Calibrating MoonGen for Reproducible Experiments

MoonGen is a modern networking tool developed for high-speed packet generation. It provides a Lua wrapper for DPDK (Data Plane Development Kit) to enable flexible and high-level packet generation. MoonGen is designed to support multiple receive and transmit queues for multi-core CPUs, which are essential for low-level packet processing tasks. It is particularly useful for developing packet generators that can be easily integrated with C libraries like DPDK.

MoonGen's core is a Lua wrapper for DPDK that provides utility functions required by a packet generator. It supports the underlying hardware features like timestamping and rate control in hardware. This allows us to fulfill requirements (R1) without violating (R4). For instance, it can effectively conceal short pause times and store large packets, even of data at 10 GbE. This is important for the design of a high-speed packet generator.

The typical operating system socket API hides important aspects of networking hardware that are important for the design of low-level packet processing tools. The MoonGen API comes with functions that configure filters (e.g. Intel Flow Director) or hashing to threads and the incoming traffic. This restricts the underlying hardware, which can be included in an operating system: netmap is already available. This restriction in netmap is critical as it is designed to support purchasing a license. In netmap user space applications do not rely on dynamic allocation in the transmit logic to avoid pause times. Typical scripts in MoonGen do not need to explicitly start the garbage collector if necessary, and run for a predetermined time. Nevertheless, scripts can be included in an operating system: netmap is already available.

MoonGen must be as flexible as possible (R3). Therefore, MoonGen moves the whole packet generation logic into user-defined scripts as this ensures the maximum possible flexibility. LuaJIT was selected because related experiments in MoonGen show that it is suitable for high-speed packet processing tasks (R2). We disable garbage collection by default in MoonGen to avoid pause times. Typical scripts in MoonGen do not need to access NIC registers directly to implement reusable functionality (R1) without violating (R4).

Programmierkenntnisse in Lua
Interesse an hardwarenaher Programmierung


Paul Emmerich emmericp@net.in.tum.de
http://go.tum.de/742098