

Method of Performance Evaluation of WANET Using NS-2

Ms.Dharmista D. Vishwakarma

Department of Elec. Engg., Faculty Tech. & Engg., M.S.University of Baroda, Vadodara

ABSTRACT: Next Generation Network demands to exchanging information at anytime using wireless network with a high data rate. To satisfy the necessity of today's generation the network requires which can be establish anywhere anytime. Wireless ad hoc Network (WANET) can established anytime anywhere without any backbone infrastructure using the mobile nodes. WANET uses wireless mobile nodes that operate as terminals as well as routers in the network, without any centralized administration. The performance of WANET can be measure using the simulation in the NS-2 tool. The NS-2 is a widely used simulator for the performance evaluation of WANET. This paper describes that the creation of wireless network and its performance evaluation using the awk script.

Keywords: AODV, NS-2, Packet Delivery Ratio, Trace file, , WANET

I. INTRODUCTION

Ad-Hoc networking is a concept in wireless networking, which means that users wanting to communicate with each other form a temporary network, without any form of centralized administration. Each node participating in the network acts both as host and a router and therefore is willing to forward packets for other nodes, which makes use of a routing protocol. Each node has to maintain some form of information regarding the network around it, and some algorithm governing the sending and receiving of data packets. This algorithm, together with the supporting information regarding network conditions, is called a Routing Protocol [1, 2].

Characteristics of Ad hoc networks impose a set of new demands on the routing protocol. The most important characteristic is the dynamic topology, which is due to the node mobility, nodes can change position quite frequently, which means that there is a need a routing protocol that quickly adapts to topology changes. Due to power limitation, the routing protocol should try to minimize control traffic, such as periodic update messages, conserving memory, power and bandwidth resources. The main function of the routing protocol is to detect and maintain the optimal route to send data packets between a source and destination via intermediate node(s).

NS-2 is the discrete event network simulator initially developed for the wired technology in 1989 with the development of the real network simulator. Monarch group from the Dept. of computer science at the University of Rice has done the necessary extension to the NS-2 for the simulation of wireless and mobile network. NS-2 uses an uncommon programming technique called split-level programming [3, 4]. In NS-2 C++ is used to provide fast computable parts of network nodes that are glued together by the tcl scripting language. Due to the combination of C++ and tcl script it provides fast computation and flexibility to change the node structures. Section II explains

the scenario of the different simulation tools and structure of Network Simulator tool and the method to create the tcl file for the wireless network section III describes that how to evaluate performance of the any wireless network using the awk script, section IV concludes the paper for the evaluation of performance of wireless network using NS-2.

II. SIMULATION TOOLS

The physical world is modeled in wireless network simulation software to evaluate the behavior and performance of protocols for wireless networks. It is a good compromise between cost, complexity and accuracy of the results. Models can capture reality only to a limited extent. Simulation results will generally not be as accurate as real implementations. There are many simulation tools available for the simulation of wireless networks or the next generation network.

GloMoSim is developed at UCLA (California, USA). It is a scalable and parallel simulator for wireless networks written in PARSEC (PARAllel Simulation Environment for Complex Systems). It consists of a set of library modules which were developed using PARSEC. PARSEC is a C-based parallel simulation language for evaluating a variety of wireless network protocols. It is an open-source network simulator which is based on parallel programming. Glomosim supports TCP Family, IEEE 802.11 CSMA/CA, MAC, UDP, CBR, FTP, HTTP, Fisheye, LAR, ODMRP, WRP, DSR, MACA, Telnet, AODV etc. protocols, wireless. It can emulate the real world more accurately. Easy configurable and fast simulation Many protocols already implemented, the help is very well documented, Dynamic behaviour of the netowrk can be visualize using VT tool. GloMoSim currently solely supports wireless networks, it provides more physical-layer models than ns-2. It may be hard to debug the parallel programs [5].

Qualnet is the commercial network simulation tool not freely available to the user. It is extension of the GLOMOSIM. It is GUI based simulator and easy to create the simulation wireless network. QualNet can support real-time and faster than real-time simulation speed. [6].

OMNet++ is a well-designed simulation package written in C++. OMNET++ is capable of simulating any system composed of devices interacting with each others. The basic entity in OMNeT++ is a module. Modules can be composed of submodules or they can be atomic. Only atomic modules capture the actual behavior. Modules communicate with each other via messages through gates. Gates are linked to each other using connections. A connection can be associated with a propagation delay, error rate and data rate. OMNeT++ does not have an energy model [7].

OPNET was first proposed by MIT in 1986. It is first commercial available simulation tool & now the most widely used GUI based commercial network simulator. OPENT has features of: Modeling & Simulation cycle,

hierarchical modeling, specialized for communication networks OPNET Consists of Three layers: The Network Model, The Node model, The Process model OPNET provides a flexible, high-level programming language with extensive support. It Supports Routing protocols OSPF, RIP, EIGRP, BGP, IGRP, DSR, TORA IS-IS, PNNI, MAC, mobility of nodes, ad hoc connectivity, different application models, Node failure models, modeling of power-consumption etc. Very well documented help is provided. This tool has Relatively high price, complex, takes time to learn, is restriction to its portability [9].

In NS-2 conventional mobile nodes consist of a number of C++ objects which are bond together by tcl-scripts (split-level-programming). Every box (square, ellipse, or trapezoid) is a C++ object. All objects are bond together via tcl-commands. Routing within the mobile gateway is carried out by the new object AGW Agent. It forwards data packets to the inner or outer wireless interface respectively. Two modified routing agents (RTAgent) then process all data and routing packets, in order to perform reasonable ad-hoc behavior [10]. NS-2 supports TCP family, UDP, CBR, FTP, HTTP, Pareto, Exponential protocols, wires, wireless, unicast, multicast routing types.

As a first approximation, the ns-2 could seem a bit difficult to work with. In order to ease its utilization, its structure is explained next. The ns-2 employs two languages. Firstly, the main characteristics of the scenario to simulate are described by OTcl [11, 12]. Secondly, the kernel of ns-2 is specified by C++. It is organized in different folders; each one corresponds to a specific protocol. For example, there is a folder called AODV where all the files that are necessary for the implementation of this protocol are stored. If a user wants to utilize the already developed software that is included in the ns-2 release, he/she will have to know how to specify the scenario by OTcl. However, if some new protocols want to be tested or studied, the inclusion requires the modification of C++ files. In our experience, the modifications are not too troublesome but finding where to modify is usually quite hard. Following are the steps to simulate the wireless network in NS-2 and to creating the tcl file.

- Create an instance of the simulator:
- Setup trace support by opening
- Next create a topology object
- Create the object God, "God (General Operations Director)
- Configuring the nodes
- Create nodes and the random-motion for nodes is disabled
- Give nodes positions to start with
- Setup node movement as the following example
- Setup traffic flow between the two nodes as follows: TCP connections between nodes
- Define stop time when the simulation ends and tell nodes to reset which actually resets their internal network components.
- Finally the generated file for any particular protocol is run and the output files are generated namely out.tr and out.nam and filename.tcl.

To run the simulation type on command prompt in terminal

\$ ns filename.tcl

\$ nam out.nam

Fig 1 and Fig. 2 shows scenario to visualize the behavior of the wireless ad-hoc network with the wireless network that exchange the packet and shows the coverage of each node in.

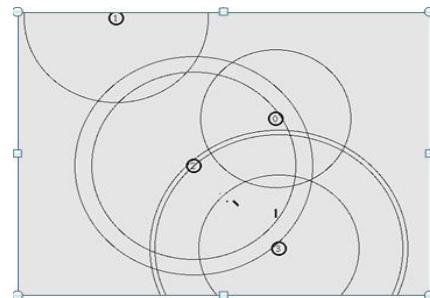


Fig 1: nam output of the wireless network with 4 nodes

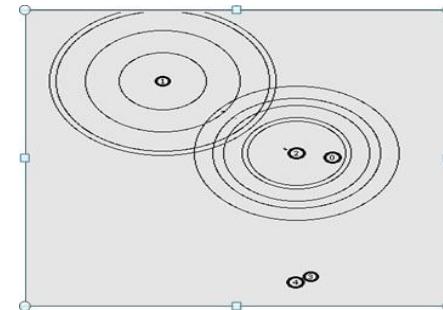


Fig 2: nam output of the wireless netowrk with 5 nodes

III. PERFORMANCE EVALUATION USING NS-2

After finishing of writing tcl file for the network we can visualize the packet communication between the nodes participating in the network when the tcl file is executed. It also generate the trace file which contains the all the data related to packet communication between the nodes. In “*.tr” file each line consists of following information as shown in Fig 3.

Event	Time	From node	To node	Packet type	Packet Size	Flags	Fid	Src addr	Dest addr	Seq number	Pkt id
r : receive (at to_node)										d: drop (at queue)	
+ : enqueue (at queue)										Src_addr: node.port(3.0)	
- : dequeue (at queue)										Dest_addr: node.port(0.0)	
r	1.3556	3	2	ack	40	-----	1	3.0	0.0	15	201
+	1.3566	2	0	ack	40	-----	1	3.0	0.0	15	201
-	1.3566	2	0	ack	40	-----	1	3.0	0.0	15	201
r	1.35576	0	2	tcp	1000	-----	1	0.0	3.0	29	199
+	1.35576	2	3	tcp	1000	-----	1	0.0	3.0	29	199
d	1.35576	2	3	tcp	1000	-----	1	0.0	3.0	29	199
+	1.356	1	2	cbr	1000	-----	2	1.0	31	157	207
-	1.356	1	2	cbr	1000	-----	2	1.0	31	157	207

Fig.3: Format of Trace file

Each line consists of:

- Event Descriptor (+, -, d, r)
- Simulation time (in seconds) of that event
- From Node & To Node, which identify the link on which the event occurred
- Packet type
- Packet size

- Flags (appeared as "-----" since no flag is set). Currently, NS implements only the Explicit Congestion Notification (ECN) bit, and the remaining bits are not used.
- Flow id (fid)
- Source and destination address in forms of "node.port".
- The network layer protocol's packet sequence number. What about UDP?
- The last field shows the unique id of the packet.

Each trace line starts with an event (+, -, d, r) descriptor followed by the simulation time (in seconds) of that event, and from and to node, which identify the link on which the event occurred. The next information in the line before flags (appeared as "-----" since no flag is set) is packet type and size (in Bytes). Currently, NS implements only the Explicit Congestion Notification (ECN) bit, and the remaining bits are not used. The next field is flow id (fid) of IPv6 that a user can set for each flow at the input OTcl script. Even though fid field may not use in a simulation, users can use this field for analysis purposes. The fid field is also used when specifying stream color for the NAM display. The next two fields are source and destination address in forms of "node.port". The next field shows the network layer protocol's packet sequence number. Note that even though UDP implementations do not use sequence number, NS keeps track of UDP packet sequence number for analysis purposes. The last field shows the unique id of the packet.

We need to extract the necessary data out of this file. GREP is a UNIX command to filter a file. The name comes from “search Globally for Lines Matching the Regular Expression and Print them”. It takes a regular expression on the command line, reads the standard input or list of files, and outputs the lines containing matches for the regular expression.

Another option is to use AWK file. AWK is a general purpose computer language. AWK is designed for processing text-based data, either in files or data streams. The name AWK is derived from the surnames of its authors Alfred Aho, Peter Weinberger, and Brian Kernighan.

To run AWK file following commands are needed:

```
awk -f file1.awk file2.txt
awk -f file1.awk file2.txt > out.txt
```

Where,

file1.awk: is a command file
 file2.txt: is a primary input file
 out.txt: is an output file

A typical AWK program consists of a series of lines; each of them is on the form

```
/pattern/ {action}
```

Pattern is a regular expression Action is a command.

- Most implementations of AWK use extended regular expressions by default.
- AWK looks through the input file; when it finds a line that matches pattern, it executes the command(s) specified in action.
- awk or perl scripting language is used to interpret the trace file, it extract data from trace file trace.tr.
- Write the awk script (that is similar to C language) and save it in *****.awk:

Since perl script is very efficient in match data file which has the form of line records [6, 7]. And is very convenient in extract numeric variables from strings, also it is easy to do the computation as C programming language does.

Following is the program segment of awk file written for calculating packet delivery ratio for the trace file for wireless network example.

```
# awk file to find packet delivery ratio.
for (packet_id = 0; packet_id <=highest_packet_id;
packet_id++){
if ((send_time[packet_id]!=0) &&
(rcv_time[packet_id]!=0)){
start = send_time[packet_id];
end = rcv_time[packet_id];
packet_duration = end-start;
} else
packet_duration = -1;
if (packet_duration > 0)
{packet_no++;
total_delay = total_delay + packet_duration; }
}
printf("%d %f %f\n", packet_no, total_delay,
total_delay/packet_no);
}
```

After applying the awk file to the network user can note down the result for different situation of the network and then it will be converted to the graphical format using XGRAPH.

IV. CONCLUSION

The paper describes the different network simulation tools and their special features with the supporting facility of the WANET. It also explains the method for evaluation mechanism developed in NS-2 is really effective for scalable performance test in NS-2. It also could be easy to use for measure the network routing protocols' performance, meanwhile, since it has the fix model of analysis of the trace file, with some minor modification, it will then be apply to measure other kinds of evaluation with the whole network simulation. Since here only some important point and fields are discussed to for the trace file and awk file. However the evaluation scheme is very much tedious job than the simulation of the network.

V. ACKNOWLEDGEMENTS

The project work was carried out at Dept. of Elect. Engg, Faculty of Tech. & Engg. M.S.University of Baroda, Vadodara, Guj, India. Author is thankful to Department of Elect. Engg. for technical help in doing this work.

REFERENCES

- [1] S.-C. Woo, S. Singh, Scalable routing protocol for ad hoc networks, Wireless Networks 7 (5), 2001,513–529.
- [2] Dr. Yogesh Chaba and Naresh Kumar Medishetti, Routing Protocols in mobile Ad hoc Networks- A simulation Study, Journal of Computer Science, JCS Vol.1 No.1, August 2005,pp 83-88.
- [3] The Network Simulator - ns-2, <http://www.isi.edu/nsnam/ns/>, last visited may 26th, 2004.
- [4] “The CMU Monarch Project’s Wireless and Mobility Extensions to NS. <http://www.monarch.cs.cmu.edu/>,”
- [5] L. Bajaj, M. Takai, R. Ahuja, R. Bagrodia, and M. Gerla. GloMoSim: A scalable network simulation environment. Technical Report 990027, UCLA Computer Science Department, 13, 1999

- [6] Scalable Network Technologies (SNT). QualNet.
<http://www.qualnet.com>
- [7] Drytkiewicz, W., S. Sroka, V. Handziski, A. Koepke and H. Karl, A mobility framework for omnet++, in: 3rd International OMNeT++ Workshop, at Budapest University of Technology and Economics, Department of Telecommunications Budapest, Hungary, 2003
- [8] G. F. Lucio, M. Paredes-Farrera, E. Jammeh, M. Fleury, and M. J. Reed. Opnet modeler and ns-2 - comparing the accuracy of network simulators for packet-level analysis using a network testbed. WSEAS Transactions on Computers, 2(3):700–707, July 2003
- [9] Alex Ali Hamidian, A Study of Internet Connectivity for Mobile Ad Hoc Networks in NS 2, Department of Communication Systems, Lund Institute of Technology, Lund University, January 2003
- [10] Introduction to PerlScript.
<http://www.cpan.org/authors/id/M/MS/MSERGEANT/PSIntro.html>
- [11] PerlTutorial. <http://www.comptechdoc.org/independent/perlmanual>