CSE 589 Project 3

Evaluate MAC random transmission protocol using NS-2

Due Date: @23:59:59, Dec 18th, 2016.

Introduction

In this project, we will use network simulator (NS-2) to implement and evaluate a wireless MAC (Medium Access Control) protocol for sensor networks. From this project, we will learn the basic operations of NS-2. We will also learn how to collect and analyze simulation statistics. We will provide a Virtual Box file with NS-2 installed. Refer to Appendix I for how to use Virtual Box file and a brief process of your experiment. Before starting the project, you are advised to study an NS-2 tutorial (e.g. Marc Greis's tutorial). Refer to Appendix II for the URL of the tutorial and other useful sites including the NS-2 manual.

Project Description

This project asks you to implement a simple MAC protocol, where source nodes only use RF (Radio Frequency) transmitters to transmit (data generated by sensors) packets. The network architecture consists of one sink node which receives the packets transmitted by source nodes. The source nodes are all within one-hop communication range of the sink node. The only reason for unsuccessful transmission is due to collision among multiple packets sent by multiple source nodes. The source nodes generate and transmit data as follows:

1) A data packet is generated every T seconds and has to be delivered to the sink before the next packet is generated.

2) The source nodes are equipped with only an RF transmitter, it is impossible for them to sense the channel or receive any acknowledgements (or negative acknowledgement) from the sink node.

3) In order to increase the chance of successful transmission, each node transmits X copies of each packet at random instants before the next packet is generated.

4) The source node picks X random instants of time within the interval [0, T] (i.e. the interval between the time the current packet is generated to the time the next packet will be generated),

and transmits the data packet at each of the X instants of time.

Simulation

The desired network configuration is as follows:

- 100 source nodes, one sink node
- Terrain Dimension: 50m x 50m
- Simulation Time: 100 s
- Packet size: 128 bits (including a header)
- Packet generation interval (T): 0.02 s

(Default values are used for other parameters)

Let each source node transmit X copies of each packet. If at least one of the X copies of a packet is received by the sink, we say the packet is successfully delivered. If, out of the X transmissions, the same packet is delivered successfully more than once, only one packet is considered to be successfully delivered. The average delivery probability P is defined to be the ratio of the total number of successfully delivered packets to the total number of packets generated by source nodes for the duration of the simulation.

Perform simulations for X = 1,…,10 transmissions. Plot P against X. The range of X is [1, 10] and the range of P on the y-axis is [Plowest, Phighest]. Discuss what you have observed from the plot. Did you find an interesting phenomenon? If so, what could be the reason of the phenomenon?

Note that for collecting the data for the plot, the NS-2 trace file will need to be examined by using a script or a programming language (e.g. awk, perl, java, c/c++ or matlab). Do not submit the raw trace files when you submit the project.

Grading Guidelines

We will grade your submission following the guidelines listed below:

[+5] Patch file is submitted (with basic functions, such as send, sendhandler, recv, timer, etc.).
[+15] Simulator can be compiled without errors after applying patch file in Virtual Box.
[+5] .tcl file is submitted (with the given default parameters).
[+15] .tcl works correct (no errors, can generate trace file accordingly, change repeatTimes will get according results).
[+15] The random transmission protocol is correctly implemented (by using the ns2 timer function).
[+15] The trace file for random transmission protocol is correctly obtained (when changing the parameter of repeat times (X copies of the packet), the result is correct as stated in the project description that there will be X copies sent from each source node in each period T).
[+10] A detailed design document.
[+10] The report has reasonable results.
[+5] Explain the simulation results in the report.
[+5] Have reasonable theoretical analysis.

Submission Guidelines

1. Create a patch file named by proj3_[your-ubit-name].patch. To do that, you need to go inside the directory ns-allinone-2.35_modified/ns-2.35/ first and run:

   make clean

   where *ns-allinone-2.35_modified/*, used as an example here, is the directory where you installed NS-2 simulator and implemented your solution. After that, go back to the directory holding *ns-allinone-2.35_modified/* and run:

   diff -ruN ns-allinone-2.35/ns-2.35/mac/ ns-allinone-2.35_modified/ns-2.35/mac/ > proj3_[your-ubit-name].patch

   where *ns-allinone-2.35/* and *ns-allinone-2.35_modified/* are at the same directory level; *ns-allinone-2.35/* is the original directory you got after untaring ns-allinone-2.35.tar.gz, without anything inside got touched after the untar operation. Replace [your-ubit-name] with your real ubit name for submission.

   Note: We recommend you to verify if you can recover your implementation by applying the patch on the original directory before *ns-allinone-2.35/* submission (hint: you can do this through the *patch* command). A failure of applying the patch on the original directory will result in 0 in grading.
2. Include your .tcl files for running simulations. The .tcl file should be named by *proj3_[your-ubit-name].tcl*.
3. Create a design document (which explains which protocol you modified, how you implement your solution in NS-2, and how you run simulations with your script to get your results) and a project report (which clearly shows the results). Pdf files only – no other file format will be accepted. The .pdf files should be named by *proj3_design_[your-ubit-name].pdf* and *proj3_report_[your-ubit-name].pdf*
4. Create a tar file, named by *proj3_[your-ubit-name].tar*, to include your .patch, .tcl, and .pdf files and use the submission command, *submit_cse589*, to submit the tar file.

APPENDIX I

NS-2 Experiment Guidance

1. Install Virtual Box from: https://www.virtualbox.org/wiki/Downloads

2. You can download the Virtual Box appliance through: https://buffalo.box.com/v/cse4589proj3ns2, where you have to log in using your UB account.

3. Import the appliance "VirtualBox_NS2.ova" into your Virtual Box.

   The username is: ns2

   The password is: 1

   You will find NS-2 has already been installed at the path *~ /ns-allinone-2.35*

4. Before you create your only mac protocol, go to *~/ns-allinone-2.35/ns-2.35/*

   run *./configure --with-tcl-ver=8.5* for configuration, and

   run *make clean* to clean off previous .o files in the mac folder

5. Implement your mac protocol through .h and .cc files (e.g., *newmac.cc*, *newmac.h*) and put them to the folder *~/ns-allinone-2.35/ns-2.35/mac*

   Add the name of your corresponding .o file (e.g., *newmac.o*) into the file *~/ns-allinone-2.35/ns-2.35/Makefile*

   You can implement your mac protocol based on the example "mac-simple".

6. Go back to *~/ns-allinone-2.35/ns-2.35/* and run *make* to compile all the source files in the folder mac/ including yours. (before you run make, be sure that the name of your .o file is already added into *Makefile* because the ./configure command may reset the *Makefile*)
7. Check whether your .o file is generated successfully in the folder mac/
8. Implement a .tcl file to do your experiment.
9. Analyze the NS-2 trace file by using a script or a programming language (e.g. awk, perl, java, c/c++ or matlab).

APPENDIX II

Useful web sites

http://www.isi.edu/nsnam/ns/tutorial/
http://www.isi.edu/nsnam/ns/
http://www.isi.edu/nsnam/ns/ns-documentation.html