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-upnetsim-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.
- 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.

	Will Hell Council	(Clabel Coste)
Solution Explorer 🔹 후 🗙	SpectrumManager.	c * X
○○☆ ७-5 ₫ "	CognitiveRadio	(Global Scope)
Search Solution Explorer (Ctr 👂 -	302	<pre>double p = fn_NetSim_Utilities_GenerateRandomNo(&DEVICE(nDevId)->ulSeed[0],</pre>
Solution 'NetSim' (I project)	303	&DEVICE(nDevid)->ulSeed[1])/NETSIM_RAND_MAX;
A to CognitiveRadio	304	if(p((double)input->nMaxProbabilityOfFalseAlram/100.0)
	305	ntlag = 1;
References	906	
P External Dependencies	307	// Code for PUE Attack
▷ ++ 802_22.c	308	
▶ 1 802_22.h	509	IT(ITP_CK)
++ DCD.c	310	Tp_ck=Topen(ck_Detect.txt , w);
++ DSFrame.c	311	// E
** DSMAP.c	212	// End
▷ ++ DSx.c	314	for(a) con-2 al conzinaut-information to con++)
Þ ⊡ DSx.h	315	/ ////////////////////////////////////
b the ECH c	216	if(input-)nstruTncumhent[nl.on]-)nTncumhentStatus TncumhentStatus (DEPATTOAL)
b the Fragmentation c	317	
h du Insumbent s	318	dDistance = fn NetSim Utilities (alculateDistance/DEVICE_POSITION(nDevId).input->nstruIncumbentInloon)->nositio
b the OFDMA	319 E	if(dDistance <= input->bstpuIncumbentinloon1->dKeenOutDistance)
P ++ OPDMA.c	320	
P ++ PacketForward.c	321	//Incumbent detected
P ++ PropagationModel.c	322	8.5.000 (Lany)************************************
++ SCH.c	323 -	// ********* PUE Attack project code start
A SpectrumManager.c	324	// dDistance is the distance between CR CPE and incumbent and is got from above
SpectrumManager.h	325	Additional_delay = dDistance / 10;
++ UCD.c	326 E	// We have randomly set 10 as the velocity of the signal based on which
** USFrame.c	327	// the additional delay is got. If you increase this you will see a lower
b ++ USMAP.c	328	// delay and vice versa
A	329	
	330	Detection_time = pstruEventDetails->dEventTime + Additional_delay;
	331	fprintf(fp_CR,"Time to detect incumbent %d by CPE%d is %d microseconds \n",nLoop+1,nDevId,Detection_tim
	332	fflush(fp_CR);
	333	
	334	// ********* Project code end
	335	
	336	//check for possible interference

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

MetSim - Microsoft Visual Studio (Administrator)			۲ :	Quick Launch (Ctri	1+Q) 🔑 🗕 🗗 🗙	
File Edit View Project Build Debug Team Tools Test Analyze Window Help					namrata saraswat 👻 NS	
💿 - 💿 🔯 - 🏩 🔐 🦻 - 🤆 - 🛛 Debug - 🛛 Win32 💿 - 🕨 Local Windows Debugger - 💋 -						
ξ.			Solu	tion Explorer	* ↓ ×	
e c		Decited		700.00	5 8 B / -	
piore	Rebuild			Solution Explorer (Ctrl+;)		
		Clean		CognitiveRadio	· · · · · · · · · · · · · · · · · · ·	
0 6		View		■ References	transfer in the second s	
×		Analyze		c 802 22.c	aencies	
		Project Only		▶ 📑 802_22.h		
		Retarget Projects		c DCD.c		
		Scope to This		C DSFrame.c		
	ē	New Solution Explorer View		c DSx.c		
		Build Dependencies	•	DSx.h		
		Add		C FCH.c		
	в	Class Wizard	Ctrl+Shift+X	C Incumbent.c	c .	
		Manage NuGet Packages		C OFDMA.c	*	
	Ф	Set as StartUp Project		p <mark>n Explorer</mark> Team Explorer		
		Debug		• ties • + ×		
		Source Control		tiveRadio Project Properties		
	ж	Cut	Ctrl+X	4 1		
Output	£	Paste	Ctrl+V	c		
Show output from:	×	Remove	Del	me)	CognitiveRadio	
		Rename		ject Dependencies	Collinear MatSine) Declares DUE A	
		Unload Project		at Namespace	CognitiveRadio	
		Rescan Solution		,		
		Display Browsing Database Errors		P)		
		Clear Browsing Database Errors	ies the project name.		2	
Error List Output	ç	Open Folder in File Explorer				
CT Ready	بر	Properties	Alt+Enter		↑ Add to Source Control ▲	

- 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

Cr_Bs							
Cr Bs	▼ DATALINK_LAYER						
	ON_Duration(s)	4	4				
GENERAL	OFF_Duration(s)	10	10 500				
INTERFACE_1 (COGNITIVE_RADIO)	Keepout_Distance(m)	500					
	Oper_Distribution	Constant	-				
	INCUMBENT2						
	Name	Incumbent 2	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					
	PHYSICAL_LAYER						

Cr_Bs		– 🗆 X				
Cr_Bs	► DATALINK_LAYER					
	PHYSICAL_LAYER					
GENERAL	FFT_Size	2048				
INTERFACE_1 (COGNITIVE_RADIO)	CP_Factor	1/16 👻				
	Self_Coexistance	No				
	DCD_Interval(s)	10				
	UCD_Interval(s) Window Snip	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	10000000000000				
	IFQP Duration(Symbols)	1				
	Connection_Medium	WIRELESS				
	Reference Distance d0(m)	1				
	ОК	Reset				

- **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:

CR_Detect.txt - Notepad								
File	Edit	Format	View Help					
Time	e to	detect	incumbent	2	by	CPE2	is	9129741 microseconds
Time	e to	detect	incumbent	2	by	CPE3	is	9129751 microseconds
Time	e to	detect	incumbent	1	by	CPE2	is	24049741 microseconds
Time	e to	detect	incumbent	1	by	CPE3	is	24049751 microseconds
Time	e to	detect	incumbent	1	by	CPE2	is	38129741 microseconds
Time	e to	detect	incumbent	1	by	CPE3	is	38129751 microseconds
Time	e to	detect	incumbent	2	by	CPE2	is	45009741 microseconds
Time	e 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.