Open-Source Testing Tools for Smart Grid Communication Network

Nguyet Quang Do, Hang See Ong, Lee Chee Lai, Yung Xuen Che, Xing Jui Ong
Department of Electronics and Communication Engineering
Universiti Tenaga Nasional (UNITEN), Kajang, Malaysia
milkydove@yahoo.com, ong@uniten.edu.my, doris_lai87@yahoo.com, yxche828@yahoo.com, alvinoxj@gmail.com

Abstract—**In any smart grid communication implementation, to determine the performance factor of the network, a testing of an end-to-end process flow is required. Therefore, an effective testing tool plays a crucial role in evaluating the performance of smart grid communications. Currently, there are a large number of tools and utilities that are available to study different performance parameters of a network. However, there have been a few studies investigating the testing tools for smart grid communication system. Moreover, most research works done previously focused on only one or two performance metrics such as delay and/or throughput. Plus, diverse communication technologies in smart grid creates major challenges for conventional testing software, for legacy testing programs are designed only for a single communication technology or standard in the network. Therefore, this paper discusses various open-source monitoring and testing tools and determines which set of tools is the most suitable for evaluating the performance of smart grid communication network.**

Keywords-smart grid; network performance evaluation; performance metrics; open-source; network testing tools

I. Introduction

Power is generated at the power plants, transported over the transmission system, and then delivered to the end users via the distribution network. Flow on the distribution network is generally one way. However, in a smart grid scheme, electricity and information flow in two-way direction where end-users can also generate electricity and information to send feedback to the grid [1]. In general, smart grid is an electricity distribution system where information and communication technologies are added to the existing power network. Previously, distributors can only view the power flows until the substations. But now with smart meters, they can see what is going on in their networks down to the end consumers [1].

II. Background

A. Smart Grid Communication Network

In addition to electricity delivery infrastructure, smart grid also comprises of several systems, including SCADA (Supervisory Control and Data Acquisition Systems), AMI (Advanced Metering Infrastructure), DMS (Distributed Management System), AMR (Automatic Meter Reading), DER (Distributed Energy Resource), ADA (Advanced Distributed Automation), GIS (Geographic Information System), CIS (Consumer Information System), etc. Devices that are used to support these systems consist of RTU (Remote Terminal Units), PLC (Power Line Communication) modems, smart meters, data concentrators and so on [2]. An example of smart grid implementation is shown in Figure 1.

A number of technologies were deployed in various smart grid communication implementations. This ranges from wired technologies (PLC, DSL, Fiber optics) to wireless (Mesh RF, ZigBee, Satellite), and cellular (3G, LTE, WiMax and GPRS) [2, 4, 5, 6]. DSL can be used as a backhaul solution for smart grid distribution network [7]. Short-range wireless technology like ZigBee can be used for HAN in indoor environment [5]. WiMax can be deployed for NAN with meshed topology and 4G for WAN [8]. Choice of a technology depends on cost, geographic location, population and accessibility, etc. In most cases, a heterogeneous and hybrid paradigm will be the best option [9], since in smart grid communications, no single technology can cater for the entire power delivery network.

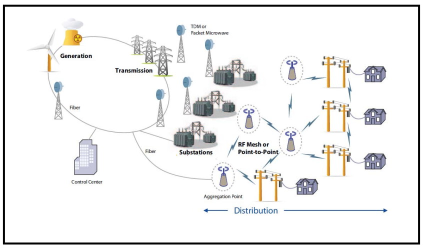

Figure 1. Smart grid network infrastructure [3]

B. Network Testing

In any communication system, it is important for network administrators to identify unexpected behaviors, predict possible problems that might occur in the network and quickly response to solve the problems. This is especially true in a large-scale communication network where a vast number of applications and technologies are implemented. In additions, performance evaluation of such network also involves real-time analysis of data which continuously generated from those devices that make up the network infrastructure. Therefore, network testing is a crucial step in network performance evaluation in order to detect unusual behaviors, anticipate future problems and prevent them from happening.

C. Performance Metrics

In order to evaluate the behavior and characteristics of a communication network, a number of performance metrics need to be considered. . The most common metrics in assessing the network performance are availability, packet loss, latency (or delay), and throughput [10]. Availability is measured in percentage of time when a specific device in the network is up and available for use. Packet loss is one of the

978-1-4799-0285-9/13/$31.00 ©2013 IEEE

important performance metrics because it can identify poor network conditions. When packets are transmitted from source to destination, they are first stored in buffers. A packet is considered as lost if it is discarded when the buffer is full. Latency or delay is the amount of time it takes for data to travel, either one way or round trip, through a communication network. Throughput is the net bit rate at which data is transferred successfully through a communication link in a given period of time. These performance metrics can be measured by a set of network monitoring and testing tools. A discussion of the available software tools is given below.

III. Open-Source Monitoring and Testing Tools

A. Network Monitoring Tools

The main challenge in managing smart grid communication network comes from the fact that different vendors provide different system to manage their own network. In this situation, network managers are required to deploy various management platforms in order to manage the whole system. Nagios, a centralized management system that provides a solution for network operators to handle multi-vendors equipment, appears to be one of the keys to overcome this challenge [11]. The features of Nagios and other available network management systems are described in Table I.

TABLE I. Network Monitoring Tools

Tool	Features		
	Parent-Child Relationship	***Flexible Check Mechanism***	***Availability Monitoring***
Nagios	✓	✓	✓
Zenoss	✗	✓	✓
Cacti	✗	✓	✗
Spiceworks	✗	✓	✗

Nagios [12] is free and open source software designed for Linux. It provides a graphical user interface, allowing network administrators to monitor various devices on the network, such as routers, switches, gateways, and computers [13]. Statistics obtained from Nagios can be used to check host status and to generate availability report for network analysis and evaluation [11].

By arranging the hosts in a hierarchical order using ‘Parent’ configuration, Nagios allows network administrators to view how hosts are connected to each other and the relationship among them. This ‘Parent-child’ relationship is a distinctive function of Nagios that other available software tools do not offer. This function is vital because it helps network managers to determine the root cause of network outages if they happen. In addition, Nagios also provides a sophisticated check mechanism to prevent false alarm from transient problems through different state types, known as hard and soft state.

Like Nagios, Zenoss [14] is an application that is used for managing network and systems. Zenoss also provides a tool for monitoring performance, availability, configuration and events. Compared to Nagios, Zenoss shows a number of disadvantages. First of all, Zenoss does not support “Parent and Child” relationship. Instead, the network configuration in Zenoss, as well as in Cacti [16] and Spiceworks [17], is host dependency. Host dependency is different from parent/child relationship because it does not represent how the network is set up. It is merely a dependency of one host upon another. Secondly, Zenoss is unable to distinguish between hard and soft state. As a result, users cannot configure alerts as flexibly as they can with Nagios.

Cacti [15] is a network management tool that is most suitable for data storage and graphing functionality. With advanced graph templates and multiple data acquisition methods, Cacti is mainly designed for monitoring SNMP hosts. Unlike Cacti, Nagios is primarily designed for status monitoring and its main focus is on availability check. In Nagios, availability percentages are computed automatically. Thus, Nagios has more advantages than Cacti in monitoring the availability status of various devices in smart grid communication network.

Spiceworks [16] is a network management and monitoring software that provides PC inventory, Help Desk and reporting solution for handling IT in small and medium-sized businesses. Even though Spiceworks offers easy and fast installation, it has limited scalability and can be slow on larger networks. Like Cacti, the primary function of Spiceworks is managing network performance. Hence, it does not provide the same depth of availability monitoring as other software tool like Nagios. Another factor that makes Spiceworks an unappealing option is the presence of graphical advertisement on the Web interface. This sometimes can be undesirable by network managers as they want to add their own icons or logo.

B. Network Testing Tools

Ping [17] is one of the most common tools used to test the availability and reachability of a host on an IP network. It acts as an administrator utility which can identify if the targeted host is reachable or not. Ping operates by sending ICMP packets to the destination host and wait for an ICMP response. During the process from transmission to reception, the time taken is measured as round-trip time, and packet loss is recorded. The ping packet is considered lost if it does not receive back a response or it has timed-out. Once the ping command has been successfully released, the obtained results will be summarized in the ping statistics. The number of packets sent, packets received, percentage of packets lost and round-trip time will be displayed in the output.

In terms of checking availability, measuring packet loss and latency, there are a number of tools available that have similar functions as Ping. These tools are summarized in Table II.

Like Ping, Fping [18] is a program that uses ICMP packets to determine if a host is up. Fping can also measure packet loss and latency. The only difference is that Fping can ping multiple hosts with a single command. Echoping [19] is a utility for measuring TCP/UDP latency. It is used to test the performance of a remote host by sending TCP/UDP or HTTP requests. Hence, Echoping is suitable for testing Web servers and measuring Web performances.

TABLE II. NETWORK TESTING TOOLS

Tool	Features		
	Availability	*Packet Loss*	*Latency*
Ping	✓	✓	✓
Fping	✓	✓	✓
Echoping	✓	✗	✓
Traceroute	✓	✗	✓
Sting	✓	✓	✗
Zing	✓	✓	✗

Traceroute [20] displays the route/path for an ICMP packet to travel from source to destination host. Like Echoping, Traceroute emphasizes on communication latency by measuring the transit delay of packets along the path. Sting [21] and Zing [22] [23] can also be used to check the availability of a host on a network. However, their main focus is on packet loss measurement. Sting measures packet loss rate in both forward and reverse directions. Zing only measures end-to-end packet loss in one direction between two hosts.

C. *Throughput Measuring Tools*

A number of tools have been developed over the years for measuring link throughput. The features of these tools are summarized in Table III. Most of the tools support TCP/UDP protocols and can run on Linux operating systems. Some tools like Iperf, Uperf, Ttcp, Nttcp, and Nuttcp require server-client installation, while others do not.

TABLE III. THROUGHPUT MEASUREING TOOLS

Tool	Features		
	Protocol	*Operating System*	*Server-Client*
Iperf	TCP, UDP	Linux, Windows	✓
Netperf	TCP, UDP, DLPI, SCTP	Linux, Unix	✓
Uperf	TCP, UDP, SCTP, SSL	Solaris, Linux, FreeBSD, Mac	✗
Bing	ICMP	Linux, Windows	✗
Pathchar	UDP, ICMP	Solaris, Linux, FreeBSD	✗
Thrulay	TCP, UDP	Linux	✗
Ttcp	TCP, UDP	Linux, Unix, Windows	✓
Nttcp	TCP, UDP	Linux, Windows	✓
Nuttcp	TCP, UDP	Solaris, Linux, Unix, Windows	✓

Iperf [24] is used as a tool to measure bandwidth and determine the quality of a specific link on a communication network. Iperf requires server and client paradigm, i.e. it must be installed at both the server and client machines. Iperf runs on Windows and Linux, and supports both TCP and UDP protocols. Netperf [25] is a software tool that is used to evaluate the performance of various network protocols. It is utilized to measure uni-directional throughput and end-to-end latency. Netperf also provides CPU utilization measurement. Its main focuses are on request/response performance, transferring data using compression, blocking and buffering method. Uperf [26] is a unified performance tool for network that supports modeling of various network patterns. It allows network administrators to measure bandwidth and latency (both uni and bi-directional) of multiple protocols. Unlike Iperf and Netperf, Uperf is a standalone application that does not require sever-client installation.

Bing [27] is a point-to-point bandwidth-measuring tool built on top of Ping. It calculates the link throughput based on the round trip time of ICMP packets with different sizes. Pathchar [28] is an alternative to Bing. Pathchar discovers the bandwidth of each link along a path from a start node to an end node. It sends a number of UDP packets with different sizes to each hop, and evaluates bandwidth, latency, average queue and loss rate across the link based on the RTT distribution. Thrulay [29] measures the capacity, delay, throughput and other performance metrics of a network by using TCP or UDP bulk transfer. Like Uperf, Bing, Pathchar and Thrulay use software at only one end of the path to measure the throughput of a network link.

Ttcp [30] is another utility program for measuring network throughput. It uses the UDP or TCP protocol to time the transmission and reception of data. Ttcp allows measurement at the remote end of a UDP transmission. Nttcp [31] measures the transfer rate on a TCP, UDP or UDP multicast connection. Nuttcp [32] determines the raw TCP or UDP network layer throughput. Nuttcp is developed based on Nttcp, which in turn is the enhancement of Ttcp.

D. *Traffic Generating Tools*

Smart grid is a complex network which implements diverse applications, information and communication technologies. Traffic generated from numerous devices in smart grid can cause certain problems to the network. Due to the heterogeneous and hybrid communication paradigm of smart grid, the characteristics of traffic carried in the network are not clearly known [33]. Therefore, a traffic generator is required to carry out stress test in order to predict the network behavior and evaluate its performance.

In recent years, more and more different types of traffic generators were created depending on different network protocols and according to different layers in the OSI model. The characteristics of several traffic generating tools are described in Table IV.

PackETH [34] is a packet generating tool for Ethernet. PackETH provides a friendly GUI with various options for users to generate packets over the Ethernet link. It supports several protocols, including Ethernet II, UDP, TCP, ICMP, IPv4, IPv6 and so on. PackETH provides a number of features to support sending sequence of packets such as number of packets, delay between packets, UDP payload, etc.

Traffic [35] is a network traffic generator used to test the effects of traffic on a network. This tool does not provide throughput or response time measurement. It supports several operating systems, including Linux, FreeBS and Microsoft

Windows. It provides a Graphical User Interface (GUI) to generate TCP/UDP traffic from client to server to test firewalls/routers.

TABLE IV. TRAFFIC GENERATING TOOLS

Tool	**Features**			
	GUI	***User Space***	***Kernel Space***	***Server-Client***
PackETH	✓	✓	✗	✗
Traffic	✓	✓	✗	✗
MGEN	✗	✓	✗	✗
Packgen	✗	✓	✗	✗
PacGEN	✗	✓	✗	✗
Mtools	✗	✓	✗	✗
Rude&Crude	✓	✗	✓	✓
Kute	✗	✗	✓	✓

MGEN [36] is a tool to generate real-time traffic patterns by providing emulation of unicast and/or multicast UDP and TCP/IP traffic. MGEN allows administrators to test and measure the performance of an IP network using different communication protocols such as TCP and UDP/IP. The data obtained from MGEN log can also be used to calculate communication delay, packet loss rates, and throughput.

Packgen [37] is a network packet generator with given bandwidth that supports both UDP and TCP flows. This utility sends packets at different time interval based on the bandwidth and the size of packets to generate. Packgen can generate various flows of data described by their names, destination, bandwidth, packet size and time range.

PacGen [38] is another packet-generating tool for Linux operating system. This tool provides custom packets with different configurable layers (Ethernet, IP, TCP and UDP) together with custom payloads.

Mtools [39] contains a collection of tools used to generate UDP traffic in Linux and FreeBSD operating systems. It uses One-way-delay Meter to send UDP packets to a host in the network and measure their transmission time, and use Round-trip-time Meter at the receiver side to send these packets back to the sender.

RUDE&CRUDE [40] is a traffic generator running on Linux, Solaris and FreeBSD. RUDE (Real-time UDP Data Emitter) is a program located at one end of the network that generates traffic. CRUDE (Collection for RUDE) is located at the other end to collect the traffic generated from RUDE. These programs can support only UDP traffic. Both RUDE and CRUDE utilize the kernel to achieve the maximum performance.

KUTE [41] is another traffic generator developed in the kernel that can generate UDP packets. It consists of two Linux kernel modules: a sender and a receiver. A sender module sends packets directly to the hardware driver and can be used to test network devices such as routers or switches. A receiver module receives the packets from the driver, count the number of packets, and measure the packet inter-arrival times. It is used together with the sender module for various performance testing.

IV. NETWORK TESTING TOOLS FOR SMART GRID

In smart grid communication implementation, no single technology can cater for the entire power delivery network. Similarly, no single software tool can be used to test the whole communication network and obtain all performance parameters. Therefore, a set of testing tools are required and the justification for each tool will be provided in the following discussion.

A. Nagios

Nagios is chosen for this particular project due to the fact that its functions are suitable for monitoring the availability status of numerous devices in the smart grid communication system. Basic features of Nagios allow it to provide more detailed information about the availability of a selected host compared to other network management software tools. For example, Zenoss is an availability monitoring tool like Nagios but it does not support parent/child relationship and hard/soft state. With advanced graphical functionality, Cacti and Spcieworks focus more on managing network performance than availability monitoring as Nagios does. Compared to other network management systems, Nagios offers a number of advantages. Initially developed for servers and application monitoring, Nagios is now widely used to monitor network availability. It is an open source software tool that includes all necessary functions to perform availability monitoring. Nagios plugins are simple to develop, using languages such as Bash, Perl, and Python. It is scalable and possible to deploy a hierarchy of communication node to provide consolidated views of network and system availability. In short, Nagios proves to be particularly useful for monitoring the availability states of various devices in communication networks. It serves the monitoring purpose, especially in smart grid context, by providing necessary information to perform availability test.

B. Ping

There are a number of tools available for network testing such as Fping, Echoping, Traceroute, Sting, and Zing. Fping provides similar functions as Ping but it requires installation prior to operation. Sting and Zing mainly focus on measuring packet loss, while Echoping and Traceroute focus on communication latency. Unlike these tools, Ping can be used to check the availability, measure both packet loss and latency of a selected host on an IP network.

Ping is selected for this project because it is one of the most common network testing tools. It is available and supported by most communication devices and operating systems. It is a built-in software tool in most communication devices and normally comes with the operating system. Therefore, it does not require extra installation or configuration. Ping tests can be performed without having a computer installed on each end of the link, which in turn saving time and labor in troubleshooting and diagnostics.

C. Iperf and Bing

Among the throughput measuring tools that require sever-client installation, Iperf was chosen as a software tool to conduct throughput tests because of several reasons. First of all, Iperf is a simple tool that focuses only on network transfers, thus measuring only network performance and not combining network, disk, and application performance like other tools. For example, Netperf includes CPU utilization measurement which is not required in this particular project. Nuttcp has additional features that are not the primary focus in the testing framework such as user, system and wall-clock time. Secondly, Iperf is under active development, whereas Ttcp and Nttcp are not well developed.

Besides Iperf, Bing is another throughput measuring tool. Unlike Iperf, Bing does not require client-server installation. It can be easily installed on Windows and Linux operating systems. After being set up at the test server, Bing can measure point-to-point throughput of a selected link. Bing requires only one end to do the measurement. This is particularly useful when installing a software tool at one end of the link is not applicable.

D. Network Traffic Generator

There are several software tools available for generating network traffic such as PackETH, Traffic, MGEN, Packgen, PacGEN, Mtools, Rude&Crude, and Kute. All of the aforementioned traffic generators were designed for Linux operating system. However, these traffic generating tools still have certain limitations. Therefore, a traffic generator is implemented in this project in order to overcome these problems. The fundamental idea of implementing a new traffic generator arises from the lack of the available ones. For example, some traffic generators like PackETH, Traffic or Rude&Crude have a GUI that occupies a lot of memory storage. Meanwhile, the traffic generator running on single board computer does not need a GUI, which can reserve more space in the memory and save execution time.

In addition, most of the existing traffic generators are user-based programs. User space traffic generation uses socket Application Programming Interface (API) available for a given system. It is more popular, simpler and easier to debug. However, in terms of speed and timing, kernel space traffic generators are more desirable since the overall performance can be improved in terms of sending rates. Therefore, the traffic generator implemented in this project is developed from the Linux kernel in order to bypass the overhead of a user-application, allowing it to achieve the best performance and utilize maximum network bandwidth.

Programming in the kernel is more complicated than in the user space. As a result, only a few traffic generators are built for the kernel such as Rude&Crude and KUTE. Nevertheless, Rude&Crude and KUTE require server-client configuration, i.e. they must be installed at both ends of a tested link. It is redundant in this particular project because there is no need for capturing the generated data. Besides, KUTE is not well maintained and incompatible with later Linux kernel [42].

In short, to serve the testing purposes of this project, a traffic generator must not have a GUI, run on kernel space and does not require sever-client configuration. So far none of the existing tools can provide the exact combination of these desired features. Consequently, a new traffic generator is implemented in this project in order to meet the specific objectives and requirements. In this project, the traffic generator is used to generate data packets in order to stress the smart grid communication network. A number of traffic generators were placed a several power substations in order to determine the traffic capacity of smart grid communication network.

In short, a set of testing tools is required to measure various performance parameters of smart grid communication network. These software tools are summarized in Table V.

TABLE V. SET OF TESTING TOOLS

No	*Parameter*	*Tool*
1	Availability	Nagios
2	Packet loss and Latency	Ping
3	Throughput	Iperf, Bing
4	Stress test	Traffic generator

V. SMART GRID TEST-BED EXPERIMENTATION

An experimental smart grid test-bed, namely UNITNE-TNBR, was set up to study the feasibility of deploying different communication technologies in smart grid. To carry out this study, a communication network was built on top of the power system and is supported by three technologies, including WiMax, Mesh RF and Power Line Communication (PLC). A performance testing framework was proposed to evaluate the performance of smart grid communications. This framework includes the combination of the testing tools discussed above.

TABLE VI. STRESS TEST

Technology	**Parameters**			
	Traffic	***Duration***	***Packet Loss***	***Delay (ms)***
Mesh RF	28 Mbps	< 450 ms	0%	0.7-23.5
		450ms – 1s	0%	22.56-42.07
		> 1s	100%	_
WiMax	370 Kbps	30.5 s	5.5%	1142.4
PLC	12.6 Mbps	6.09 s	7.0%	83.78

Table VI shows an example of the results obtained from the network tests. Traffic generators were used to test different networks, including Mesh RF, WiMax, and PLC. The results obtained from the stress test show that the maximum capacity of traffic in the Mesh RF network occupies 51.85% of the rated bandwidth. This ensures critical traffic such as SCADA control messages can be issued without interruption. The results indicate that the current network capacity is able to support around 25,000 meters and 200 RTUs.

VI. CONCLUSION

In conclusion, this paper provides a comprehensive study on various open-source testing tools for measuring a set of performance metrics and evaluating the behavior of smart grid communications. The evaluation criteria for the communication network include availability, packet loss, latency, and throughput. The software tools used to obtain these parameters are Nagios, Ping, Iperf, and Bing. In addition, a traffic generator was also used to investigate the effect of increasing network traffic on the performance of smart grid communication system.

ACKNOWLEDGMENT

I would like to thank all the lecturers and students of Universiti Tenaga Nasional who have been involve directly and indirectly in this research. I would also like to extend gratitude to TNBR engineers and technicians for their cooperation in allowing this research to take place.

REFERENCES

[1] N. Al-Hariri et al., Smart Grids for Dummies, Logica Limited ed., West Sussex: John Wiley & Sons Ltd, 2010.

[2] A. Zaballos, A. Vallejo, and J. Selga, “Heterogeneous communication architecture for the smart grid,” IEEE Network, vol. 25, pp. 30-37, September-October 2011.

[3] JDS Uniphase Corporation, JDSU smart grid solutions, Communication test & measurement, retrieved on 3 January 2012, available at http://www.jdsu.com/ProductLiterature/smartgrid_pb_tfs_tm_ae.pdf

[4] P. P. Parikh, M. G. Kanabar, and T. S. Sidhu, “Opportunities and Challenges of Wireless Communication Technologies for Smart Grid Applications,” Pow. En. Soc. Gen. Meet., pp. 1-7, July 2010.

[5] V. C. Gungor et al., “Smart grid technologies: communication technologies and standards,” Proc. IEEE Trans. Ind. Info., vol. 7, pp. 529-539, November 2011.

[6] X. Fang, S. Misra, G. Xue, and D. Yang, “Smart grid – the new and improved power grid: a survey,” Proc. IEEE Comm. Sur. Tu., vol. 14, pp. 944-980, Fourth quarter 2012.

[7] D. M. Laverty, D. J. Morrow, R. Best, and P. A. Crossley, “Telecommunications for smart grid: backhaul solutions for the distribution network,” Proc. IEEE Pow. En. Soc. Gen. Meet. pp. 1-6, July 2010.

[8] M. Kim, “A survey on guaranteeing availability in smart grid communications,” Proc. 14th Int. Conf. Ad. Comm. Tech., pp. 314-317, February 2012.

[9] Q. Bin, “Next generation information communication infrastructure and case studies for future power systems,” Doctoral dissertation, Virginia Polytechnic Institute and State University, 2002.

[10] Drysdale et al., “Method and apparatus for performing service level analysis of communications network performance metrics”, United States Patent, May 2000.

[11] A. D. Kora, and M. M. Soidridine, “Nagios-Based Enhanced IT Management System,” Int. J. Eng. Sci. Tech., pp. 1199-1207, March 2012.

[12] Nagios, retrieved on 18 February 2013, available at http://www.nagios.org/

[13] T. Shimomura, Q. L. Chen, N. S. Lang, and K. Ikeda, “Integrated Laboratory Network Management System,” Proc. 7th WSEAS Int. Conf. App. Info. Comm., pp. 188-193, August 2007.

[14] Zenoss – Open Source IT Management, retrieved on 18 February 2013, available at http://community.zenoss.org/index.jspa

[15] Cacti – The Complete RRDTool-based Graphing Solution, retrieved on 18 February 2013, available at http://www.cacti.net/

[16] Spiceworks, retrieved on 18 February 2013, available at http://www.spiceworks.com/

[17] Wikipedia: The Free Encyclopedia, “Ping (networking utility),” retrieved on 17 March 2012, available at http://en.wikipedia.org/wiki/Ping_(networking_utility)

[18] R. J. Schemers, RL B. Morgan, and D. Papp, “Fping(8) – Linux man page,” retrieved 2 February 2013, available at http://linux.die.net/man/8/fping

[19] S. Bortzmeyer, “Echoping(1) – Linux man page,” retrieved 2 February 2013, available at http://linux.die.net/man/1/echoping

[20] Traceroute(8) – Linux man page, retrieved on 2 February 2013, available at http://linux.die.net/man/8/traceroute

[21] S. Savage, “Sting: a TCP-based network management tool,” Proc. 2nd Conf. USENIX Symp. Int. Tech. Syst., vol. 2, pp. 7-7, October 1999.

[22] J. Sommers, P. Barford, N. Duffield, and A. Ron, “A geometric approach to improving active packet loss measurement,” IEEE/ACM Trans. Net. J. Mag., vol. 16, pp. 307-320, April 2008.

[23] S. Kanugula, K. Srinivas, T. P. Shekhar, and S. K. Konga, “Measuring end-to-end packet loss characteristics with probes,” Int. J. Comp. Sci. Man. Res., vol. 1, pp. 641-647, November 2012.

[24] Iperf, retrieved on 20 February 2013, available at http://iperf.sourceforge.net/

[25] R. Jones, “Netperf,” retrieved on 20 February 2013, available at http://www.netperf.org/netperf/

[26] Uperf – a network performance tool, retrieved on 20 February 2013, available at http://www.uperf.org/

[27] Bing, retrived on 21 February 2013, available at http://fgouget.free.fr/bing/bing_src-readme.shtml

[28] V. Jacobson, “Pathchar,” retrieved on 20 February 2013, available at http://www.caida.org/tools/utilities/others/pathchar/

[29] S. Shalvunov, “Thrulay,” retrieve on 20 February 2013, available at http://e2epi.internet2.edu/thrulay/

[30] Ttcp – Linux man page, retrieved on 20 February 2013, available at http://linux.die.net/man/1/ttcp

[31] E. Bartel, “Nttcp(1) – Linux man page,” retrieved on 20 February 2013, available at http://linux.die.net/man/1/nttcp

[32] Nuttcp Development Team, “Nuttcp,” retrieved on 20 February 2013, available at http://www.nuttcp.net/nuttcp/Welcome%20Page.html

[33] Z. Fan, G. Kalogridis, C. Efthymiou, M. Sooriyabandara, M. Serizawa, and J. McGeehan, “The new frontier of communications research: smart grid and smart metering,” Proc. 1st Int.l Conf. En.-Eff. Comp. Net., pp. 115-118, April 2010.

[34] PackETH, retrieved on 25 November 2012, available at http://packeth.sourceforge.net/sourceforge/Home.html

[35] Traffic, retrieved on 25 November 2012, available at http://robert.rsa3.com/traffic.html

[36] Multi-Generator (MGEN), Networks and communication systems branch, retrieved on 25 November 2012, available at http://cs.itd.nrl.navy.mil/work/mgen/index.php

[37] Packgen-Network packet generator, retrieved on 25 November 2012, available at http://packgen.rubyforge.org/files/README.html

[38] PacGen, retrieved on 25 November 2012, available at http://pacgen.sourceforge.net/

[39] Mtools, retrieved on 25 November 2012, available at http://www.grid.unina.it/grid/mtools/

[40] RUDE & CRUDE, retrieved on 25 November 2012, available at http://rude.sourceforge.net/

[41] KUTE—Kernel-based rraffic engine, retrieved on 25 November 2012, available at http://caia.swin.edu.au/genius/tools/kute/

[42] V. C. Valgenti, “Dynamic Content Generation for Evaluation of Network Applications,” Phd. Dissertation, School of Electrical Engineering and Computer Science, Washington State University, May 2012.