OpenAirInterface Traffic Generator (OTG): A Realistic Traffic Generation Tool for Emerging Application Scenarios

Aymen Hafsaoui, Navid Nikaein, Lusheng Wang
Mobile Communications Department, Eurecom, Sophia Antipolis, France
firstname.name@eurecom.fr

***Abstract*—Traffic generation represents one of the main challenge in modeling and simulating the application and network load. In this work, we present a tool, called OpenAirInterface Traffic Generator (OTG), for the generation of realistic application traffic that can be used for testing and evaluating the performance of emerging networking architectures. In addition to the traffic of conventional applications, OTG is capable of accurately emulating the traffic of new application scenarios such as online gaming and machine-type communication. To highlight the capability and new features of the tool, the one-way delay of OpenArena online gaming application in the presence of the background traffic is analyzed over the LTE network using OpenAirInterface emulation platform.**

I. Introduction

Over the last decade, the heterogeneity of the Internet is constantly increasing, with new access technologies, new client devices and with more and more services and applications. High-performance online gaming and machine-to-machine (M2M) are two examples of emerging massive applications for next-generation networks. Both applications are expected to create an increasing number of connected devices over the following years and to be an integral part of the traffic transported by the network [1]. At the present time, the most interesting applications from the commercial point of view for online gaming class are first-person shooter (e.g. OpenArena), racing (e.g. kart rider), and sports, and for M2M are intelligent transport, smart meters (automatic electricity, water and gas meters reading) and tracking and tracing. Support for such applications with a massive number of connected devices and their heterogeneous traffic have deep implications on the end-to-end network architecture [2], [3]. Consequently, understanding and modeling the traffic of such applications are a key for designing and optimizing a network and the applicable QoS scheme capable of providing adequate communication services without necessarily compromising the conventional services such as data, voice, and video. This is critical as the current networks are primarily designed and optimized for a continuous flow of information, at least in terms of the time-scales needed to send several IP packets, and mostly from the server to the client, while the traffic of emerging applications are considerably sporadic (not continuous) with low-throughput packet arrivals and mostly originated from the client to the server [4].

In the present work, we propose a *realistic* packet-level traffic generator, called OpenAirInterface traffic generator (OTG), that in addition to conventional traffics, it also takes into account the intrinsic traffic characteristics of the emerging applications such as online gaming and M2M. In particular for online gaming, it implements statistical traffic models of the Open Arena, Team Fortress, and Dirt2 applications derived from the real measurements, and for M2M it implements analytical *state-full* traffic models of Auto-Pilot, Virtual Game, and Sensor-Based Alarm or Event Detection derived from rigorous theoretical models [4], [5].

Different from the existing traffic generator [6], [7], [8], [9], OTG captures the specific characteristics of the emerging M2M and online gaming application scenarios as well as the background traffic (e.g. file download, web, email, update), and thus it is capable of generating mixed human and machine type traffic patterns. Furthermore, it has a dual operation mode: soft realtime and hard realtime, based on RTAI under Linux (i.e. LXRT module) [10], to meet application and/or protocol timing constraint.

In the next section, we present the main idea of the tool following the end-to-end one-way delay measurement and analysis. Finally, we summarize and conclude this work.

II. OpenAirInterface Traffic Generator (OTG)

OTG is a realistic packet-level traffic generation tool for emerging application scenarios. It is developed in C under Linux allowing the traffic to be generated with soft realtime and hard realtime constraint. The main difference is about the timing: soft realtime operation is designed to respect the timing on average making this mode of operation more suitable for large scale experiment, while realtime operation is designed to respect the timing strictly as would be in a real application. If OTG is attached directly to user-plane protocols, it is capable of reproducing the packet headers as in a real networking protocol stack according to the user-defined configuration. Both transmitter and receiver traffic statistics are generated and analyzed to derive the various measurements on the application-specific key performance indicators (i.e. throughput, goodput, loss-rate, latency, jitter). In OTG, the traffic generation is defined by five *ordered processes* as described below [11]:

- **State:** handling randomly distributed sojourn time at each state and probabilistic traffic state transitions (e.g. On-Off traffic models). Note that no traffic is generated within the idle/off state and during the state transition.
- **Inter-departure time (IDT):** determining the time between the transmission of two successive packets.
- **Packet size (PS):** generating the amount of payload being transferred by the packet.
- **Aggregation:** combining traffic of multiple sources into a single packet for specific nodes such as gateways.
- **Background (BG):** generating the traffic of background applications such as file download, email, and syncs/updates, modeled by PS and IDT derived from [12].

Both IDT and PS processes are modeled as i.i.d. series of variables following a user-defined distribution (e.g. constant, uniform, gaussian, exponential, poisson, weibull, pareto, gamma, cauchy and lognormal). OTG allows to reproduce exactly the same stochastic experiment by choosing the same seed values for IDT and PS random processes. Fig. 1 shows the process of packet generation in OTG.

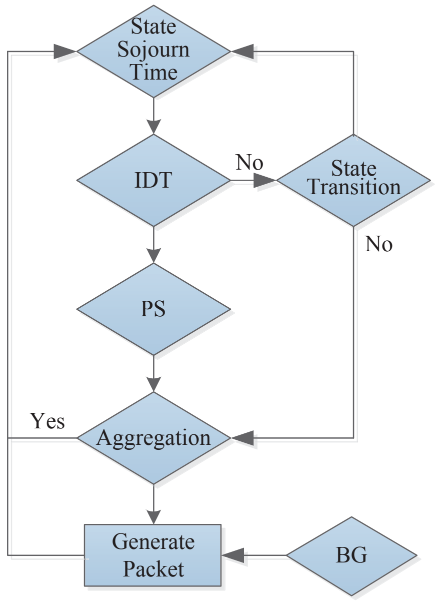

Fig. 1. Process of packet generation in OTG

In a given traffic state, the decision upon the transition to the next state is made when the sojourn time in that state expires. During the sojourn time, packet payloads are generated according to the application-specified traffic model (i.e. IDT and PS distributions). If the traffic aggregation is applicable, payloads of multiple nodes are combined together to generate a single packet. The background traffic is a parallel process that generates packets from/to server in order to emulate the cell load. Please note that OTG is reproducing the traffic of each single device, which in turn does not mean that any correlations between machines can be captured [13]. For example, assume hundreds of temperature sensors are spread over a small area, on which temperature is uniformly passing a threshold at a certain point of time. In that case all sensors would trigger simultaneously, causing strong correlation in network traffic. Such cases could be also captured by OTG if the state transition is controlled (e.g. with a predefined frequency) such that a group of devices are in the same state at a given time.

Fig. 2 shows the high level architecture we adopt in the design of OTG. It is composed of five main components as follows:

- **Configuration:** sets up the OTG parameters using user-defined xml traffic descriptor or predefined realistic configuration templates.
- **Transmitter (TX):** builds packets according to OTG packet generation process (see Fig. 1) with additional control information used for traffic statistics, and updates and log transmitter statistics.
- **Receiver (RX):** captures the packets and updates and log receiver statistics.
- **Log generation (Log Gen):** collects and formats OTG TX and RX statistics for each traffic flow.
- **Statistics and KPIs (Stats/KPIs):** analyzes and derives the measured statistical data sets for both data and background traffics, and computes key performance indicators (KPI).

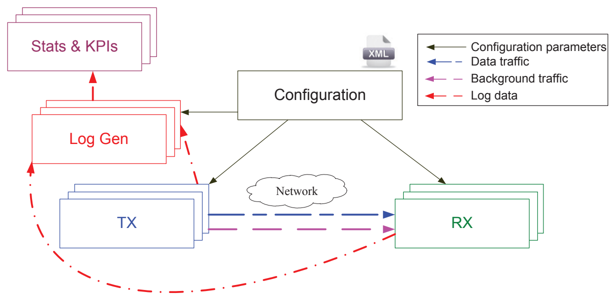

Fig. 2. High-level architecture of OTG

III. Experimentation

The experimentation is performed on the top of OpenAirInterface emulation platform [11], which is an integrated tool allowing large-scale networking experimentation applicable to both evolving cellular (i.e. LTE/LTE-A) and adhoc/mesh topologies. In the cellular configuration, the platform mainly implements the radio access network (E-UTRAN) following the 3GPPP specifications. The hardware platform is a laptop equipped with a quad-core CPU running OAI emulator and protocol stack using Linux on Ubuntu 11.10. An overview of the experimentation setup is given in Fig. 3. We carried out one-way delay (OWD) measurements in the soft realtime mode for LTE operating in TDD frame configuration 3 for 5MHz bandwidth, in a simple cellular network topology composed of one eNB (enhanced-NodeB) and one static UEs (User Equipment) to measure the best case performance. The rest of the network including mobile core network, IP backbone, and application server are emulated in terms of additional latency as the purpose of the experiment is to measure the end-to-end OWD in the data-plane. We make use of OAI scenario descriptor to layout the experiment such that the reproducibility is preserved and results can be regenerated. The simulation is run for 1 minute (i.e. 6000 LTE TDD frames). We applied the traffic pattern of OpenArena (OA) online gaming application from and to the gaming server characterized by small constant sized packets with random inter-arrival times in uplink, and variable sized packets with constant arrival time in downlink [14], [4]. The data rate is between 1 kByte/s and 5 kByte/s for most cases, and the RTT should be below 50ms to avoid any impairments in gaming experience [15]. We further applied the cell background traffic in both directions as measured and modeled in [12].

To measure the OWD, device synchronization is required. For the RAN, we used the time synchronization between eNB and UE in terms of frame and subframe number and convert it to time in milli-seconds (i.e. each frame represents 10 ms and each sub-frame represents 1ms). For mobile core network, we applied the latency measurement in [15], and for IP backbone and application server, we applied the latency estimation in [1]. The end-to-end network setup is emulated on the same physical machine, thus avoiding additional time-synchronization.

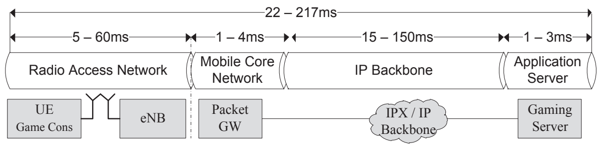

Fig. 3. Experimentation setup and OWD latency budget

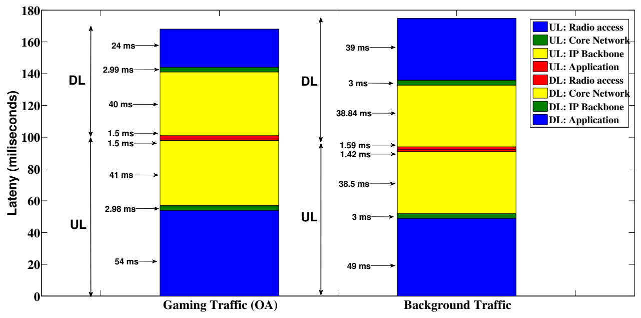

Fig. 4. OWD analysis for OpenArena online gaming application

Fig. 3 and 4 present the measured and estimated OWD performance for OpenArena per network segment. It can be seen that the delay performance is better in downlink (from server) than in uplink (to server). The reason is that the LTE TDD frame configuration 3 has more downlink subframes (i.e. 6) than that of uplink (i.e. 3) indicating that the DL and UL subframe allocation should be balanced to meet the load conditions. Similar results have been reported in an LTE FDD frame format with unknown number of users [15]. Furthermore, we can observe that the delay of the radio access network is much larger than that of mobile core network, which calls for further optimization of cell configuration, uplink channel access method and scheduling especially when the number of users increases. When comparing with the IP backbone (Internet), we note that this segment has a large delay in both direction, which depends on the region, the number of nodes in the network, and their processing delays [1]. The IP backbone represents a high delay variation, which can be improved by providing service locally closer to the client (e.g. within the mobile packet gateway). From the results, we can conclude that 4G network is not yet ready to host certain applications.

IV. Conclusion

Accurate modeling and generation of realistic application traffic are difficult and challenging tasks in view of emerging application scenarios. In this work, we briefly discussed three new traffic features - state-full, aggregated, and background - that OTG implements to generate realistic application traffic. The tool is used to measure and analyze the one-way-delay of online gaming in presence of background traffic over LTE network. We plan to extend the scope of our analysis to the M2M traffic with large number of user equipment to validate state-full traffic generation and add supports for application-specific key performance indicators.

Acknowledgment

The research leading to these results has received funding from the European Research Council under the European Community Seventh Framework Programme (FP7/2007-2013) n^o 248993 (LOLA) and n^o 257616 (CONECT).

References

[1] N. Nikaein and S. Krco. Latency for real-time machine-to-machine communication in LTE-based system architecture. In *EW'11, 17th European Wireless Conference, Sustainable Wireless Technologies, April 27-29, 2011, Vienna, Austria*, 2011.
[2] M. Claypool and K. Claypool. Latency and player actions in online games. 2006.
[3] H. Lenz and J. Koss. M2m communication - next revolution on wireless interaction, 2008.
[4] LoLa Consortium. Lola project (achieving low-latency in wireless communications), "d3.5 traffic models for m2m and online gaming network traffic,". 2011.
[5] LoLa Consortium. Lola project (achieving low-latency in wireless communications), "d2.1 target application scenarios," available on: www.ict-lola.eu. 2010.
[6] A. Botta, A. Dainotti, and A. Pescap. A tool for the generation of realistic network workload for emerging networking scenarios. 2012.
[7] Vishwanath, K. Venkatesh, and A. Vahdat. Realistic and responsive network traffic generation. In *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '06. ACM, 2006.
[8] J. Sommers and P. Barford. Self-configuring network traffic generation. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, IMC '04. ACM, 2004.
[9] www.netperf.org/.
[10] www.rtai.org/.
[11] www.openairinterface.org.
[12] M. Laner, P. Svoboda, S. Schwarz, and M. Rupp. Users in cells: a data traffic analysis. WCNC, 2012.
[13] M. Laner, P. Svoboda, and M. Rupp. Modeling randomness in network traffic. In *Proceedings of the Sigmetrics/Performance*, 2012.
[14] www.openarena.ws.
[15] M. Laner, P. Svoboda, P. Romirer-Maierhofer, N. Nikaein, F. Ricciato, and M. Rupp. A comparison between one-way delays in operating HSPA and LTE networks. In *8th International Workshop on Wireless Network Measurements (WinMee)*, 2012.