Sink Hole Attack using RPL in IOT

Software Recommended: NetSim Standard v11.0, Visual Studio 2015/2017 Project Download Link: https://github.com/NetSim-TETCOS/SINK_HOLE_RPL_v11.0/archive/master.zip

In sinkhole Attack, a compromised node or malicious node advertises fake rank information to form the fake routes. After receiving the message packet it drop the packet information. Sinkhole attacks affect the performance of IoT networks protocols such as RPL protocol.

Implementation in RPL (for 1 sink)

- In RPL the transmitter broadcasts the DIO during DODAG formation.
- The receiver on receiving the DIO from the transmitter updates its parent list, sibling list, rank and sends a DAO message with route information.
- Malicious node upon receiving the DIO message it does not update the rank instead it always advertises a fake rank.
- The other node on listening to the malicious node DIO message the update their rank according to the fake rank.
- After the formation of DODAG, if the node that is transmitting the packet has malicious node as the preferred parent, transmits the packet to it but the malicious node instead of transmitting the packet to its parent, it simply drops the packet resulting in zero throughput.

A file Malicious.c is added to the RPL project.

The file contains the following functions

- fn_NetSim_RPL_MaliciousNode()
 This function is used to identify whether a current device is malicious or not in-order to establish malicious behaviour.
- fn_NetSim_RPL_MaliciousRank()
 This function is used to give a fake rank to the malicious node.
- rpl_drop_msg()
 This function is used to drop the packet by the malicious node if it enters into its network layer.

Sink Hole attack – The malicious node advertises the fake rank.

fn_NetSim_RPL_MaliciousRank() is the sink hole attack function.

Black Hole attack – The malicious node drops the packet. rpl drp msg() is the black hole attack function

You can set any device as malicious and you can have more than one malicious node in a scenario. Device id's of malicious nodes can be set inside the fn_NetSim_RPL_MaliciousNode() function.

Steps:

1. Open the Code folder and double click on the NetSim.sln fie to open the project in Visual Studio 2015.

nclude	File folder	
IP	File folder	
📙 lib	File folder	
RPL	File folder	
NetSim	Microsoft Visual S	2 KB
NetSim.VC	Data Base File	220 KB

2. Set malicious node id and the fake rank.

Solution Explorer 🔹 म 🗙	Malicious.c 👳 🗙	
ⓒ○☆ ७-५률@ "	💁 RPL	- (Global Scope) -
Search Solution Explorer (Ctrl- 🔎 -	1	⊟#include "main.h "
Solution 'NetSim' (1 project)	2	#include "RPL.h"
▲ SPL ▶ ■ References	3	<pre>#include "RPL enum.h"</pre>
External Dependencies	4	#define MALICIOUS_NODE1 7
++ DAO.c ++ DIO.c	5	#define MALICIOUS_RANK1 3
> ++ DIS.c	6	
 A B Malicious.c 	7	#define MALICIOUS_NODE2 4
** Neighbor.c ** PDL c	8	#define MALICIOUS_RANK2 4
P IN RPL.b	9	
*** RPL_enum.c B RPL enum.h	10	=/**
++ RPL_Message.c	11	Function prototypes
RPL_Message.h ++ SequenceNumber.c	12	*/
↓ ++ Trickle.c	13	<pre>int fn_NetSim_RPL_MaliciousNode(NetSim_EVENTDETAILS*);</pre>

3. Add the code that is highlighted in RPL.c file



4. Now right click on RPL project in the solution explorer and select Rebuild.

Malicious.c	+ X	_				
💁 RPL		Soluti	on E	xplorer		* 🗆 X
1	⊟#include "main.h"	G	ខែផ	∐ ™ ⊡ •	÷.	a 🕼 🌶 🗕 🎝
2	#include "RPL.h"	Searc	h Sol	ution Exp	lorer ((Ctrl+;)
3	#include "RPL_enum.h"	1 1 1 1	Solut	ion 'NetS	im' (1	project)
4	#define MALICIOUS_NODE1 7		E R	PI		projecty
5	#define MALICIOUS_RANK1 3			Referen	4	Build
6				External		Rebuild
7	#define MALICIOUS_NODE2 4		- +	DAO.c		Clean
8	#define MALICIOUS_RANK2 4	Þ	- 44	DIO.c		View
9		Þ	- 44	DIS.c		view
10		Þ	- 44	DODAG		Analyze
11	Function prototypes	Þ	- 44	 Malicio 		Project Only
12	*/	Þ	- •	Neighb		Retarget SDK Version
13	<pre>int fn_NetSim_RPL_MaliciousNode(NetSim_EVENTDETAILS*);</pre>	Þ	- 44	F RPL.c		Scone to This
14	<pre>void fn_NetSim_RPL_MaliciousRank(NetSim_EVENTDETAILS*);</pre>	Þ	h	RPL.h	-	New Celutine Fueleses View
15	<pre>void rpl_drop_msg();</pre>	Þ	- *	RPL_en		ivew solution Explorer view
16	<pre>int fn_NetSim_RPL_FreePacket(NetSim_PACKET*);</pre>	⊳	6) RPL_en	32	Show on Code Map
17		Þ	- *	RPL_Me		Profile Guided Optimizatio
18	<pre>int fn_NetSim_RPL_MaliciousNode(NetSim_EVENTDETAILS* pstruEventDetails)</pre>	Þ	Б) RPL_Me		Build Dependencies
19	{	Þ	- *	Sequen		build Dependencies
20	<pre>if(pstruEventDetails->nDeviceId == MALICIOUS_NODE1)</pre>	Þ	- *	 Trickle. 		Add
21	<pre>{ /*For multiple malicious nodes use if(pstruEventDetails->nDevi</pre>	C			₽.	Class Wizard
22	return 1;				苗	Manage NuGet Packages
23	}				-	Set as Startlin Droject
24	return 0;				~*	D I
25	}					Debug
103.20					v	A

- 5. Now Copy the newly built libRPL.dll from the DLL folder inside the Simulation Sinkhole Attack directory.
- 6. Replace the DLL in NetSim bin folder in the NetSim installation directory, after renaming the original libRPL.dll file.

Local D	Disk (C:) 🔸 Program Files 🕨 NetSim Sta	ndard ⊁ bin ⊁ 🗸 🚽	Search bin
🔲 Ope	en with New folder		
	Name	Туре	Size
	🚳 libRouting.dll	Application extens	. 64 KB
	🚳 libRPL.dll	Application extens	. 75 KB
	libRPL_orig.dll	Application extens	. 41 KB
	🚳 libTCP.dll	Application extens	. 59 KB
	🚳 libTDMA.dll	Application extens	. 23 KB
	libToken_BR.dll	Application extens	. 16 KB
	libUDP.dll	Application extens.	. 17 KB
	libWiMax.dll	Application extens	. 37 KB

- 7. Create a network scenario in IoT (Internet of Things) with UDP running in the Transport Layer and RPL in Network Layer.
- 8. For example, you can create a scenario as shown in the following screenshot:



Environment Properties:

- Right click anywhere on the Environment Grid and select Properties.
- Select the Channel Characteristics and set the parameters accordingly.

N Link Properties Wir	dow		—		×					
Link_Type		MULTIPOINT_TO_MULTIPOINT								
Link_Medium		WIRELESS								
Link_Mode		HALF_DUPLEX	HALF_DUPLEX							
MEDIUM_PROPERT	Y									
Channel_Character	istics	PATHLOSS_ONLY	PATHLOSS_ONLY -							
Path Loss Model		LOG_DISTANCE			-					
PathLoss_Exponent	(n)	2								

Output

- Press 💽 + R and type %temp%, Temp folder will be opened.
- In Temp folder you will find a folder named NetSim.
- In NetSim, you will find a txt file named rpllog.txt

Open **rpllog.txt** then you will find the information about DODAG formation. For every DODAG, 6LoWPAN Gateway is the root of the DODAG

- Root is 1 with rank = 1 (Since the Node Id_1 is 6LoWPAN Gateway)
- Wireless_Sensor_Node_7(Malicious Node)

Packet is transmitted by node 8(Sensor_8) is received by node 7(Sensor_7) since the node 7 is malicious node it drops the packet. So the Throughput in this scenario is 0.

Open **Packet trace** file from simulation results window and filter only the data packets now check the **Transmitter_Id and receiver_Id** column. Since the node 7 is malicious node it drops the packet without forwarding it further.

X≣	5	- 0		Pa	cket Trace -	Excel				TABLE TOC	LS		? 🗈	- 6
FI	LE H	OME	INSERT PAGE LAY	OUT FORMULAS	DATA	REVIEW	VIEW	ADD-IN	IS TEAM	DESIGN				Sign
Pas	■	Calibri B I	- 11 - A			General \$ - %	€.0 .00 .	.00 .00	Conditional For Format as Table Cell Styles *	matting • •	E Insert ▼ E Delete ▼ Format ▼	∑ . ↓ . 	Sort &	Find & Select •
Clip	board 🗔		Font	Alignment الآ	- Fa	Nui	nber	Fail I	Styles		Cells		Editin	g
G2		-	$\times \sqrt{f_x}$	SINKNODE-1										
L														
	PACKET_I	C 🖵 SE	GMEL PACKET_TYPE	CONTROL_PACKE	T_TYPE/APF	_NAME	▼ SOURC	CE_ID 💌	DESTINATION	I_ID 🖵 TI	RANSMITTER	_ID →	RECEIV	ER_ID 👻
297		5	0 Sensing	App1_SENSOR_A	PP		SENSO	R-8	NODE-3	SE	NSOR-8		SENSOF	-7
308		6	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8		SENSOF	- 7
320		7	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8		SENSOF	ł-7
341		8	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8		SENSOF	}-7
361		9	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8		SENSOF	-7
382		10	0 Sensing	App1_SENSOR_A	PP		SENSO	R-8	NODE-3	SE	NSOR-8		SENSOF	-7
396		11	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8		SENSOF	-7
407		12	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8		SENSOF	-7
419		13	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8	1	SENSOF	-7
439		14	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8		SENSOF	-7
455		15	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8	1	SENSOF	-7
470		16	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8		SENSOF	-7
492		17	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8	1	SENSOF	-7
504		18	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8		SENSOF	-7
517		19	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8		SENSOF	-7
529		20	0 Sensing	App1_SENSOR_A	РР		SENSO	R-8	NODE-3	SE	NSOR-8		SENSOF	}-7