FlowPing - The New Tool for Throughput and Stress Testing

Ondrej VONDROUS, Peter MACEJKO, Zbynek KOCUR

Department of Telecommunication Engineering, Faculty of Electrical Engineering,
Czech Technical University in Prague, Technicka 2, 166 27 Prague, Czech Republic

ondrej.vondrous@fel.cvut.cz, peter.macejko@fel.cvut.cz, zbynek.kocur@fel.cvut.cz

DOI: 10.15598/aeee.v13i5.1497

Abstract. *This article presents a new tool for network throughput and stress testing. The FlowPing tool is easy to use, and its basic output is very similar to standard Linux ping application. The FlowPing tool is not limited to reach-ability or round trip time testing but is capable of complex UDP based throughput stress testing with rich reporting capabilities on client and server sides. Our new tool implements features, which allow the user to perform tests with variable packet size and traffic rate. All these features can be used in one single test run. This allows the user to use and develop new methodologies for network throughput and stress testing. With the FlowPing tool, it is easy to perform the test with the slowly increasing the amount of network traffic and monitor the behavior of network when the congestion occurs.*

Keywords

Ping, stress testing, throughput, variable traffic rate.

1. Introduction

In this article, we would like to introduce the new tool for network throughput and stress testing. We name our new tool FlowPing [1].

The main difference between our tool and other commonly used open source tools is that our tool allows us to examine the behavior of networks in reaction to the dynamic change of generated traffic amount.

The unique feature of our tool is the ability to generate increasing or decreasing traffic data flow. The tool is capable of generating complex variable traffic flows because it is possible to read complex test scenario from a file.

Flowping tool utilize UDP (User Datagram Protocol) [2]. One of the reason to use UDP protocol is that it is not a priory blocked or influenced by traffic control policies on gateways and firewalls like standard ping application which utilize ICMP (Internet Control Message Protocol) protocol [3]. Another reason for using UDP protocol is the fact that our other application [4] uses UDP as the underlying protocol for communication and thus we can easily compare these application measurement results with FlowPing results.

The main reason for developing and publishing the FlowPing tool under open source license is the fact that there are many open source tools such as Iperf2, iperf3, Ostinato, Seagull, pacgen, Bittwist, Nping, CLAudit [5] and others. Appointed tools allow the user to generate general traffic flow to stress test network, but we do not know any open source tool, which is capable of generating variable increasing or decreasing traffic flows. In addition, our tool has some unique features such as high precision time reporting. On the other hand, there are commercial tools (such as Ixia IxChariot) but they are too expensive for general use. That is also why we decided to provide this tool free of charge under Creative Commons 3.0 BY-NC-SA License.

2. FlowPing Tool

This tool is aimed to be used especially for latency and network throughput stress testing. The FlowPing tool can be run on several platforms like x86-32, x86-64 and ARM.

This tool is able to send and to receive UDP packets in the very similar way as the standard ping application handles ICMP packets. The basic output of this tool is similar to well known ping application. That is why the usage of FlowPing tool is very intuitive. On the other hand this tool has more features than standard ping tool and it is not limited only for network reachability testing but it covers large variety of network throughput and stress tests.

The great advantage of the FlowPing tool is the possibility of writing prescription of test into file. This is plain text file (it is possible to use white spaces, comas or semicolons as the field separators) with simple and intuitive structure (see Fig. 1). It allows users to define very complex tests therefore user can save large amount of time when performing network stress tests. That is due to the fact that many different tests can be combined into one complex test and it is not necessary to run every single test manually.

```
# Time[s]  Bit_rate[kbps]  Packet_size[B]
0          128             64
15         128             64
30         256             64
30         64              500
45         64              500
60         256             24
60         512             24
75         256             24
90         256             24
105        128             24
120        256             24
135        256             64
135        640             1472
150        640             1472
150        512             1000
165        128             1000
180        128             1000
```

Fig. 1: FlowPing - test configuration file data structure.

The extended output can provide advanced statistics such as immediate sending and receiving bit rate, inter-packet intervals and time stamps with nanosecond resolution. The FlowPing tool is also capable of storing and displaying statistics in CSV format for further data processing.

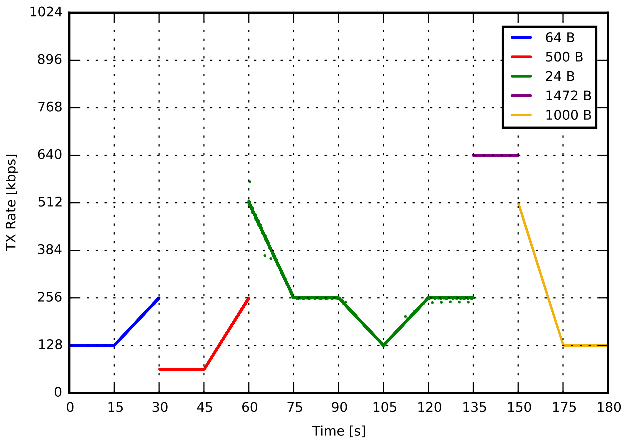

Fig. 2: FlowPing - variable traffic flow.

The FlowPing tool have many other interesting features as described on applications homepage [1].

3. Traffic Generator Performance and Precision

The FlowPing tool is capable of running tests with variable inter-packet intervals resulting in a variable rate of data flow (increasing or decreasing). The tests with variable packet size are also available. Figure 2 shows these possibilities of variable flow definition including packet size. It is also possible to define the variable flow test on the command line (only simplified scenario).

The FlowPing tool allows user to use special modes to increase traffic generator accuracy. It is possible to select passive waiting mode which is effective in way of CPU utilization, but with lower accuracy (similar to Iperf 2 tool). If higher accuracy is important it is possible to use busy loop mode (active waiting) which utilizes at least 100 % of one CPU core for duration of test. The traffic generator stability and accuracy are greatly increased in such case.

The FlowPing tool also implements the possibility to compute all packet intervals before test starts. This approach combined with busy loop mode greatly increases packet timing precision and overall performance. This feature is internally associated with other feature which stores all output in memory and writes results after the test is finished. These features ensure maximum possible performance and timing precision of FlowPing tool. It is possible to generate traffic rates up to 1 Gbps on standard Linux machine (application runs in user space on Intel i5-2500k with packet size of 1470 B).

We have performed many tests to study the FlowPing tool performance. We have mainly focused on traffic generator precision and stability. The FlowPing tool is optimized to achieve comparative or even better results than well-known and widely used tools such as the Iperf 2 [6] or the iperf 3 [7].

The results of traffic flow generator stability test (test setting were as follows: Flowping [-b 5000 -s 160], Iperf 2 & 3 [-u -b 5M -l 160]) of FlowPing, Iperf 2 and iperf 3 tools are shown in Fig. 3. From top left to bottom right there is comparison of packet delay stability of Flowping tool in standard mode, FlowPing tool with packets interval computed before test start (utilizing busy-loop waiting), Iperf tool version 2 and finally totally redesigned iperf at version 3.

For validation purposes we used tcpdump utility to capture packets directly on outgoing interface to ensure objectivity of measurements when dealing with traffic generator stability tests. This approach allows us to compare results from FlowPing tool with results from other tools such as Iperf version 2 or iperf version 3.

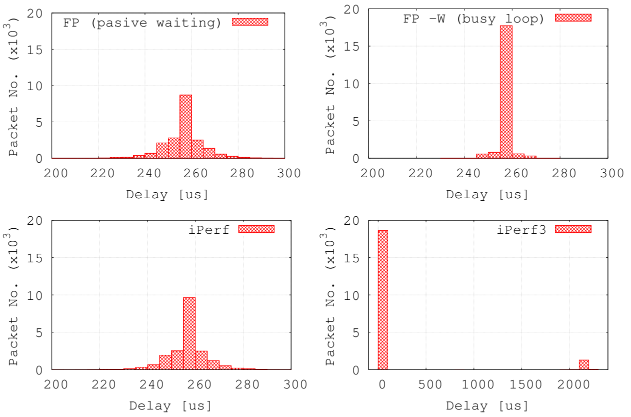

Fig. 3: Packet delay stability - FlowPing vs. Iperf 2 vs. iperf 3.

The results show that the precision of traffic generator is mainly determined by waiting mode used. Results also showed that traffic generator was completely redesigned in iperf3. Iperf3 generates packets in bursts. At first it generates packets at maximum speed for the specific time period and then it adds longer delay to create average data rate.

The traffic generator precision is also influenced by current system load of machine where the FlowPing tool is running.

4. Variable Flow Stress Testing

We found feature of generating variable traffic flow essential for measurement of important network parameters.

Methods utilizing variable traffic rates can simplify network testing in some cases. The measurements utilizing variable data rates make detection of traffic shaping and policing easier as opposed to methods used in article: "End-to-end detection of isp traffic shaping using active methods" [8] where sustained date rate was used for active network detection of traffic policing.

The variable flows can be very valuable source of data for network parameter estimation. It is possible to use it for observing the dynamic behavior of network under changing load. It is also possible to detect traffic engineering methods such as network buffering along network path, traffic shaping or traffic policing.

Methods for network throughput and stress testing based on increasing and decreasing the amount of network traffic provide valuable information about networks under changing load. This approach allows us to observe dynamic network behavior in reaction to small traffic amount changes. This approach is very useful for observing UE (User Equipment) resource allocation in mobile wireless networks as a reaction to traffic amount change.

With FlowPing tool, it is not necessary to repeat measurement with different settings again and again because this tool allows user to create complex measurement scenario and put it into single configuration file. When finding congestion point of network, especially this method is very effective. Utilizing variable flows for network stress testing can be very effective in situation when it is necessary to find the exact amount of traffic which causes network congestion. In first run, it is possible to perform very quick test and find approximately the traffic amount which causes congestion and in second run it is possible to target the range of test on this congestion point and perform precise measurements just around this congestion point of the network.

In following sections we would like to present several cases when usage of variable flow in stress testing is useful.

4.1. Congestion Point Detection

This test is used to find the exact amount of traffic flow when congestion occurs. It is very convenient to use this type of network throughput test in a situation when the parameters of the network are unknown. On the chart in the Fig. 4 is shown the difference between generated traffic flow and received traffic flow. By comparing sending and receiving traffic, RTT and loss rate you can very easily find exact throughput and the point where congestion occurs. It is also possible to detect if some mechanism of traffic control was used as shown in the subsection 4.2.

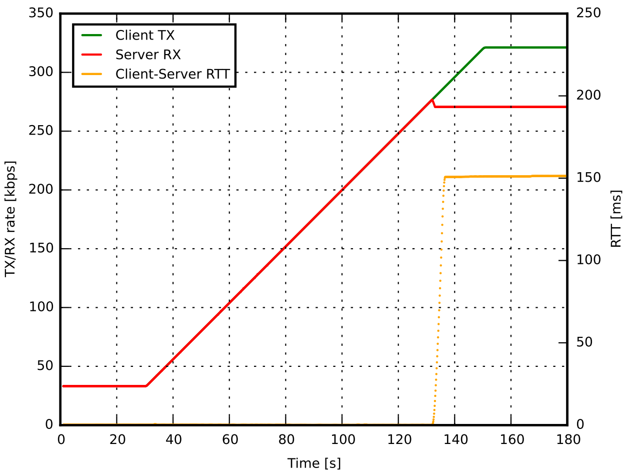

Fig. 4: Congestion point detection.

The advantage of using FlowPing which utilizes a stateless protocol such as UDP for throughput stress testing is the ability to find real maximum throughput of the transport channel. The maximum throughput is indicated either by data flow or by a dramatic change of RTT and packet loss.

In case that we use state full protocol such as TCP the results will be influenced by packet re-transmissions in case of packet loss, by congestion control and congestion avoidance mechanisms.

4.2. Traffic Control Mechanism Detection

In this test scenario, we used traffic generated by FlowPing tool. We observe the behavior of network traffic on the outgoing interface of the router which was configured to policy network traffic to Committed Informational Rate (CIR) of 5 Mbps with allowed Committed burst size (Bc) of 256 kB.

At first let's take a closer look on curves shown on Fig. 5. There is a traffic rate drop at the time of 12.43 s (these values were obtained directly from result dataset). This indicates that some method of traffic shaping or traffic policing was used in this case in conjunction with allowed burst of traffic. It is also possible to determine which method was used for traffic conditioning. This can be simply obtained from analysis of RTT, when significant increase of RTT indicates that traffic shaping method was used because traffic policing does not use packet buffering.

Target data rate can be obtained as a current traffic rate just after the traffic rate drop is detected and final data rate is stabilized. Finally when we have target rate value we can compute value of committed burst size as represented by Eq. (1).

Where TR_f is the Final Target Rate enforced by traffic conditioning, PS is the packet size, t_1 is the time when increasing traffic bit rate is equal to target bit rate and finally t_2 is the time when traffic rate drops to target rate level.

$$Bc = \sum_{t=t_1}^{t_2} PS - \frac{TR_f \cdot (t_2 - t_1)}{8} \text{ [B]}, \tag{1}$$

$$\begin{aligned} Bc &= \sum_{t=6.37}^{12.43} 500 - \frac{5 \cdot 10^6 \cdot (12.43 - 6.37)}{8} = \\ &= 252000 \text{ [B]}. \end{aligned} \tag{2}$$

The results of Eq. (2) show very accurate value of Bc used in policing configuration. As shown and measured on corresponding Fig. 5.

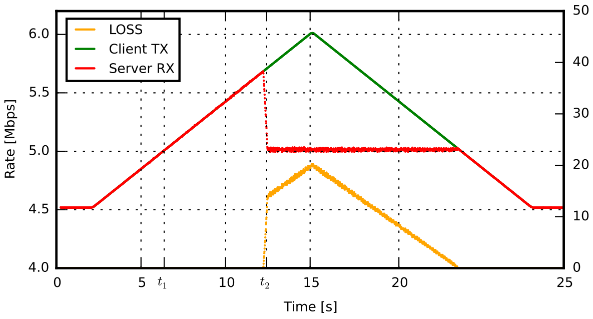

Fig. 5: Traffic policing detection.

4.3. Real Network Measurement - Mobile Network

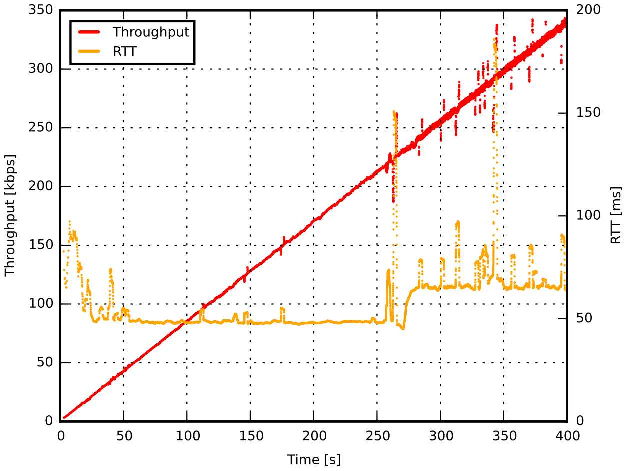

Fig. 6: Mobile network throughput measurement - variable increasing data flow.

The advantage of using increasing or decreasing traffic flow for network stress testing is shown in Fig. 6. In this test, we used slowly increasing UDP flow to probe mobile network. The aim of this test was to examine the behavior of network under changing load. You can see how RTT is changing accordingly to increasing amount of data flow. The most interesting part of this is the moment when RTT is suddenly increased. The cause of RTT increase is not the network buffering but rather an advanced network resource allocation in the transport network because generated traffic still passes network without significant losses or throughput drops. This behavior was not observed when sustained data rate was used.

Long term tests of mobile technologies show the critical dependence of transmission speed and reliability of communication on the size of the delay for both protocols (TCP and UDP). Once the delay begins to deteriorate (even slightly) it is a manifestation of strange behavior, which can be subsequently reflected by decreased throughput or total connection break-up. This behavior was observed in both networks 2G, 2.5G and also in WiMAX networks and LTE (see [9]).

5. Spurious Traffic Generation

Injecting spurious (parasitic) traffic into the network is a necessary part of network stress testing. In some cases, we need to observe behavior of network or specific data flows inside network in reaction to spurious traffic. Great advantage of FlowPing tool is possibility to allow user to use user defined traffic profile even with variable packet sizes as shown on Fig. 2.

The Fig. 7 represents one possible scenario where robustness of communication is stress tested by injecting spurious traffic into the network.

Impact of spurious traffic is twofold. At first communication path ca be congested by spurious traffic. At second communication device can be overloaded by excessive traffic. In both cases we can expect packet loss and increased response time. The duration of these conditions can be simply defined by FlowPing traffic profile.

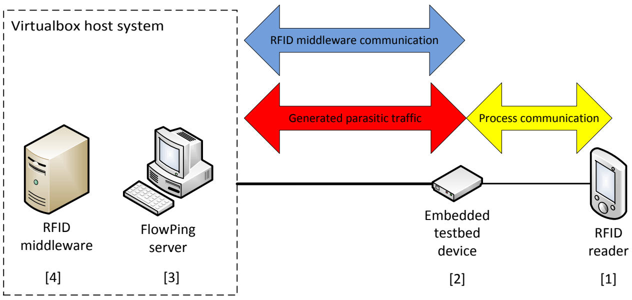

Fig. 7: IoT test bed platform.

Another possible deployment scenario is the generation of spurious traffic in order to simulate the behavior of the transmission line during real-time transmission. Typically videos [10]. Various loads of the transmission channel influence transmitted data stream and thus the resulting video quality. It is all about the loss and delay that manifest loss of images or deterioration of image quality.

6. Conclusion

The presented FlowPing tool is a complex tool for network throughput and stress testing. In combination with proper methodology, it is possible to increase the precision of results such as finding the amount of traffic when congestion in the network occurs.

Traffic generator precision and performance can be increased by busy loop waiting mode and other techniques limiting Flowping's sending code complexity such as computing packets intervals before the stress test starts.

The ability to generate variable linearly increasing or decreasing traffic flow is unique feature of FlowPing which we didn't find in any other open source traffic generator and the possibility to read complex test scenario from file make this ability very useful for throughput and stress testing. This ability allows user to observe not just static behavior of network but user can observe network behavior under slowly changing conditions.

The FlowPing tool can also simplify network parameters estimation especially when increasing or decreasing variable traffic rate profiles are used as mentioned in section 4.

Acknowledgment

This work was supported by the Grant of the Technology Agency of the Czech Republic, No. TA04011571, Radio for Smart Transmission Networks, and was researched in cooperation with RACOM and Brno University of Technology. This work was also supported by Students grant of Czech Technical University in Prague, No.SGS13/200/OHK3/3T/13.

References

[1] VONDROUS, O., Z. KOCUR, P. MACEJKO and P. JARES. FlowPing - UDP based ping application. *Flowping.comtel* [online]. 2013. `http://flowping.comtel.cz`.

[2] IETF RFC 768. *User Datagram Protocol.* Geneva: ITU-T, 1980.

[3] IETF RFC 792. *Internet Control Message Protocol.* Geneva: ITU-T, 1981.

[4] KOCUR, Z., P. MACEJKO, P. CHLUMSKY, J. VODRAZKA and O. VONDROUS. Adaptable System Increasing the Transmission Speed and Reliability in Packet Network by Optimizing Delay. *Advances in Electrical and Electronic Engineering.* 2014, vol. 12, no. 1, pp. 13–19. ISSN 1336-1376. DOI: 10.15598/aeee.v12i1.878.

[5] TOMANEK, O. and L. KENCL. CLAudit: Planetary-scale cloud latency auditing platform. In: *IEEE 2nd International Conference on Cloud Networking (CloudNet).* San Francisco: IEEE, 2013, pp. 138–146. ISBN 978-1-4799-0568-3. DOI: 10.1109/CloudNet.2013.6710568.

[6] DUGAN, J., J. ESTABROOK, J. FERBUSON, A. GALLATIN, M. GATES, K. GIBBS, S. HEMMINGER, N. JONES, F. QIN, G. RENKER, A. TIRUMALA and A. WARSHAVSKY. Iperf - measurement tool. *Sourceforge* [online]. 2013. Available at: `http://sourceforge.net/projects/iperf/`.

[7] DUGAN, J., S. ELLIOTT, B. A. MAH, J. POSKANZER and K. PRABH. Iperf3 - measurement tool. *ESnet* [online]. 2014. Available at: `http://software.es.net/iperf/`.

[8] KANUPARTHY, P. and C. DOVROLLIS. Shaperprobe: End-to-end detection of isp trafic shaping using active methods. In: *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference, IMC'11*. New York: ACM Press, 2011, pp. 473–482. ISBN 978-1-4503-1013-0. DOI: 10.1145/2068816.2068860.

[9] BECVAR, Z., P. MACH and B. SIMAK. Improvement of handover prediction in mobile WiMAX by using two thresholds. *Computer Networks*. 2011, vol. 55, iss. 1, pp. 3759–3773. ISSN 1389-1286. DOI: 10.1016/j.comnet.2011.03.020.

[10] FRNDA, J., M. VOZNAK and L. SEVCIK. Impact of packet loss and delay variation on the quality of real-time video streaming. *Telecommunication Systems*. 2015, vol. 1, iss. 1, pp. 1–11. ISSN 1018-4864. DOI: 10.1007/s11235-015-0037-2.

About Authors

Ondrej VONDROUS was born in Czech Republic in 1981. He received his M.Sc. degree in electrical engineering from the Czech Technical University in Prague in 2011. Since 2011 he has been studying Ph.D. degree in telecommunication engineering. His research interests include network transmission control, data flow analysis and data flow optimization.

Peter MACEJKO was born in Czech Republic in 1980. He received his M.Sc. degree in electrical engineering from the Czech Technical University in Prague in 2006. He is teaching networking technologies and distributed systems. His research is focused on scheduling in distributed systems and data flow and protocol analysis. He is currently actively involved in projects focused on high speed data transmission from fast moving objects.

Zbynek KOCUR was born in 1982. He received his M.Sc. degree in electrical engineering from the Czech Technical University in Prague in 2008 and Ph.D. degree in electrical engineering in 2014. He is teaching communication in data networks and networking technologies. His research is focused on wireless transmission and data flow analysis, simulation and optimization.