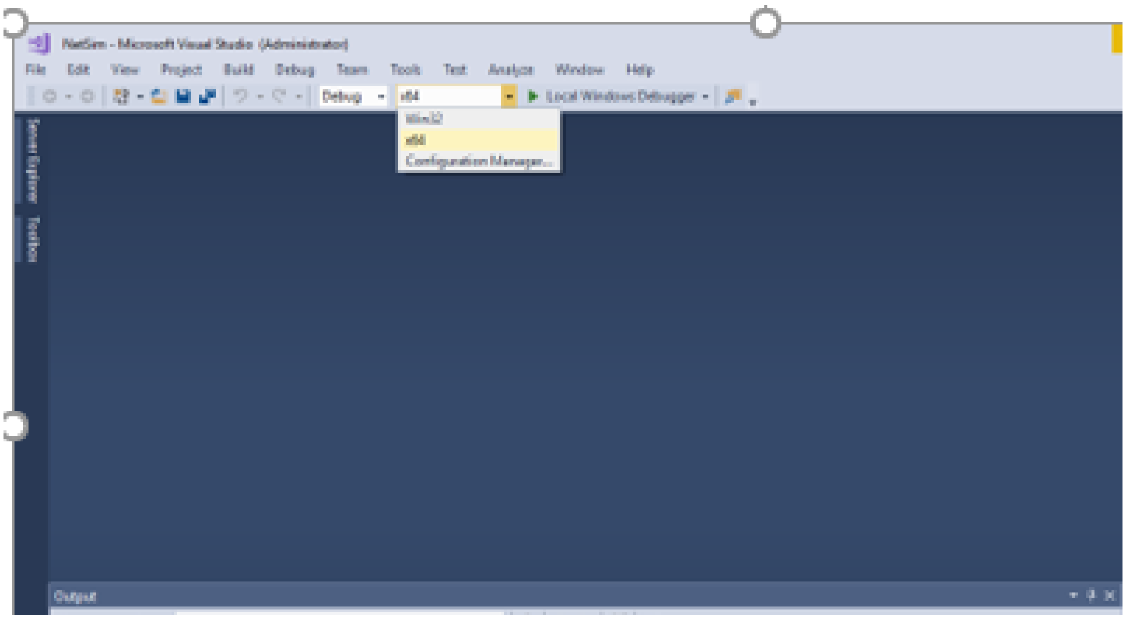


```
    {
        //Generate the route reply
        AODV_GENERATE_RREP();
        //Free the rreq packet
        fn_NetSim_Packet_FreePacket(packet);
        pstruEventDetails->pPacket=NULL;
    }
    else
    {

        int dev_counter = AODV_DEV_VAR(pstruEventDetails->nDeviceId)->ncounter;
        if (dev_counter > 25)
        {
            fn_NetSim_Packet_FreePacket(packet);
            pstruEventDetails->pPacket = NULL;
            return 1;
        }

        if(AODV_CHECK_ROUTE_FOUND(rreq->DestinationIPAddress) &&
            rreq->JRGDU[3] != '1' /* Destination only flag*/)
        {
            if(AODV_GENERATE_RREP_BY_IN())
            {
                fn_NetSim_Packet_FreePacket(packet);
                pstruEventDetails->pPacket=NULL;
            }

            else
            {
                 //Forward the rreq
                AODV_FORWARD_RREQ();
            }
        }

    }
    return 1;
}
/**
This function checks if the RREQ is there in the AODV seen table
*/
bool fnCheckRREQSeenTable(AODV_DEVICE_VAR* devVar,AODV_RREQ* rreq)
{
    AODV_RREQ_SEEN_TABLE* table = devVar->rreqSeenTable;
```

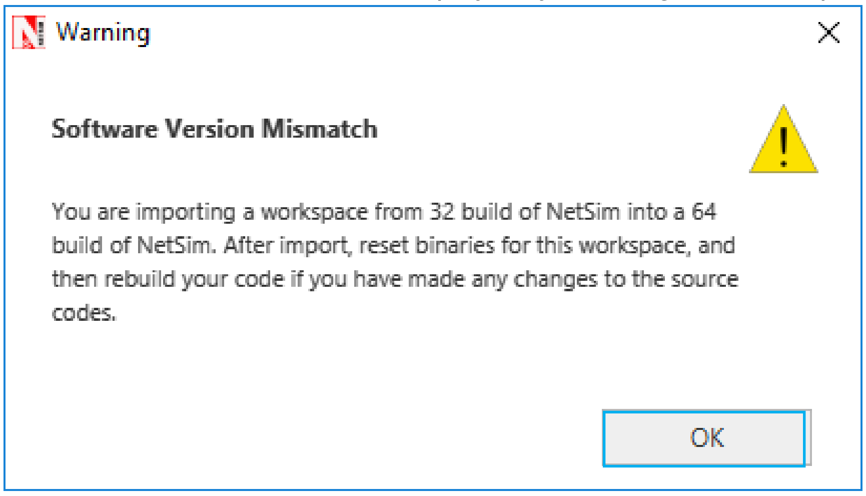
7. Upon rebuilding, **libAodv.dll** will automatically get updated in the respective bin folder of the current workspace.

Note:

- Based on whether you are using NetSim 32 bit or 64 bit setup you can configure Visual studio to build 32 bit or 64 bit Dll files respectively as shown below:


NetSim - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug
x64
Local Windows Debugger
Win32
x64
Configuration Manager...
Server Explorer
Toolbox
Output

- While importing the workspace, if the following warning message indicating Software Version Mismatch is displayed, you can ignore it and proceed.

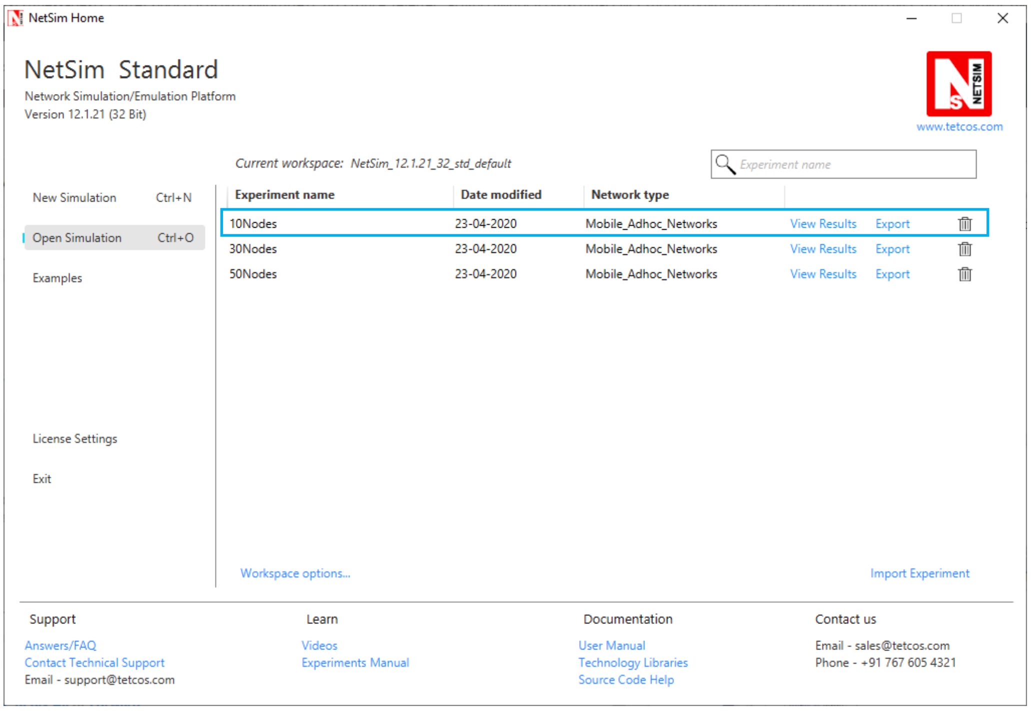


Warning

Software Version Mismatch

You are importing a workspace from 32 build of NetSim into a 64 build of NetSim. After import, reset binaries for this workspace, and then rebuild your code if you have made any changes to the source codes.

8. Go to NetSim home page, click on **Open Simulation**, Click on **10Nodes_Example**.


NetSim Home

NetSim Standard
Network Simulation/Emulation Platform
Version 12.1.21 (32 Bit)

www.tetcos.com

Current workspace: NetSim_12.1.21_32_std_default

New Simulation Ctrl+N
Open Simulation Ctrl+O
Examples
License Settings
Exit

Experiment name	Date modified	Network type			
10Nodes	23-04-2020	Mobile_Adhoc_Networks	View Results	Export	🗑
30Nodes	23-04-2020	Mobile_Adhoc_Networks	View Results	Export	🗑
50Nodes	23-04-2020	Mobile_Adhoc_Networks	View Results	Export	🗑

Workspace options... Import Experiment

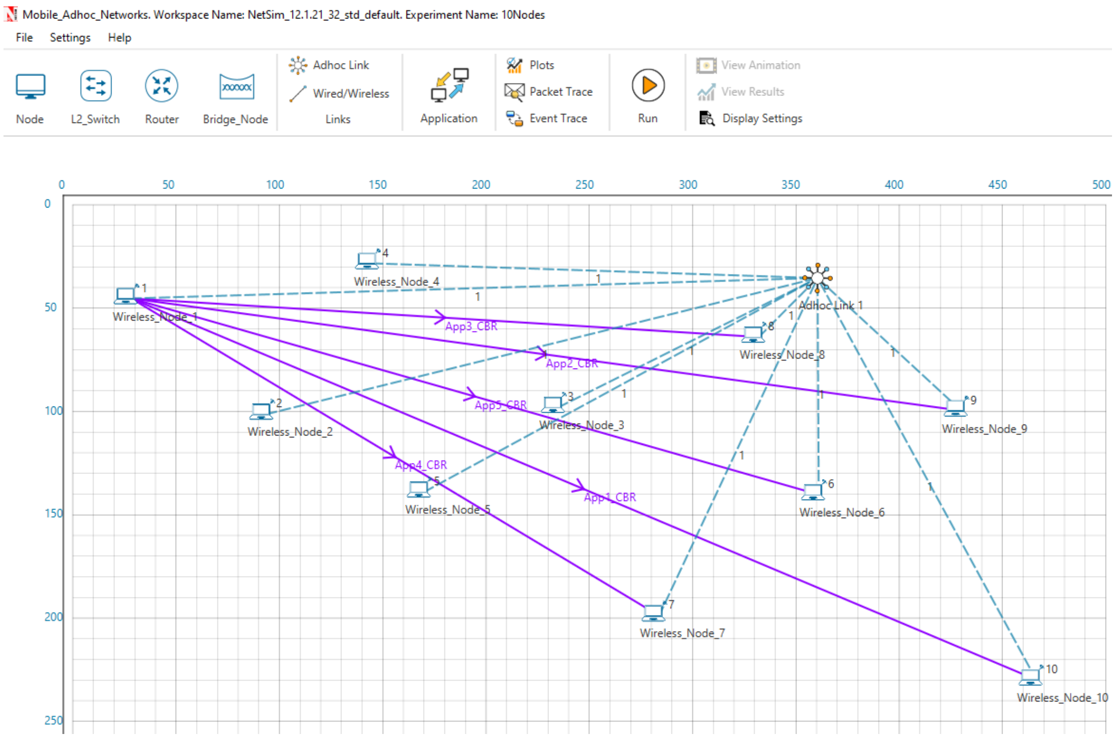
Support
Answers/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiments Manual

Documentation
User Manual
Technology Libraries
Source Code Help

Contact us
Email - sales@tetcos.com
Phone - +91 767 605 4321

9. Now create a Network Scenario in NetSim Mobile Adhoc Network as per the screenshot below.


Mobile_Adhoc_Networks. Workspace Name: NetSim_12.1.21_32_std_default. Experiment Name: 10Nodes

A sample Configuration.netsim file is also provided in the Config_File folder along with this project which can be opened in NetSim directly.

10. Run the simulation for 30 sec

Simulations have been carried out using a different number of nodes in a network to symbolize different practical applications of wireless network. For example, 10 nodes symbolize a small network that can be used in an agricultural setup. 30 nodes symbolize a medium size network that can be used in an industrial setup and a large 50 nodes network that can be used in an army base.

Result:

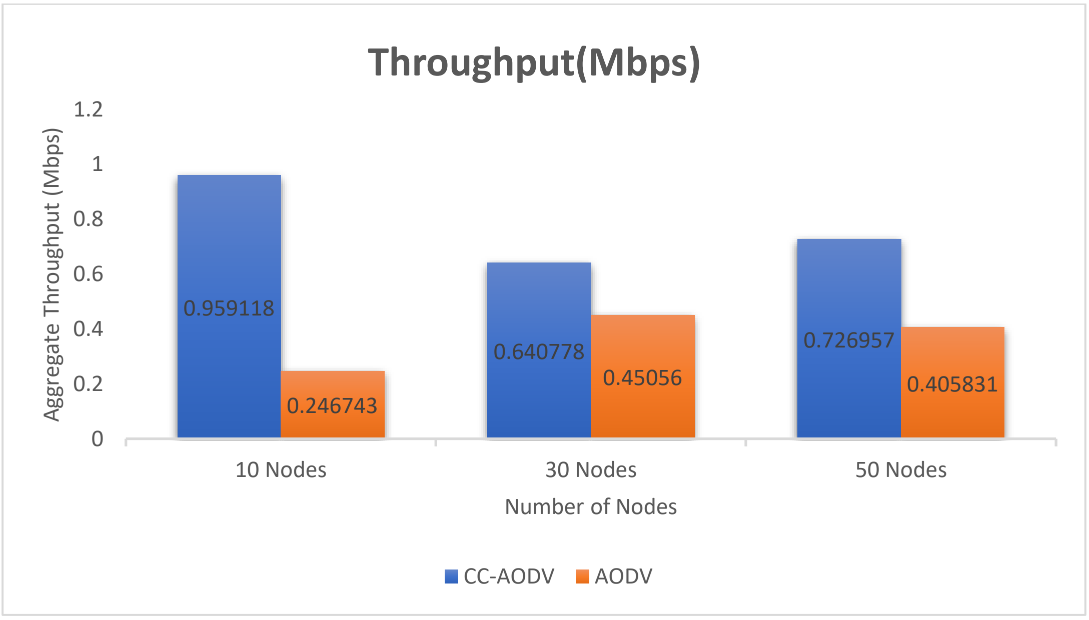
Performance of CC-AODV has been compared with other reactive protocol AODV based on different performance metrics such as Throughput, End to End delay etc.

Number of Nodes	AODV Aggregate Throughput (Mbps)	CC_AODV Aggregate Throughput (Mbps)
10Nodes	0.246743	0.959118
30Nodes	0.45056	0.640788
50Nodes	0.405831	0.726957

Table 1 : Aggregate Throughput comparison between AODV and CC_AODV

As per the Table 1 the proposed CC-AODV has higher throughput than the AODV. In CC-AODV, the internal nodes can be utilized much efficiently than AODV because the counter helps to reroute the path if the internal node is busy. This can increase the network channel utilization.

This can be further understood with the help of following graph:

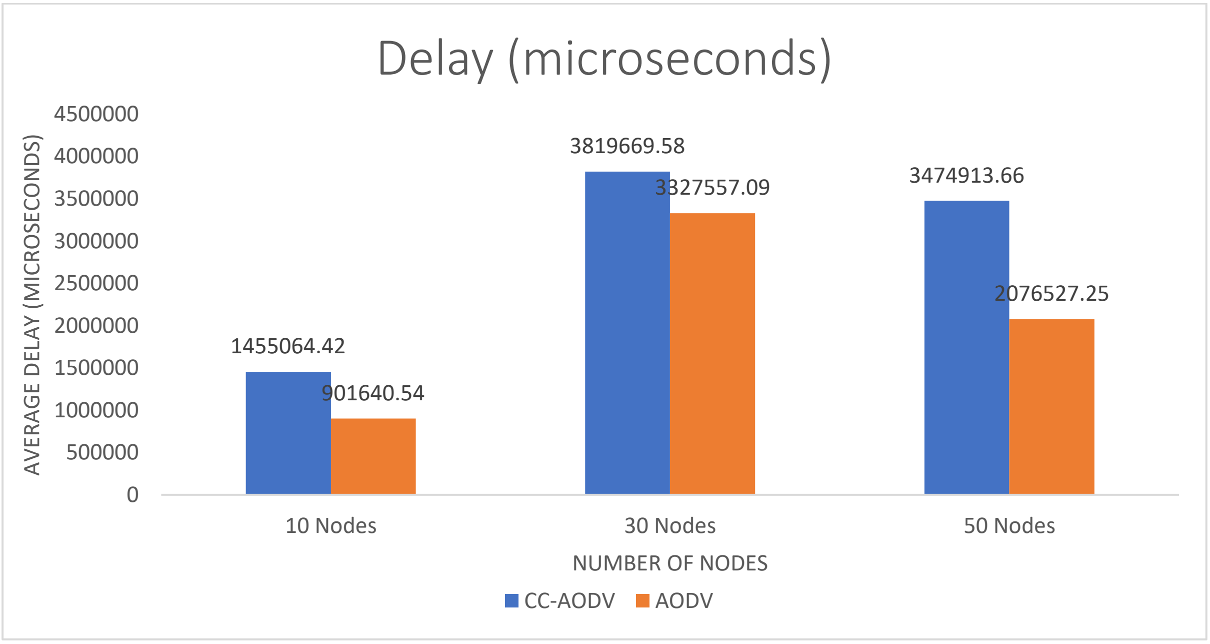


Number of Nodes	AODV Average Delay (microseconds)	CC_AODV Average Delay (microseconds)
10Nodes	901640.64	1455064.42
30Nodes	3327557.09	3819669.58
50Nodes	2076527.25	3474913.66

Table 2: End to End delay comparison between AODV and CC_AODV

Table 2 demonstrate that CC-AODV has slightly higher End-to-End performance than the AODV, the result is achieved by rerouting the path of the data if the router is on a busy state.

This can be further understood with the help of following graph:


Delay (microseconds)
AVERAGE DELAY (MICROSECONDS)
10 Nodes
30 Nodes
50 Nodes
NUMBER OF NODES
CC-AODV
AODV