# Primary User Emulation (PUE) Attack in Cognitive Radio Networks

**Software Recommended:** NetSim Standard v13.0 (32/64-bit), Visual Studio 2017/2019

**Project Download Link:**
https://github.com/NetSim-TETCOS/Probability-based-rebroadcast_v13.0/archive/refs/heads/main.zip

Cognitive Radio (CR) is a promising technology that can alleviate the spectrum shortage problem by enabling unlicensed users equipped with CRs to coexist with incumbent users in licensed spectrum bands while causing no interference to incumbent communications. Spectrum sensing is one of the essential mechanisms of CRs and its operational aspects are being investigated actively.

In a hostile environment, an attacker may modify the air interface of a CR to mimic a primary user signal's characteristic, thereby causing legitimate secondary users to erroneously identify the attacker as a primary user. We coin the term *primary user emulation (PUE) attack* to refer to this attack. There is a realistic possibility of PUE attacks since CRs are highly reconfigurable due to their software-based air interface.

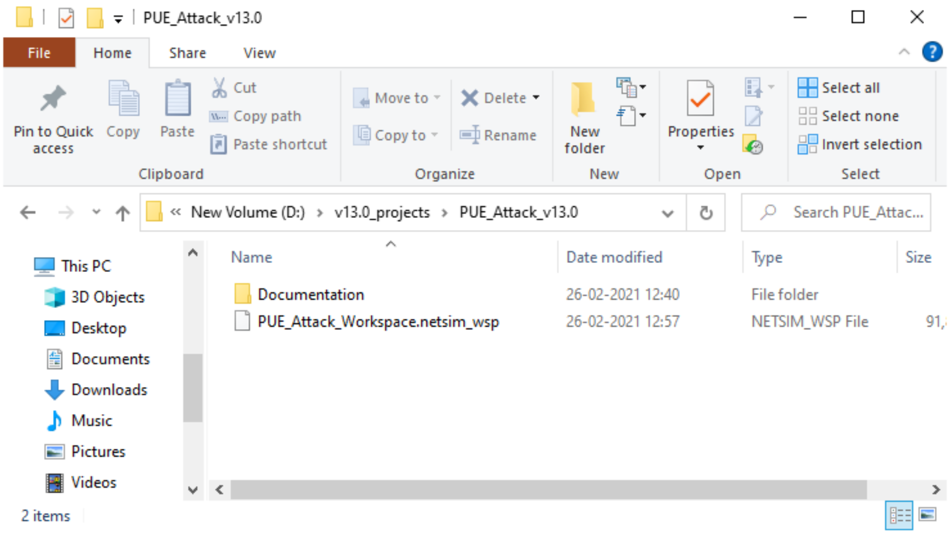
We create a PUE attack by adding two incumbents in the scenario in NetSim. One of the incumbents represents a "real" primary user while the second represents a "Malicious" primary user.

Our next goal is to detect the PUEA by the secondary users. For example purposes we have set the detection time as proportional to the distance of the secondary users from the malicious primary user.

The code given below is for an example implementation of PUE Attack.

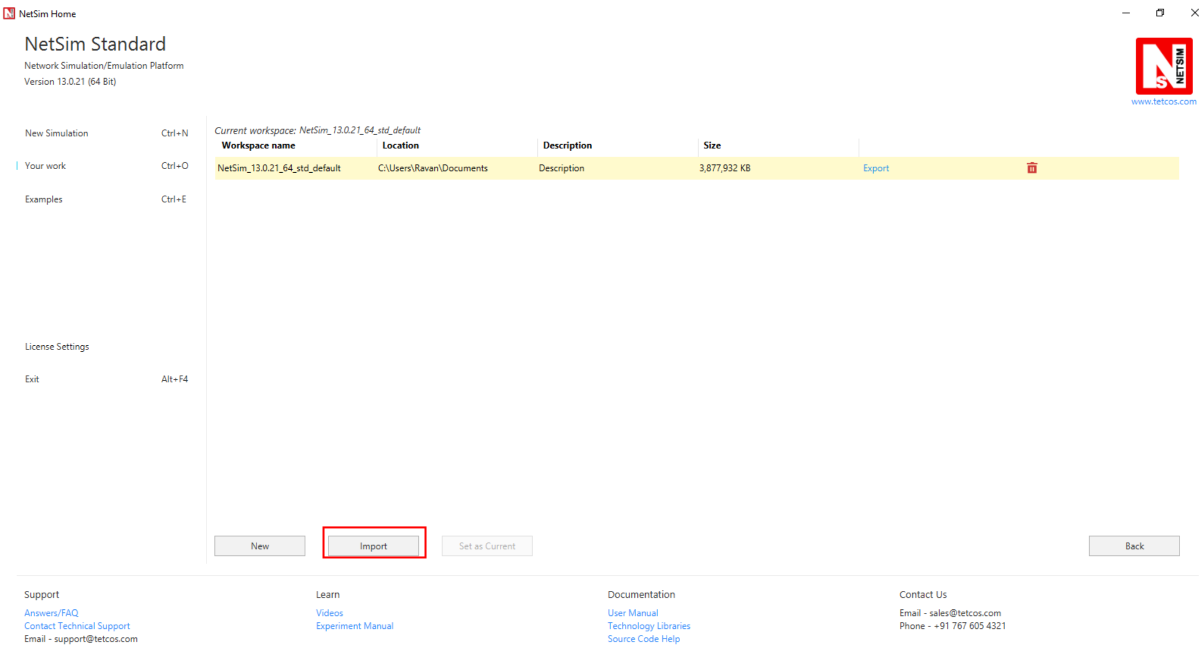
## Steps:

1. The downloaded project folder contains the folders Documentation and PUE_Attack_Workspace.netsim_wsp directory as shown below:



| Name | Date modified | Type | Size |
|---|---|---|---|
| Documentation | 26-02-2021 12:40 | File folder | |
| PUE_Attack_Workspace.netsim_wsp | 26-02-2021 12:57 | NETSIM_WSP File | 91, |

2. Import PUE_Attack_Workspace.netsim_wsp by going to Your work->Workspace Options->More Options in NetSim Home window. Then select Import as shown below:



NetSim Standard
Network Simulation/Emulation Platform
Version 13.0.21 (64 Bit)

*Current workspace: NetSim_13.0.21_64_std_default*

| Workspace name | Location | Description | Size | |
|---|---|---|---|---|
| NetSim_13.0.21_64_std_default | C:\Users\Ravan\Documents | Description | 3,877,932 KB | Export |

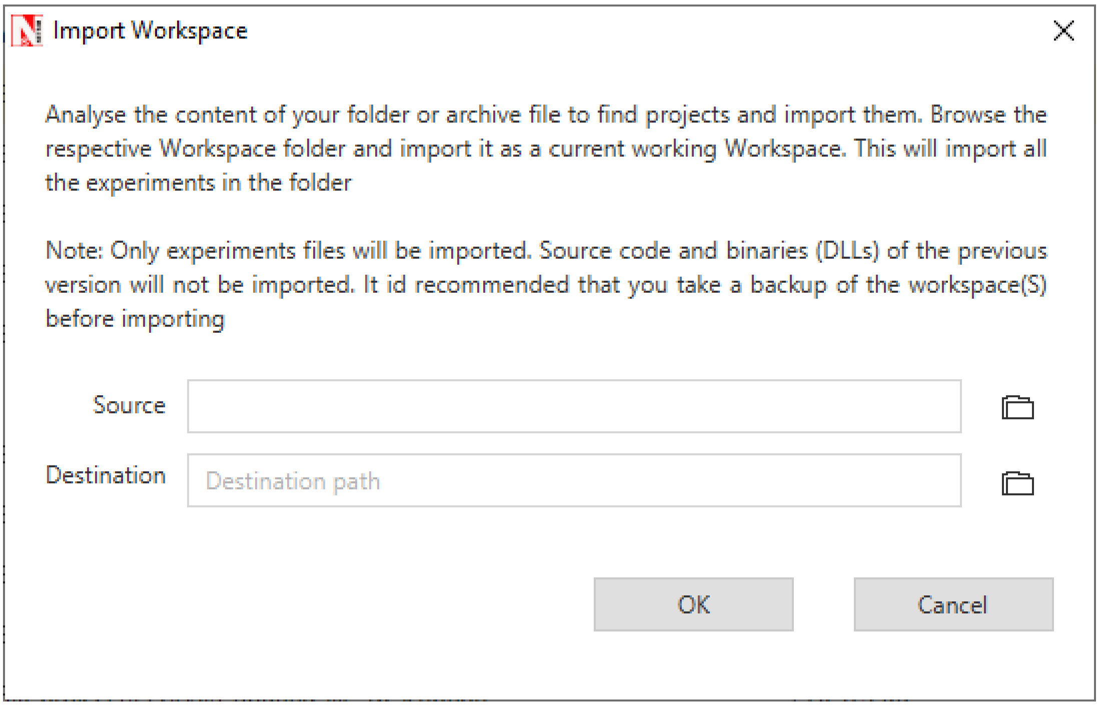
Support
Answers/FAQ
Contact Technical Support
Email - support@tetcos.com

Learn
Videos
Experiment Manual

Documentation
User Manual
Technology Libraries
Source Code Help

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

3. It displays a window where users need to give the path of the workspace folder and click on OK as shown below:



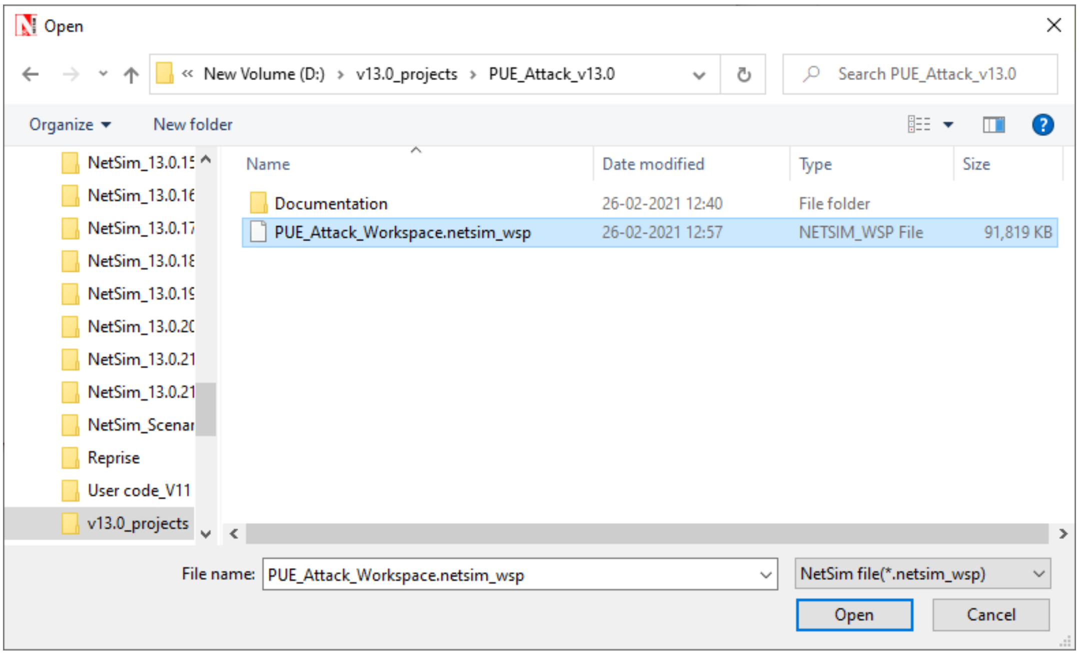
Analyse the content of your folder or archive file to find projects and import them. Browse the respective Workspace folder and import it as a current working Workspace. This will import all the experiments in the folder

Note: Only experiments files will be imported. Source code and binaries (DLLs) of the previous version will not be imported. It id recommended that you take a backup of the workspace(S) before importing

Source

Destination Destination path

4. Browse to the PUE_Attack_Workspace folder and click on select folder as shown below



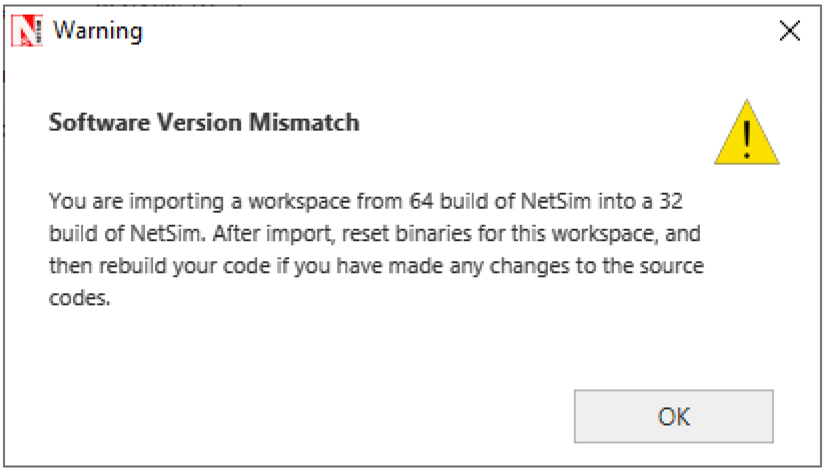
| Name | Date modified | Type | Size |
|---|---|---|---|
| Documentation | 26-02-2021 12:40 | File folder | |
| PUE_Attack_Workspace.netsim_wsp | 26-02-2021 12:57 | NETSIM_WSP File | 91,819 KB |

File name: PUE_Attack_Workspace.netsim_wsp

NetSim file(*.netsim_wsp)

**5.** After this click on OK button in the Import Workspace window.

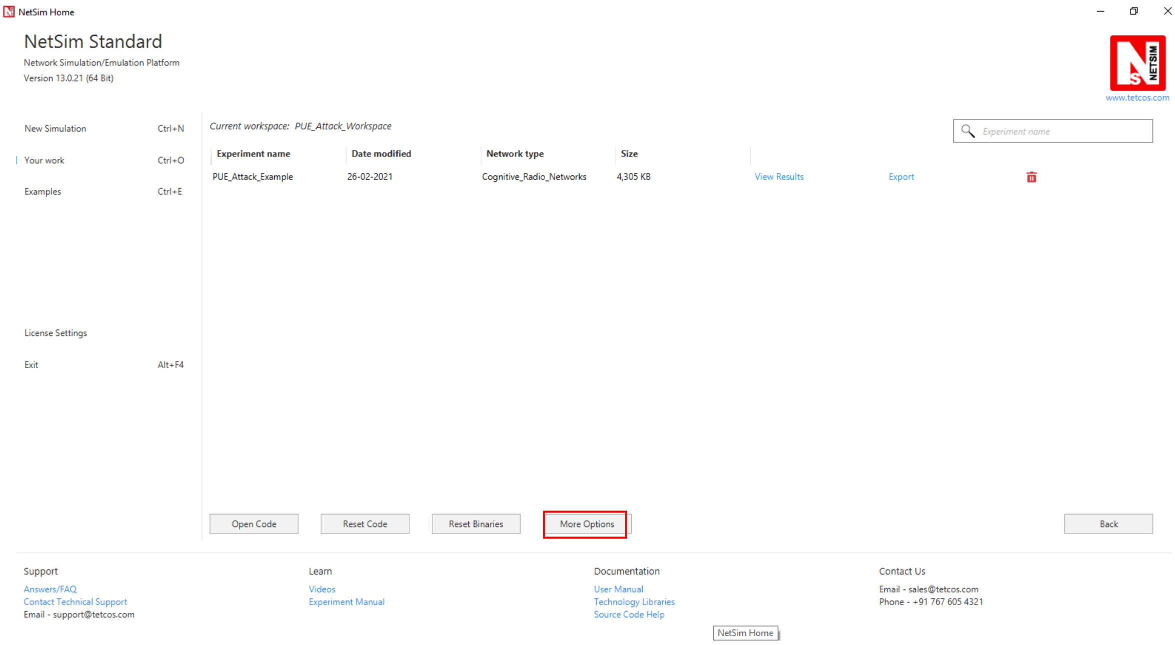
6. While importing the workspace, if the following warning message indicating Software Version Mismatch is displayed, you can ignore it and proceed.



**Software Version Mismatch**

You are importing a workspace from 64 build of NetSim into a 32 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.

7. The Imported workspace will be set as the current workspace automatically. To see the imported workspace, click on Your work->Workspace Options->More Options as shown below:



NetSim Standard
Network Simulation/Emulation Platform
Version 13.0.21 (64 Bit)

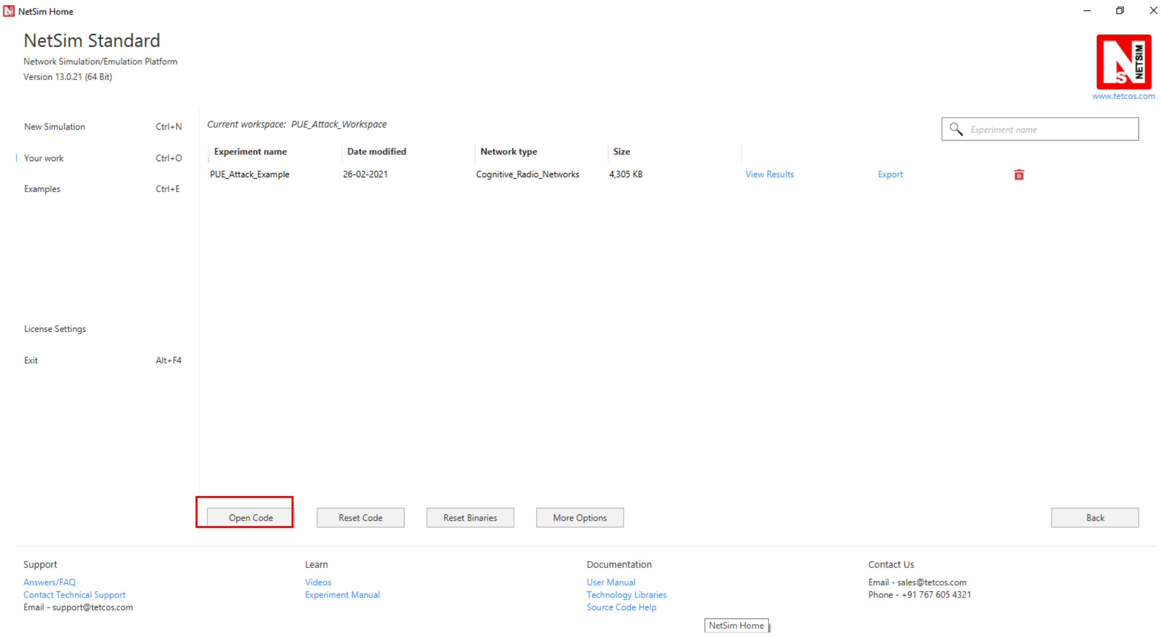
*Current workspace: PUE_Attack_Workspace*

| Experiment name | Date modified | Network type | Size | | | |
|---|---|---|---|---|---|---|
| PUE_Attack_Example | 26-02-2021 | Cognitive_Radio_Networks | 4,305 KB | View Results | Export | 🗑 |

Email - support@tetcos.com

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

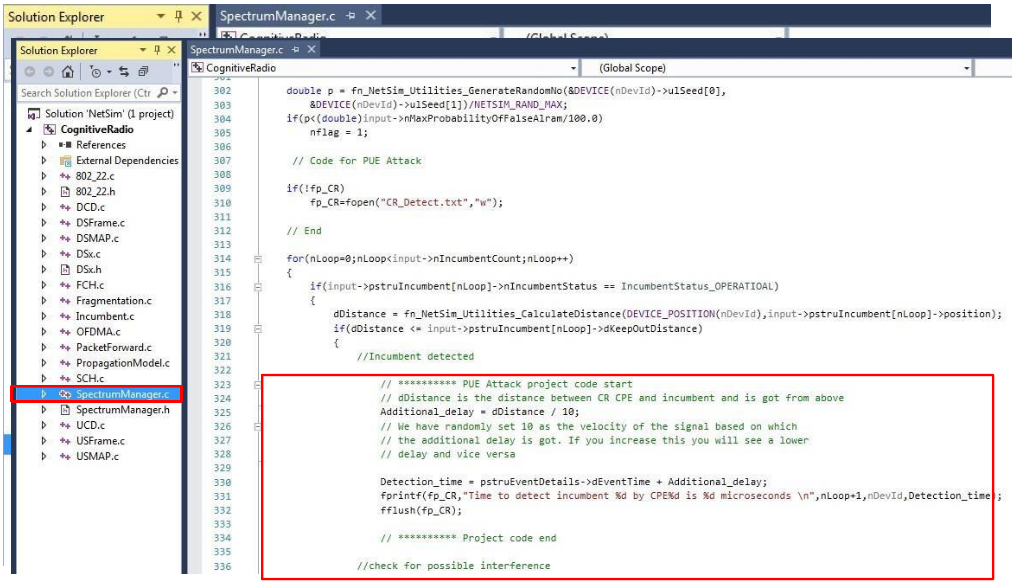
8. Open the Source codes in Visual Studio by going to Your work-> Workspace Options and Clicking on Open code button as shown below:



*Current workspace: PUE_Attack_Workspace*

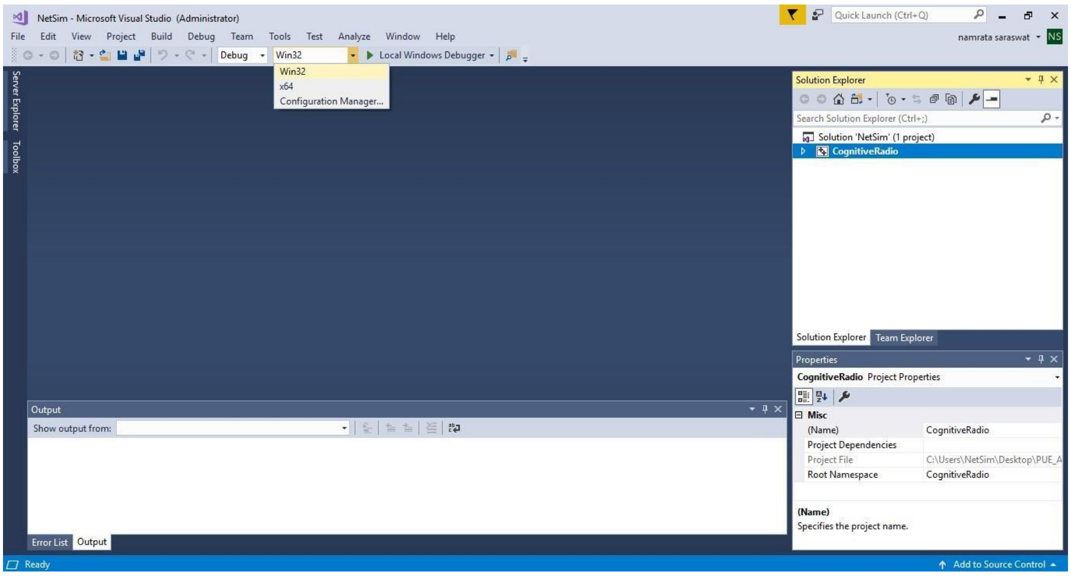
| Experiment name | Date modified | Network type | Size | | |
|---|---|---|---|---|---|
| PUE_Attack_Example | 26-02-2021 | Cognitive_Radio_Networks | 4,305 KB | View Results | Export |

**9.** Go to CognitiveRadio project->Open SpectrumManager.c. Inside the **SpectrumManager.c** file, the code to be modified is commented as **PUE Attack code.** Do the required modifications.



```
double p = fn_NetSim_Utilities_GenerateRandomNo(&DEVICE(nDevId)->ulSeed[0],
    &DEVICE(nDevId)->ulSeed[1])/NETSIM_RAND_MAX;
if(p<(double)input->nMaxProbabilityOfFalseAlram/100.0)
    nflag = 1;

 // Code for PUE Attack

if(!fp_CR)
    fp_CR=fopen("CR_Detect.txt","w");

// End

for(nLoop=0;nLoop<input->nIncumbentCount;nLoop++)
{
    if(input->pstruIncumbent[nLoop]->nIncumbentStatus == IncumbentStatus_OPERATIOAL)
    {
        dDistance = fn_NetSim_Utilities_CalculateDistance(DEVICE_POSITION(nDevId),input->pstruIncumbent[nLoop]->position);
        if(dDistance <= input->pstruIncumbent[nLoop]->dKeepOutDistance)
        {
            //Incumbent detected

                // ********** PUE Attack project code start
                // dDistance is the distance between CR CPE and incumbent and is got from above
                Additional_delay = dDistance / 10;
                // We have randomly set 10 as the velocity of the signal based on which
                // the additional delay is got. If you increase this you will see a lower
                // delay and vice versa

                Detection_time = pstruEventDetails->dEventTime + Additional_delay;
                fprintf(fp_CR,"Time to detect incumbent %d by CPE%d is %d microseconds \n",nLoop+1,nDevId,Detection_time);
                fflush(fp_CR);

                // ********** Project code end

            //check for possible interference
```

**10.** 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:



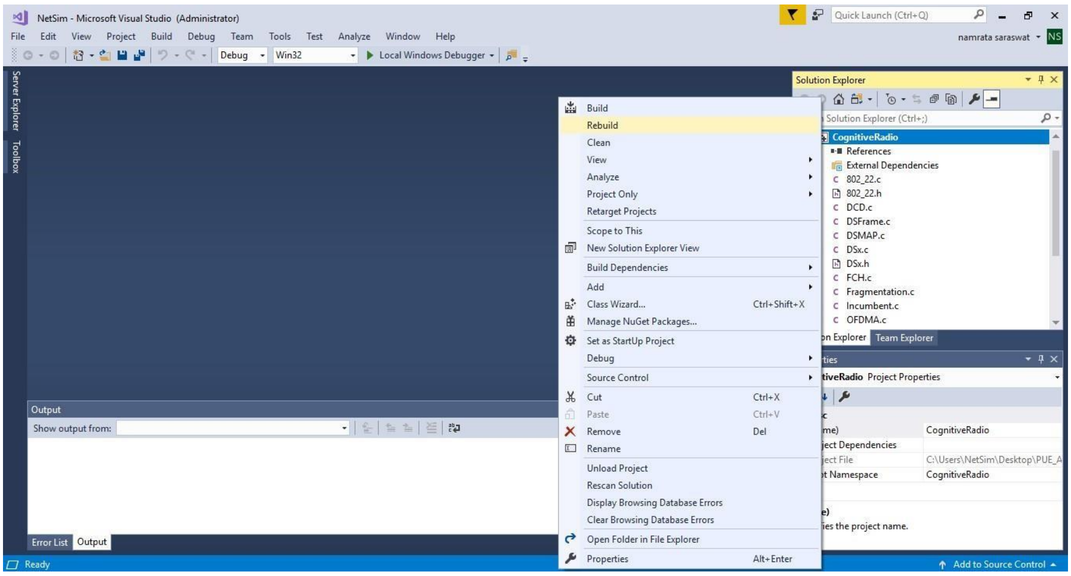
CognitiveRadio Project Properties

Misc

| | |
|---|---|
| (Name) | CognitiveRadio |
| Project Dependencies | |
| Project File | C:\Users\NetSim\Desktop\PUE_A |
| Root Namespace | CognitiveRadio |

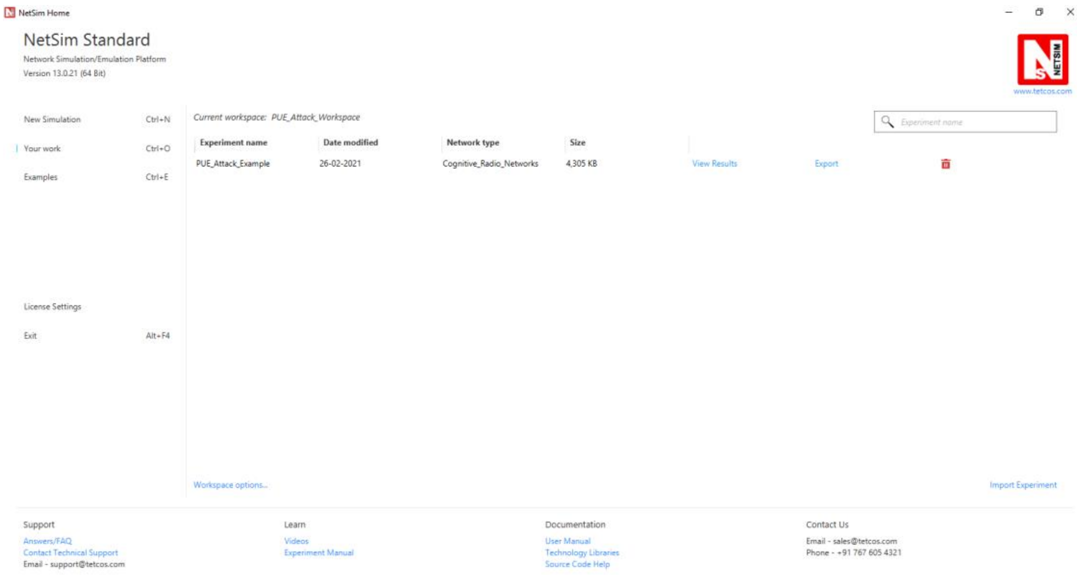
(Name)
Specifies the project name.

11. Right click on the Solution in the solution explorer and select Rebuild.



12. Upon successful build modified libCognitiveRadio.dll file gets automatically updated in the directory containing NetSim binaries.

13. Then PUE_Attack_Workspace comes with a sample configuration that is already saved. To open this example, go to Open Simulation and click on the PUE_Attack_Example that is present under the list of experiments as shown below:



NetSim Standard

Network Simulation/Emulation Platform

Version 13.0.21 (64 Bit)

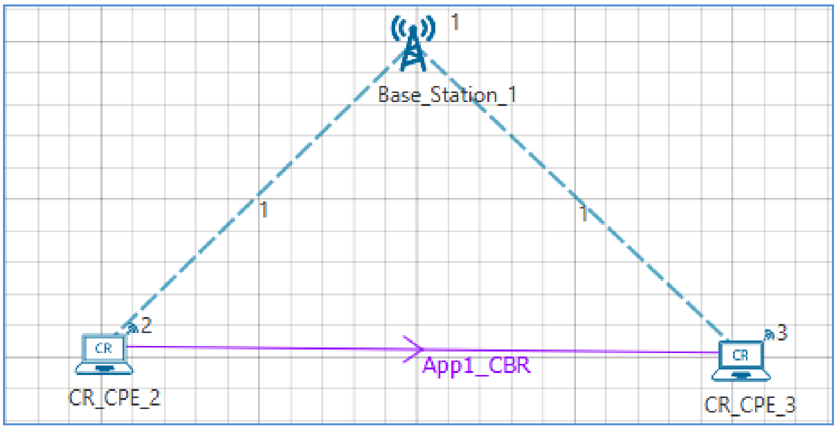
*Current workspace: PUE_Attack_Workspace*

| Experiment name | Date modified | Network type | Size |
|---|---|---|---|
| PUE_Attack_Example | 26-02-2021 | Cognitive_Radio_Networks | 4,305 KB |

Email - support@tetcos.com

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

14. The network scenario loads as shown below:

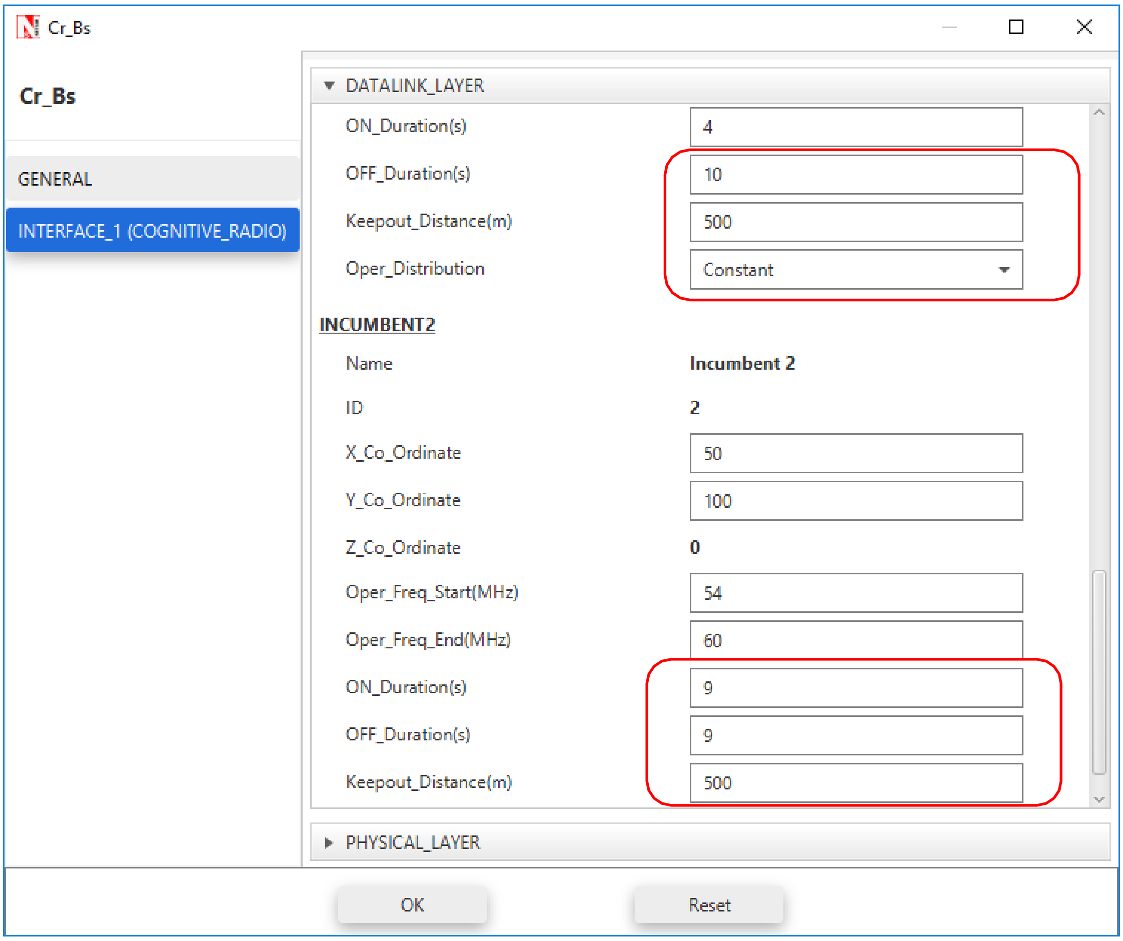


Following settings were done in the devices for this example.

15. In **CR-Base_Station_1/INTERFACE_1 (COGNITIVE_RADIO)->DATALINK_LAYER Incumbent** properties, the **Incumbent count** is set as **2**
16. In the Incumbent properties:
    In malicious (Incumbent_1)**, ON_Duration(s) – 4, OFF_Duration(s) –10**
    In Incumbent (Incumbent_2), **ON_Duration(s) – 9, OFF_Duration(s) –9**
    **Keep Distance = 500m in both incumbent** and the distance between the CPE and Incumbent is <500. This ensures that the incumbent is detected. If the incumbent is beyond the keep out distance then it is not detected.

The timing diagram is as follows:

Malicious --- 0s to 10s (OFF), 10s to 14s (ON), 14s to 24s (OFF), 24s to 28s (ON) ... and so on
Incumbent --- 0s to 9 s (OFF), 9s to 18s (ON), 18s to 27s (OFF), 27s to 36s (ON) ... and so on

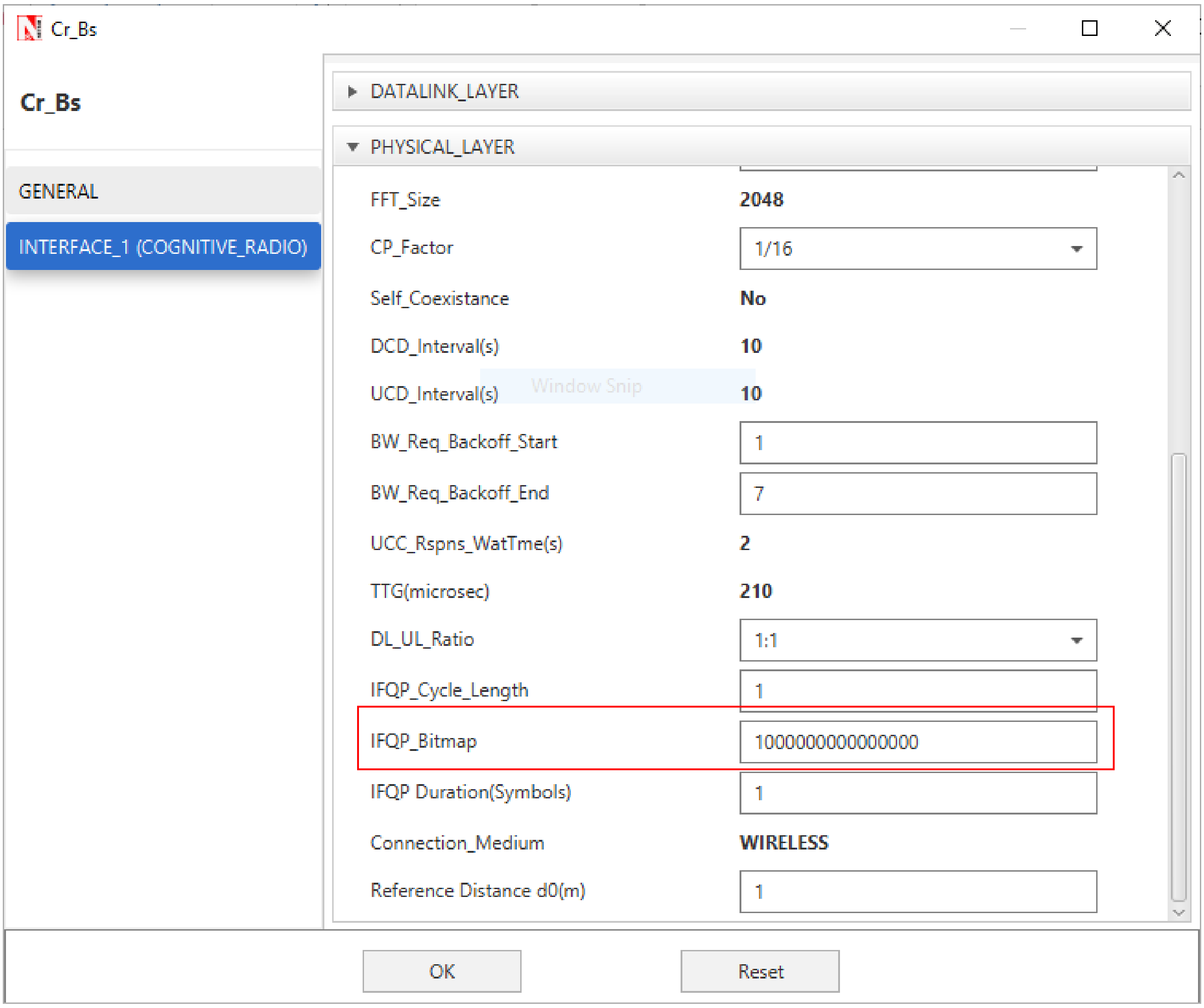


| | |
|---|---|
| ON_Duration(s) | 4 |
| OFF_Duration(s) | 10 |
| Keepout_Distance(m) | 500 |
| Oper_Distribution | Constant |

**INCUMBENT2**

| | |
|---|---|
| Name | **Incumbent 2** |
| ID | **2** |
| X_Co_Ordinate | 50 |
| Y_Co_Ordinate | 100 |
| Z_Co_Ordinate | **0** |
| Oper_Freq_Start(MHz) | 54 |
| Oper_Freq_End(MHz) | 60 |
| ON_Duration(s) | 9 |
| OFF_Duration(s) | 9 |
| Keepout_Distance(m) | 500 |

**17.** In physical layer, the **IFQP_Bitmap** is set to 1000000000000000



Cr_Bs

| | |
|---|---|
| FFT_Size | 2048 |
| CP_Factor | 1/16 |
| Self_Coexistance | No |
| DCD_Interval(s) | 10 |
| UCD_Interval(s) | 10 |
| BW_Req_Backoff_Start | 1 |
| BW_Req_Backoff_End | 7 |
| UCC_Rspns_WatTme(s) | 2 |
| TTG(microsec) | 210 |
| DL_UL_Ratio | 1:1 |
| IFQP_Cycle_Length | 1 |
| IFQP_Bitmap | 1000000000000000 |
| IFQP Duration(Symbols) | 1 |
| Connection_Medium | WIRELESS |
| Reference Distance d0(m) | 1 |

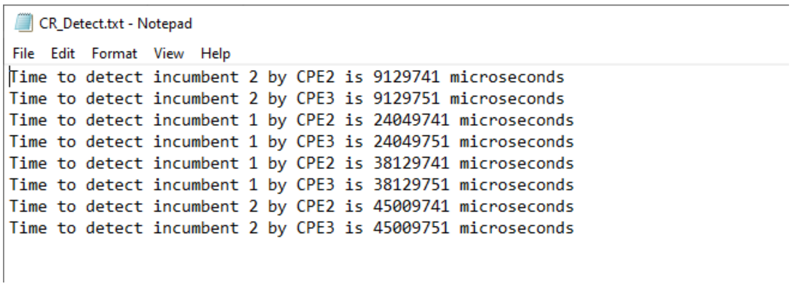
**18.** Now run the simulation 50 Sec.

**19.** You can see the delay in the **CR_Detect.txt** file inside bin folder. This additional delay has been set by the following code,

**Additional_delay = dDistance / 10;**

(You can also change the values as 10/100/1000 and analyse different variation in delay.)

A file “**CR_Detect.txt**” will be created in the bin folder (NetSim installation directory) with the following contents:



```
Time to detect incumbent 2 by CPE2 is 9129741 microseconds
Time to detect incumbent 2 by CPE3 is 9129751 microseconds
Time to detect incumbent 1 by CPE2 is 24049741 microseconds
Time to detect incumbent 1 by CPE3 is 24049751 microseconds
Time to detect incumbent 1 by CPE2 is 38129741 microseconds
Time to detect incumbent 1 by CPE3 is 38129751 microseconds
Time to detect incumbent 2 by CPE2 is 45009741 microseconds
Time to detect incumbent 2 by CPE3 is 45009751 microseconds
```

This is a simple implementation of creating and detecting a PUE Attack by making modifications to primary user detection in CR.