DTNperf_3 at work: Aims and Use.

Carlo Caini
DEI/ARCES
University of Bologna, Italy
ccaini@arces.unibo.it

Anna d'Amico
DEI/ARCES
University of Bologna, Italy
anna.damico@studio.unibo.it

Michele Rodolfi
DEI/ARCES
University of Bologna, Italy
michele.rodolfi@studio.unibo.it

ABSTRACT

Delay-/Disruption- Tolerant Networking (DTN) aims at providing interoperable communications in "challenged networks", where long delays, disruption, link intermittency and other challenges prevent, or make difficult, the use of the ordinary Internet architecture. Given their heterogeneity, to assess DTN performance in challenged networks is challenging in and of itself. For this reason, it is essential to develop suitable evaluation tools, with very flexible use. In this demo we present the third major release of DTNperf, a client-server evaluation tool designed to assess goodput and to provide status report logs in DTN Bundle Protocol (BP) architectures. In this third version DTNperf has been greatly enhanced in many respects, including full support of both DTN2 and ION (the BP reference implementation and that developed by NASA JPL, respectively). The demo wants to present both new and enhanced features in a variety of application examples, derived in part from the authors' experience on satellite and space communications. The final aim is to promote its use within the DTN community and receive suggestions from researchers. DTNperf_3 is free software and thanks to its double support is to be included not only in DTN2, as were previous versions, but also in ION.

Categories and Subject Descriptors

C.2.2 [**Computer Communication Networks**]: Network Protocols

General Terms

Measurement, Performance.

Keywords

Delay Tolerant Networking; Satellite communications; Space Communications; Performance evaluation.

1. INTRODUCTION

Delay-/Disruption- Tolerant Networking (DTN) originated from research on Inter-Planetary Networking (IPN) when its scope was enlarged to encompass all "challenged networks", [1] [2]. These include a broad spectrum of different environments, from space communications to wireless sensor networks. This heterogeneity makes the assessment of DTN performance quite unique and demands the use of specific evaluation tools.

CHANTS'13, September 30, 2013, Miami, Florida, USA.
ACM 978-1-4503-2363-5/13/09.
http://dx.doi.org/10.1145/2505494.2505508.

DTNperf was first developed to assess goodput performance in DTN bundle protocol architectures [3], [4], aiming to be a DTN equivalent of the Iperf tool, widely used to test TCP and UDP performance. From its first version (released in 2005), DTNperf has been included in the DTN2 Bundle Protocol (BP) reference implementation of DTNRG [5],[6]. DTNperf was then greatly improved with the second major release (DTNperf_2, 2008) [7], with improved robustness and many new features, including a congestion window and the possibility of logging BP status reports in a .csv (Comma Separated Values) file. DTNperf_2 has been extensively used to evaluate DTN performance in satellite communications [8] and in other fields as well. After a few years of "honored service" however the need of new major release has become evident. First, experiments with LEO satellites and in deep-space environments made clear that it would be necessary to extend the DTNperf support to ION, the BP implementation developed by NASA JPL [9]. Second, in "data-mule" experiments, or just when dealing with intermittent links, it would be very useful to collect status reports on an external monitor, instead of on the bundle source. Finally, to emulate streaming traffic, a rate-based congestion control would be necessary. All these features have now been included in the third major release, DTNperf_3, which comprises a wide variety of other important enhancements as well, as described below. DTNperf_3 has required the code to be completely rewritten; it now consists of two parts, the DTNperf application core and the Abstraction Layer (AL), designed to make DTNperf_3 compatible with both DN2 and ION.

The aim of the demo is to present DTNperf_3 to the DTN research community and to receive comments and suggestions from the audience.

2. DTNPERF_3 OVERVIEW

DTNperf_3 has three operating modes: client, server and monitor. The client generates and sends bundles, the server receives it and the monitor collects status reports in .csv files. Client, server and monitor correspond to the BP addresses "from", "to" and "reply to"[3], [4].

2.1. DTNperf Client

SYNTAX: dtnperf --client -d <dest_eid> <[-T <s> | -D <num> | -F <filename>]> [-W <size> | -R <rate>] [options]

The most important parameters are explained below. The full list of options can be obtained with the command "dtnperf --client --help".

2.1.1 Tx modes

The client has three mutually exclusive Tx modes to send bundles to the server. In the time-mode (-T), a series of bundles with a dummy payload of desired dimension (-P option) is generated and "sent", i.e. passed to the BP daemon for transmission, until the pre-set transmission time elapses. Data-mode (-D) is the same as time-mode, except that the bundle generation process ends after a given

amount of data has been "sent" to BP, and not after a time interval. File-mode (-F) differs from data-mode because a file is transferred, instead of dummy data. A noteworthy feature is the possibility to split the file into multiple bundles of desired dimension.

2.1.2 Congestion control (window- or rate- based)

Independently of the Tx mode, two alternative congestion control policies are available: window-based (-W) and rate-based (-R). In the former, the "congestion window" W represents the maximum number of bundles in-flight (i.e. sent but not acknowledged). The mechanism is similar to the TCP congestion window, but: 1) W has a fixed dimension; 2) in-flight bundles can be non-consecutive (to cope with non-ordered delivery of BP); 3) there are not retransmissions (acknowledgments are used only to trigger the transmission of new bundles). DTNperf_3 has enhanced this function with respect to previous versions, by replacing "delivered" status report [3], [4] generated by the BP of the destination node, by ACK bundles specifically created by the DTNperf_3 server, as acknowledgments of bundles sent. This innovation is essential to decouple the "reply to" from the source node, thus allowing the monitor to be launched on a node different from the source. Although the window-based mechanism is generally useful, in some cases, e.g. to emulate streaming traffic, it would be preferable to generate traffic at constant rate. For this reason, in DTNperf_3 a rate-based congestion control has been added. In response to rate-based traffic the server does not generate any ACKs, like UDP.

2.2. **DTNperf Server**

The server receives bundles, acknowledges them if sent in window-based mode, and reassembles bundle payloads if a file is transferred.

SYNTAX: dtnperf --server [options]

In this third version, one instance can serve multiple clients in parallel. Moreover, the file transfer is now robust against disordered bundle delivery (incoming payloads are buffered until either the entire file is received or a timeout expires).

2.3. **DTNperf Monitor**

The monitor receives and collects status reports and some DTNperf control bundles. It can be launched either as an independent process on an external node (external monitor), in which case it can serve many concurrent clients, or as a "child" process of a client (dedicated monitor), in which case it serves only its "parent" client. The syntax to launch an external monitor is the following:

SYNTAX: dtnperf --monitor [options]

The monitor creates a different .csv log file for each DTNperf client session (i.e. one client "launch"), containing all status reports generated by all nodes during the session.

2.4. **The Abstraction Layer**

To facilitate interoperability tests, DTNperf_3 has been designed to be independent of the BP implementation. It relies on the "Abstraction Layer", i.e. a library expressly designed to provide a common interface for DTNperf towards the APIs of different BP implementations. The present version of the AL supports DTN2 and ION, but it could be extended to other implementations. If the AL is compiled either for DTN2 or ION, DTNperf can run only on top of DTN2 or ION (Figure 1a and b). If the AL is compiled for both, DTNperf can run on top of both, as the switch between DTN2 and ION API calls is performed at run time (Figure 1c)

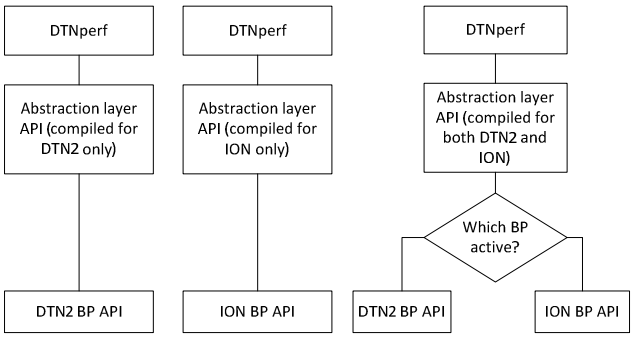

Figure 1: DTNperf compatibility. a) DTN2 only, b) ION only, c) DTN2 and ION with API call selection at run time.

3. DTNPERF_3 USE.

Although derived from authors' experience on space communications, DTNperf_3 has a general scope aiming at embracing most DTN applications. In the examples below, to be shown in the demo, the use of DTNperf_3 in some typical DTN applications is presented, assuming that we have three DTN nodes, vm1, vm2 and vm3, as source, destination and external monitor. We will focus on client syntax, which is by far the most complex.

3.1. **Basic applications**

3.1.1 Ping

To ping vm2 from vm1:

dtnperf - -client -d dtn://vm2.dtn -T 15 –W 1 - -debug=1

With this command vm1 will send bundles of 50 kB (default) to vm2 for 15s (-T15), one by one (-W1 allows just one bundle in flight). The short default lifetime (60s) is useful to force the deletion of undelivered bundles, which could interfere on subsequent experiments. As no external monitor is indicated, the .csv log file will be created on vm1 by a dedicated monitor.

3.1.2 Trace

To trace the route of a bundle:

dtnperf - -client –d dtn://vm2.dtn –D100kB -P100kB –W1 –C –f –r

This command sends a single bundle of 100 kB as the total amount of data (-D100kB) coincides with the bundle payload (-P100kB). The custody option is on (-C) and forwarded and received status reports are requested (-f, -r). From the .csv file it is straightforward to trace the route of the bundle sent.

3.1.3 File transfer.

To transfer a file segmented into multiple bundles of desired dimension:

dtnperf - -client –d dtn://vm2.dtn –F picture.jpg –P 100kB –W4

Here a file is sent (-F) instead of dummy data. The dimension of the bundle payload (-P100kB) is the dimension of segments into which the file is split. This feature is useful to match limited contact volumes as an alternative to proactive fragmentation [3]. Note however that file segmentation is carried out at application layer (by DTNperf), while bundle fragmentation is performed at BP layer (by the BP daemon).

3.2. **Performance evaluation in continous and disrupted networks**

3.2.1 Goodput (macro-analysis)

Goodput evaluation (i.e. data_ACKed/time), makes sense especially if the DTN network is not partitioned (e.g. in GEO satellite communications, where the challenges are long delay and losses). To this end, it is necessary to send dummy bundles with the window-based congestion control for a reasonable amount of time to reach and maintain a steady state. The following command could be suitable in the case of a hypothetical GEO satellite hop:

dtnperf - -client -d dtn://vm2.dtn -T 30 –P 1MB –W 4 –l 60

With this command vm1 will send bundles to vm2 for 30 s (-T30), i.e. for a much longer interval than the typical GEO RTT (600ms including processing time). To fill the (likely) large bandwidth-delay product and to reduce the impact of bundle overhead, bundles are large (-P1MB) and the congestion window W is greater than one (-W4). The same experiment should be repeated increasing W until the goodput reaches a maximum. Note that goodput evaluation should always be complemented by the analysis of status reports, collected in the example by the internal monitor (default), to control the regularity of the bundle flow and to recognize the reasons of the macro-results achieved.

3.2.2 Status report analysis (micro-analysis).

The evaluation of goodput is useful when links are continuous or only moderately disrupted (e.g. in GEO satellites with mobile terminals). As the chances of disruption increase (e.g. LEO satellites, deep space communications), the study of individual bundles (i.e. micro-analysis) becomes more important than goodput. A possible command in the presence of disruption is:

dtnperf - -client -d dtn://vm2.dtn –m dtn://vm3.dtn –D30MB –P 1 MB –W4 –l 200

This command dispatches 30 bundles of 1 MB each, with a limit of 4 bundles in flight. Status reports are collected (possibly in real time by means of dedicated links) by an external monitor (-m option). Note that the lifetime has been increased to 200s (-l200) to cope with disruption. Moreover, Data mode (-D30) is preferable because disruption makes uncertain the test duration.

3.2.3 Status report analysis of streaming traffic.

To emulate a streaming source a possible command is:

dtnperf - -client -d dtn://vm2.dtn -T30 –P100kB –R2b

This command generates a stream of 100 kB bundles for 30s, at 2 bundles per second (-R2b). No ACKs are generated by the server, as the congestion control is rate-based.

3.3. **Performance evaluation in partitioned networks: “data mule” communications**

One of the most evident advantages of DTN architecture is the possibility to cope with network partitioning. The extreme case is that of “data mule” communications, where an intermediate node (the “mule”, or “ferry”) is alternatively connected either to sender or destination. In this case the evaluation of goodput is useless, while the micro-analysis is essential. A possible command is:

dtnperf - -client -d dtn://vm2.dtn –m dtn://vm3.dtn –D 10 MB –P1 MB –W10 –l 200 -- debug=1

The external monitor (-m option) is very useful here, especially if connected to other nodes through dedicated links (e.g. in a lab testbed). By setting the window to the total number of bundles (10 in the example) these are sent in one burst, which can be preferable in this kind of experiments, in order not to have to wait for ACKs Alternatively, the user can take advantage of the rate-based congestion control, which does not imply any ACKs (e.g. by setting -R10b instead of –W10).

3.4. **Interoperability tests**

Thanks to the AL library, DTNperf_3 can run on top of either ION or DTN2 BP. Moreover, if the AL is compiled for both, the very same executable can be used. Interoperability tests, however, have to cope with the different EID schemes preferentially used by DTN2 and ION (“dtn” and “ipn” respectively). At present while ION supports also the dtn scheme, the ipn scheme in DTN2 requires a NASA patch to work properly. To facilitate experiments, DTNperf_3 in ION can register itself also with the dtn scheme, thus allowing full interoperability also with DTN2 nodes unable to use the ipn scheme. In brief, DTNperf_3 can cope with every combination of DTN2 and ION nodes, independently of the EID scheme adopted by these.

4. CONCLUSIONS

The features of DTNperf, conceived as a sort of Iperf equivalent for DTN networks, have been greatly extended in this third major release. Among the most important innovations we have the support of both DTN2 and ION, the external monitor and the rate-based congestion control. This third major release also includes an enhanced file transfer mode, an enhanced window-based congestion control, an enhanced server and many minor improvements. The demo aim is to show DTNperf_3 “at work” in a large variety of possible DTN applications and to receive feedback from the audience.

5. REFERENCES

[1] Burleigh S., Hooke A., Torgerson L., Fall K., Cerf V., Durst B. and Scott K., 2003, Delay-tolerant networking: An approach to interplanetary internet, IEEE Commun. Mag., 41, 6 (Jun. 2003), 128–136. DOI=10.1109/MCOM.2003.1204759

[2] McMahon A., Farrell S., 2009, Delay- and Disruption-Tolerant Networking, IEEE Internet Computing 13, 6 (Nov. /Dec. 2009), 82-87,. DOI=10.1109/MIC.2009.127

[3] Cerf V., Hooke A., Torgerson L., Durst R., Scott K., Fall K., Weiss H., 2007, Delay-Tolerant Networking Architecture, Internet RFC 4838,.

[4] Scott K., Burleigh S., 2007, Bundle Protocol Specification, Internet RFC 5050.

[5] Internet Research Task Force DTN Reseach Group (DTNRG) web site: http://www.dtnrg.org/

[6] DTN2 code: http://sourceforge.net/projects/dtn/.

[7] Caini C., Cornice P., Firrincieli R., Livini M., 2009, DTNperf_2: a Performance Evaluation tool for Delay/Disruption Tolerant Networking, in Proc. of ICUMT’09 (St.-Petersburg, Russia, October 2009, 1-6. DOI=10.1109/ICUMT.2009.5345423

[8] Caini C., Cruickshank H., Farrell S., Marchese M., 2011, Delay- and Disruption-Tolerant Networking (DTN): An Alternative Solution for Future Satellite Networking Applications, Proceedings of IEEE, 99, 11 (Nov.2011), 1980-1997. DOI=10.1109/JPROC.2011.2158378

[9] ION code: http://sourceforge.net/projects/ion-dtn/