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/PUE_Attack_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

Introduction:

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.

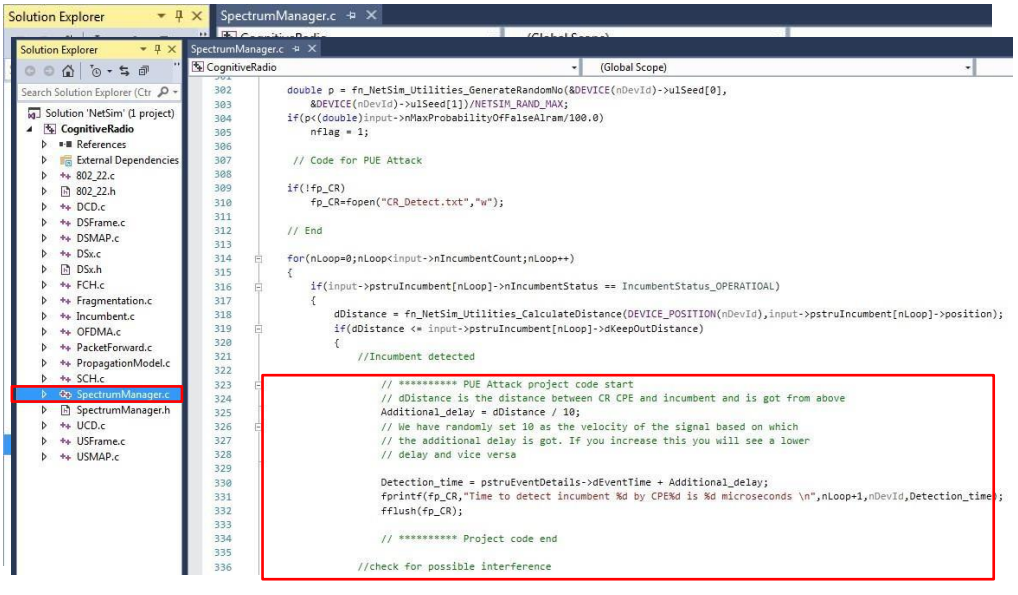
Our next goal is to detect the PUEA by the secondary users. For example purposes we have set the detection time as proportional to the distance of the secondary users from the malicious primary user.

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

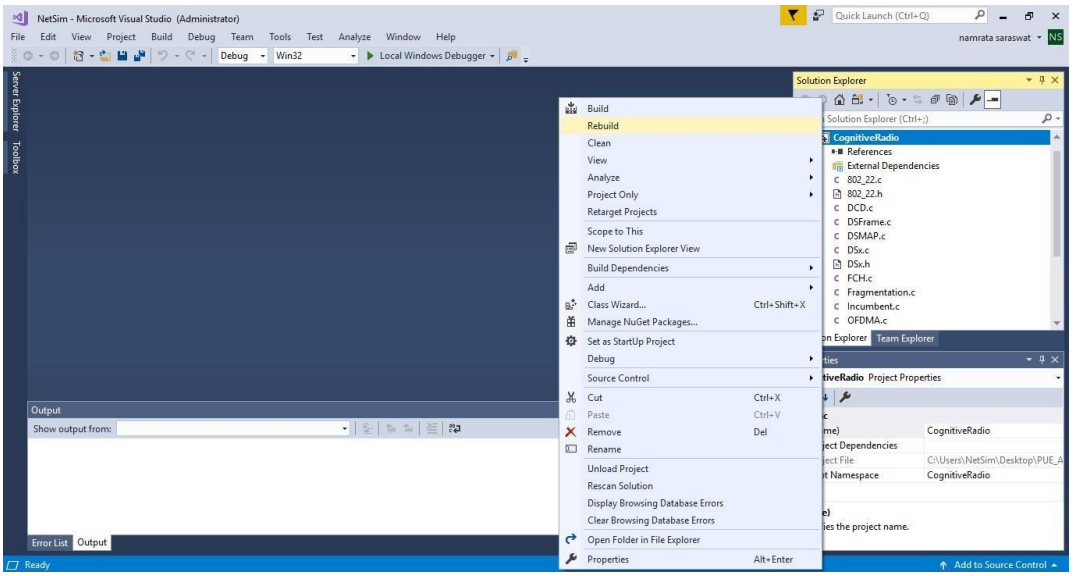
Steps:

1. Open the Source codes in Visual Studio by going to Your work-> Workspace Options and Clicking on Open code button.

2. 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.



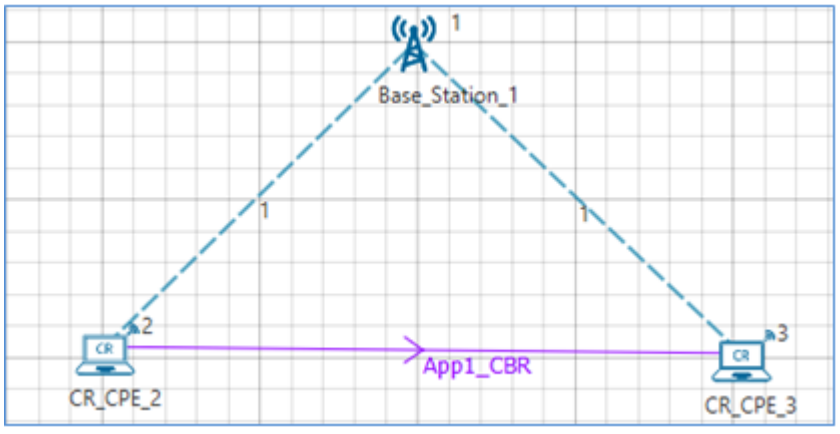
3. Right click on the Solution in the solution explorer and select Rebuild.



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

5. Then PUE_Attack_Workspace comes with a sample configuration that is already saved. To open this example, go to Your Work and click on the PUE_Attack_Example that is present under the list of experiments.

6. The network scenario loads as shown below:



Following settings were done in the devices for this example.

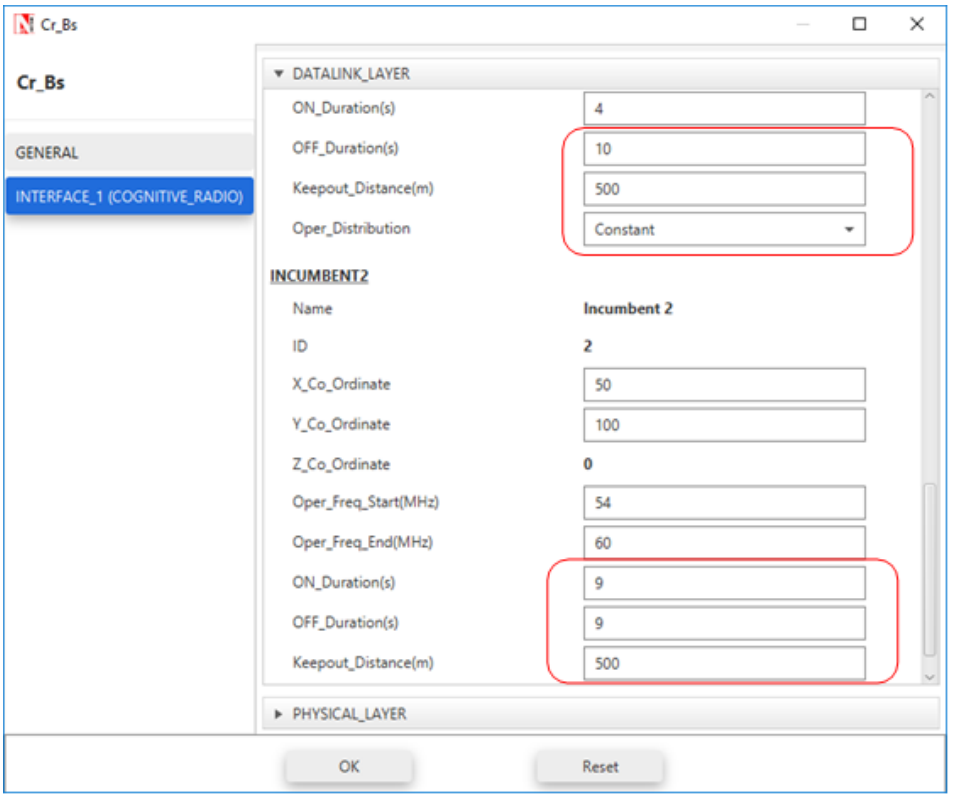
7. In **CR-Base_Station_1/INTERFACE_1 (COGNITIVE_RADIO)->DATALINK_LAYER Incumbent** properties, the **Incumbent count** is set as **2**
8. In the Incumbent properties:

In malicious (Incumbent_1)**, ON_Duration(s) – 4, OFF_Duration(s) –10**

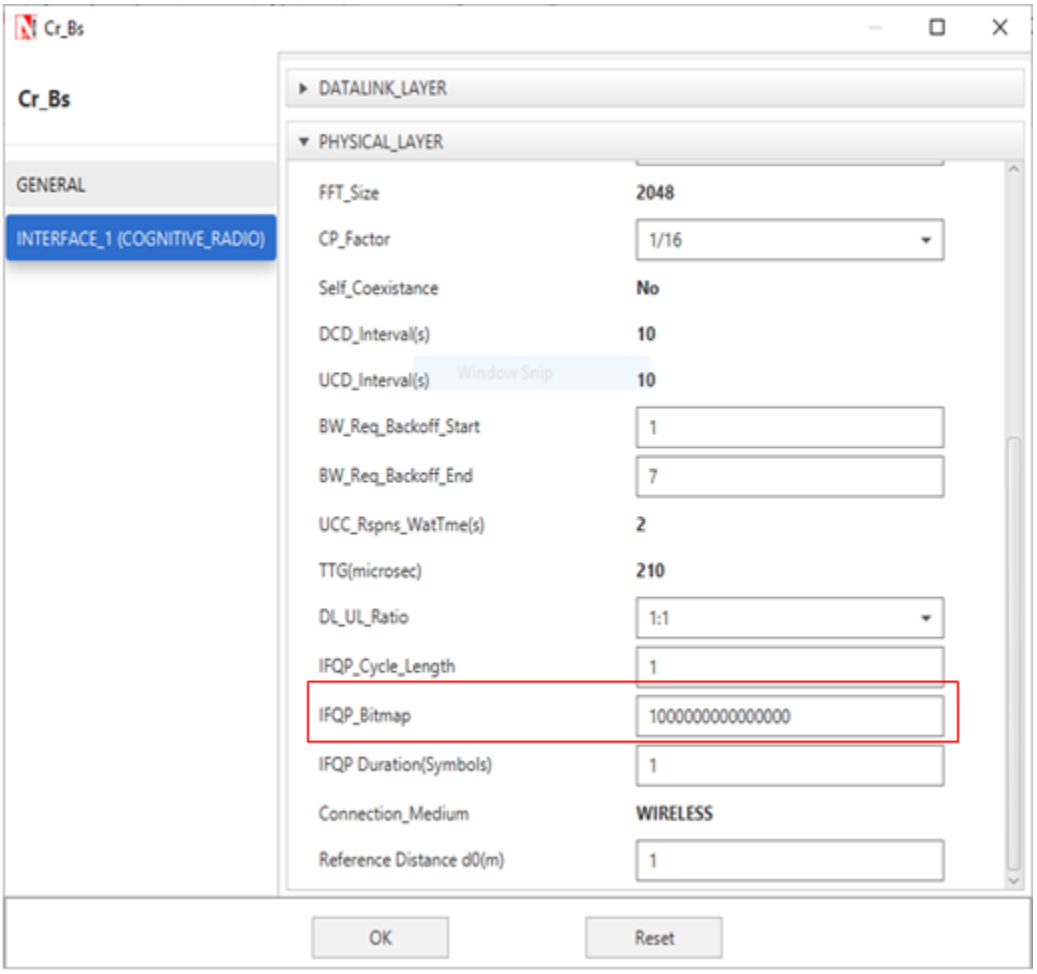
In Incumbent (Incumbent_2), **ON_Duration(s) – 9, OFF_Duration(s) – 9**
Keep Distance = 500m in both incumbent and the distance between the CPE and Incumbent is <500. This ensures that the incumbent is detected. If the incumbent is beyond the keep out distance then it is not detected.

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



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

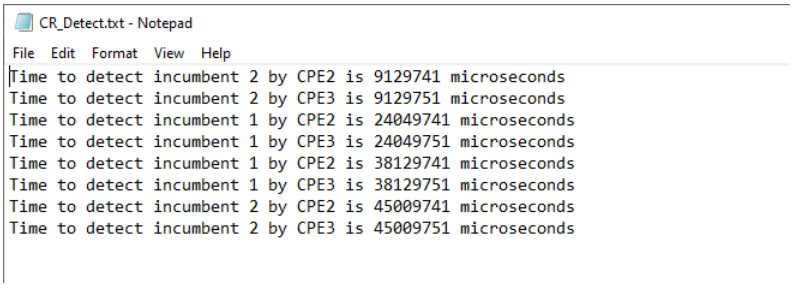


10. Now run the simulation 50 Sec.
11. You can see the delay in the **CR_Detect.txt** file inside bin folder. This additional delay has been set by the following code,

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:



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