Demo Abstract of Net-Controller: a Network Visualization and Management Tool

Julien Herzen
EPFL
Lausanne, Switzerland
julien.herzen@epfl.ch

Adel Aziz
EPFL
Lausanne, Switzerland
adel.aziz@epfl.ch

Patrick Thiran
EPFL
Lausanne, Switzerland
patrick.thiran@epfl.ch

***Abstract*— Net-Controller is a user-friendly network visualization and management tool developed at EPFL in order to easily retrieve and display in real time network statistics, such as link throughput and queue occupancy from a large testbed composed of wireless routers. Additionally, Net-Controller allows to control and modify the parameters of a complete network from a central point, and to easily generate traffic between different nodes. We intend to illustrate some of the features of Net-Controller through two examples that show how easily this tool detects and helps elucidate the throughput degradation that occurs in a wireless multi-hop network. The first example shows how and why fair queuing [6] improves performance compared to the standard FIFO policy used in off-the-shelf routers. The second example shows how and why a hop-by-hop congestion control mechanism, such as EZ-Flow [4] is needed to tackle the instability problem of a multi-hop scenario.**

I. Introduction

Wireless mesh networks (WMNs) based on a multi-hop backbone promise to revolutionize Internet services by providing customers with ubiquitous high-speed access at a low cost. However, even though multiple deployments of WMNs already exist, many technical challenges still need to be addressed before the wide adoption of this technology.

Indeed, current deployments use off-the-shelf hardware that run on technologies such as IEEE 802.11 and were developed for single-hop topologies. Thus, as a multi-hop environment is fundamentally different from a single-hop one, the performances (throughput, delay, etc.) achieved by these deployments appears to be significantly sub-optimal [3, 7].

Wireless links are extremely time-varying, and therefore difficult to model and simulate in a realistic way. For this reason, measurements obtained on-the-fly from real networks are often needed to assess the performance of such wireless communication systems. Although long term traces are needed to obtain statistically meaningful results, it may be very useful to easily obtain instantaneous visualization of some performance metrics when looking for the source of a problem, in the early evaluation phases, or when finely tuning a system.

Net-Controller is a program that can create nice dynamic plots representing data collected on-the-fly from the network. Besides, it also allows for some arbitrary commands to be sent to any set of nodes. This type of tool is useful for quickly analyzing a scenario both for research and educational purposes. Moreover, it is also useful for network administrators who need to be able to easily and quickly diagnose and solve a performance problem that occurs in their wireless networks. Some tools, such as Jigsaw [5], are proposed in prior work to allow the creation of a packet-level trace by merging the data obtained from multiple sniffers.

Net-Controller significantly differs from Jigsaw-like tools in three ways: (i) it is both very user-friendly with a graphical interface and easily expandable to monitor arbitrary metrics of interest in the network; (ii) second, it does not require dedicated monitoring nodes in the network and thus is also able to display internal data that cannot be sniffed on the air, such as the queue occupancy or the variation in contention window; (iii) third, it is not only a monitoring tool. Indeed, Net-Controller is designed to allow for the parameters of any node in the network to be simply modified directly from the graphical interface. Moreover, it also enables to generate traffic from any set of nodes in the network to any other node with only a few mouse clicks (see video [1]).

II. Design of the Tool

We deployed Net-Controller on our testbed composed of multiple Asus WL-500gP wireless routers that form a IEEE 802.11 mesh network and a computer that acts as the entire network monitoring and management center. Both the routers and the computer are connected to a wired network, as depicted in Figure 1, in order to ensure that the monitoring data does not interfere with the real traffic happening in the network. The routers run the OpenWRT firmware [2].

Net-Controller runs as two separated sub-programs that communicate together through a network socket.

- **The server part** runs on the computer. It consists in a simple graphical interface that allows the dynamic plots

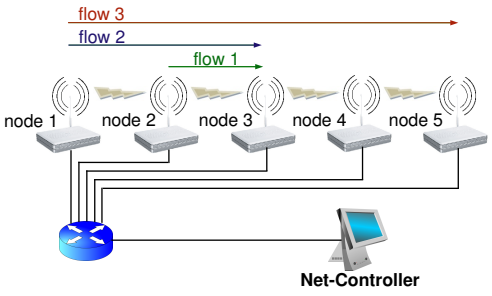

Fig. 1. Illustration of the design structure of Net-Controller on a 5-node linear topology, where 3 different flows are launched.

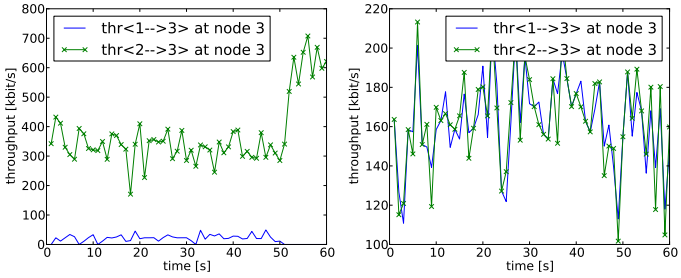

Fig. 2. Throughput received at the destination for the two flows. With the standard FIFO policy (left) and with fair queuing (right). The plots are produced by Net-Controller [1]. In the legend, "thr$\langle x \rightarrow y \rangle$" denotes the throughput of the flow from the source node x to the destination node y.

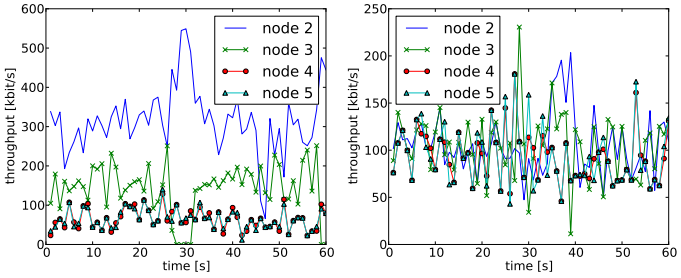

Fig. 3. Link throughput at the intermediate nodes and end-to-end throughput at the destination. With default 802.11 (left) and with EZ-flow running at the first node (right). Again, the plots are produced directly by Net-Controller [1].

to be generated, the commands to be launched on the nodes, and traffic to be started (or stopped). Requests are sent periodically to the nodes in order to retrieve the data to plot. The commands are launched through `ssh`. This program is written in Python and is multi-platform. It simply needs to know the IP addresses of the routers to interact with them.

- **The client part** runs on the wireless routers. Its role is to answer the requests sent by the server with the appropriate values. These values can be instantaneously obtained every time (e.g., a buffer occupancy), or aggregated (e.g., the number of bytes received since the last request, in order to compute a throughput). The current version of this module is implemented in C and can answer requests for: (i) the link level throughput, for each IP flow going through this router; (ii) the occupancy of the IP and MAC layer queues; and (iii) the value of the IEEE 802.11 CW_{min} parameter. We designed our program to facilitate the addition of new parameters.

III. Demo of the Tool

A. Capturing the Benefits of Fair Queuing

Currently, almost all off-the-shelf routers use a single queue with a FIFO policy (First-In, First-Out). Nevertheless, other policies have been proposed. In *fair queuing* [6], each node uses a separated queue for each individual flow and these queues are then scheduled in a round-robin manner. We implement fair queuing in Click [8] and capture its gain by having a fully backlogged UDP 1-hop flow (flow 1) and 2-hop flow (flow 2); both go through node 2 as depicted in Figure 1. Figure 2 shows that the standard FIFO policy completely starves the 2-hop flow, whereas fair queuing fairly shares the throughput between the two flows (max-min fairness). The starvation problem in FIFO occurs because the single queue of node 2 is always full of packets from flow 1, and thus drops most packets from flow 2.

B. Capturing the Benefits of EZ-Flow

The default IEEE 802.11 protocol is known to exhibit a *turbulent* (unstable) behavior when used in a multi-hop scenario [3]. These turbulences take the form of buffer build-up at the relay nodes and they result in high end-to-end delays and wasted wireless resources due to buffer-overflows. EZ-flow is a hop-by-hop congestion control mechanism that tackles this problem by dynamically adapting the MAC parameters (CW_{min}) without requiring any form of message passing [4]. Our tool successfully captures the gain of EZ-flow in a 4-hop topology (flow 3 of Figure 1). Figure 3 shows the received link-layer throughput at each relay node. We note that with EZ-flow, the link throughput at all the relay node equals the end-to-end throughput (i.e. smooth flow). In contrast, when IEEE 802.11 is used only by itself, the link throughput at the two first relays is significantly higher than the end-to-end throughput (by a factor 2-3). Therefore wireless resources are wasted and queues build up (i.e. turbulent flow).

IV. Conclusion

We have presented our network management tool, called Net-Controller, designed to facilitate the control and monitoring of a multi-hop network. Net-Controller is designed to be as user-friendly as possible by enabling - from a single graphical interface - both the control of parameters and real-time plotting of statistics from the entire network. We have demonstrated the utility of this tool through two examples by showing: (i) how easily Net-Controller captures the benefit of fair queuing and EZ-flow and (ii) how it helps elucidate the source of these gains in performance. In addition to the live demo given at Infocom, we provide two videos of our tool in use on our website [1].

References

[1] *Net-Controller: a network visualization and management tool*. http://icawww1.epfl.ch/NetController/.
[2] *OpenWRT firmware*. http://openwrt.org/.
[3] A. Aziz, D. Starobinski, and P. Thiran. Elucidating the instability of random access wireless mesh networks. In *Proc. of SECON*, Rome, Italy, June 2009.
[4] A. Aziz, D. Starobinski, P. Thiran, and A. El Fawal. Ez-flow: Removing turbulence in ieee 802.11 wireless mesh networks without message passing. In *Proc. of ACM CoNEXT*, Rome, Italy, Dec. 2009.
[5] Y.-C. Cheng, J. Bellardo, P. Benko, A. Snoeren, and G. V. nd S. Savage. Jigsaw: Solving the puzzle of enterprise 802.11 analysis. In *Proc. of ACM Sigcomm*, Pisa, Italy, Sept. 2006.
[6] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. In *Proc. of ACM Sigcomm*, Austin, TX, Sept. 1989.
[7] V. Gambiroza, B. Sadeghi, and E. Knightly. End-to-end performance and fairness in multihop wireless backhaul networks. In *Proc. of ACM MobiCom*, Philadelphia, PA, Sept. 2004.
[8] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The click modular router. *ACM Trans. Comput. Syst.*, 18(3):263–297, Aug 2000.