Research and Development Proposal (v0)

Anderson dos Santos Paschoalon

M.Sc. Student

Dr. Christian Esteve Rothenberg

Mentor

Agenda

- ➔ Requirements and Goals
- ➔ Main components (v0)
- ➔ Component implementation
- ➔ Validation
- ➔ Next Workplan Preview

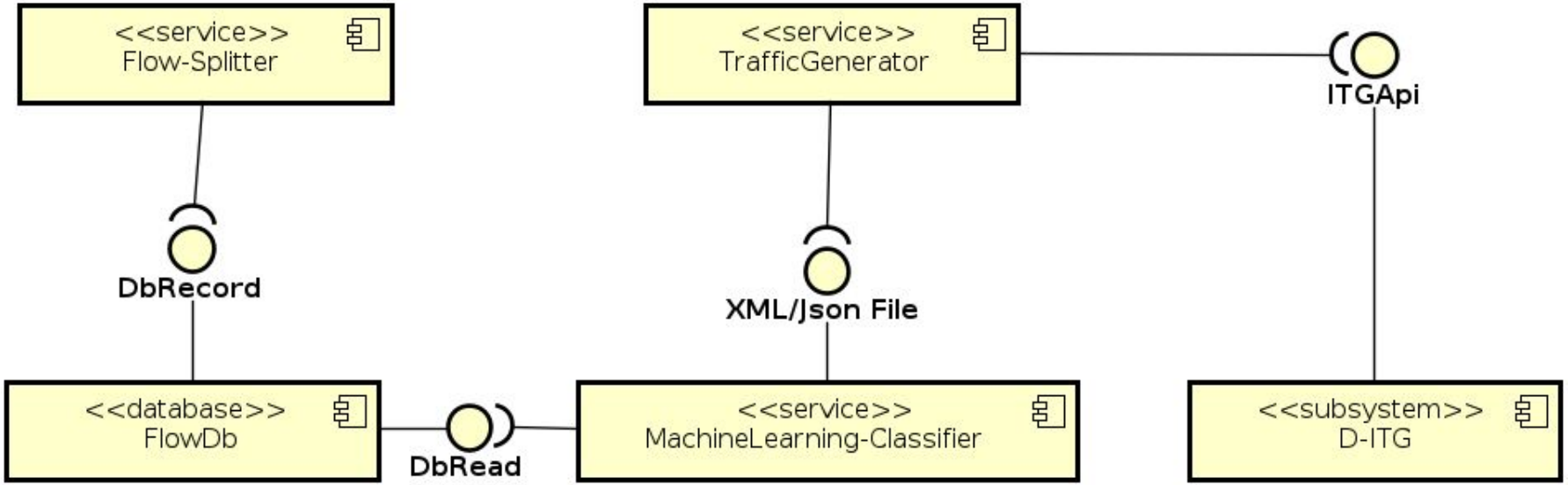
Requirements and Goals

- What?
  - Create a software capable of “learn” traffic patters from real traces, and capable of mimic this behavior
- How?
  - Create “**compact trace descriptors**” that store flow information in any markup language (XML/Json). Then, use a trafic genenerator API capable of “execute” these flows.

Requirements and Goals

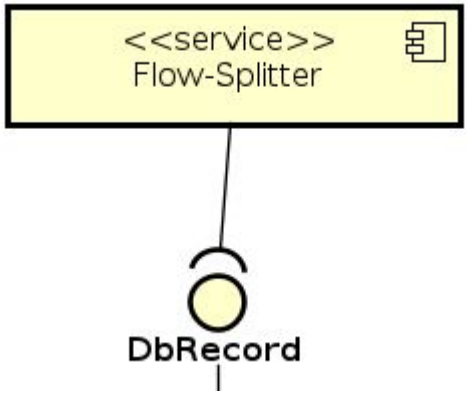
- ➔ How?
  - ◆ To implement these trace descriptors, it is proposed the use of machine learning classifiers, capable of "**guess**" what sort of flows are being transmitted. Based on a previous learning process. Simple measurable flow features, like throughput and start/end time, could be directly measured.
  - ◆ And, to classify these flows, a flow-spliter is need.

Main components (v0)



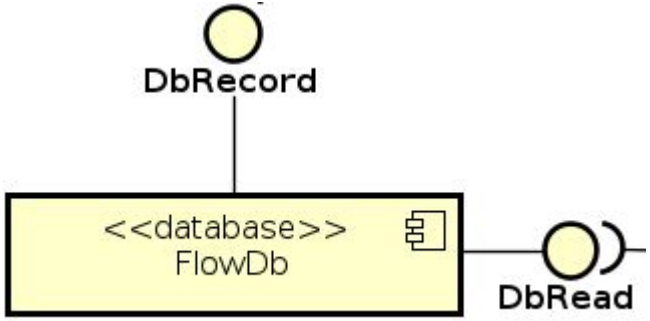
Main components (v0)

- Flow-spliter
  - Responsible for split the colected traffic trace into flows, and store the useful colected data into a database.



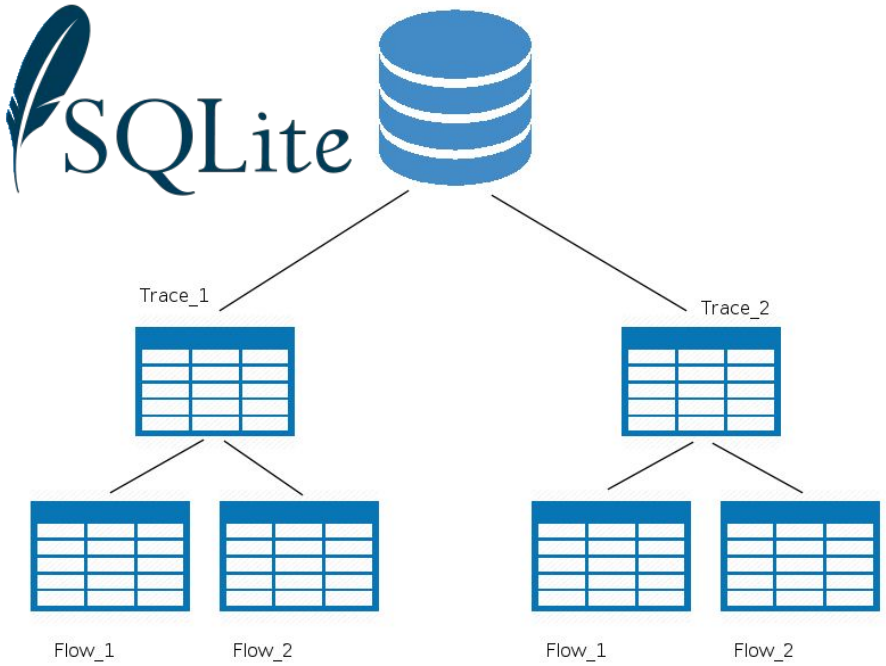
Main components (v0)

- FlowDb
  - A database that store colected data into flows, for analysis.
  - The information should be written by the Flow-Spliitter component, and read by the Machine Learning component.



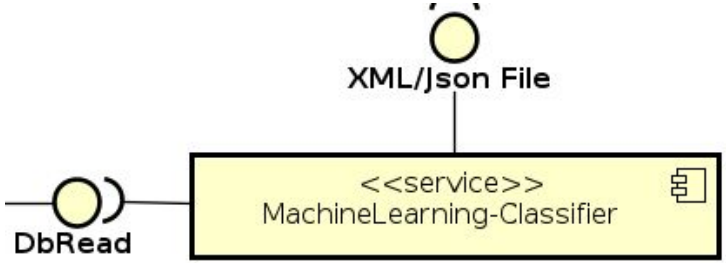
Main components (v0)

- The database must store a table for each trace captured (one for each experiment), and each one should store a table for each flow from this trace.
- According to SQLite specifications, it should be the best option for a datase: simple and suitable for the amount of data (less then terabytes).



Main components (v0)

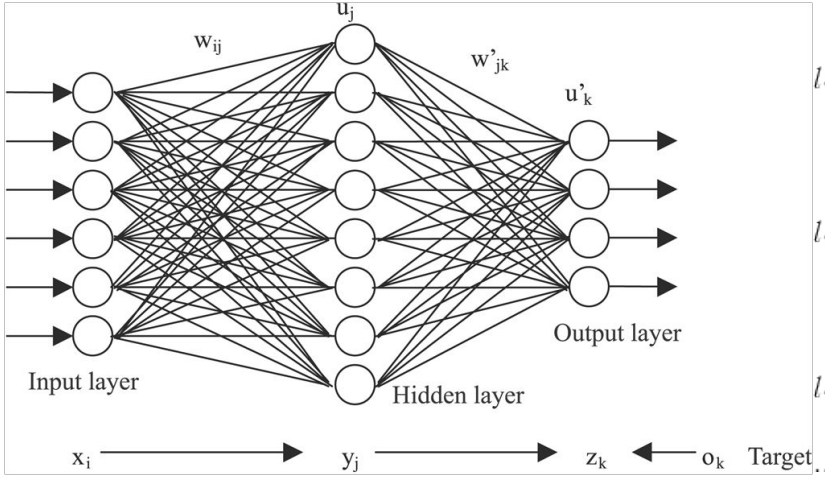
- The Machine Learning Classifier should be able learn features form the measured traces, and record them into a machine-readable file (XML/Json).
  - Flow-level features, Packet-level QoS metrics and Leyer-4 features could be directely measured:
    - Flow-level properties: diration, start delay, total number of packets, total number of KBytes.
    - Packet-level QoS metrics: bitrate, packet, RTT, Jitter, packet loss.
    - Leyer-4 features: protocols TCP, UDP, ICMP, DCCP and SCTP.



Main components (v0)

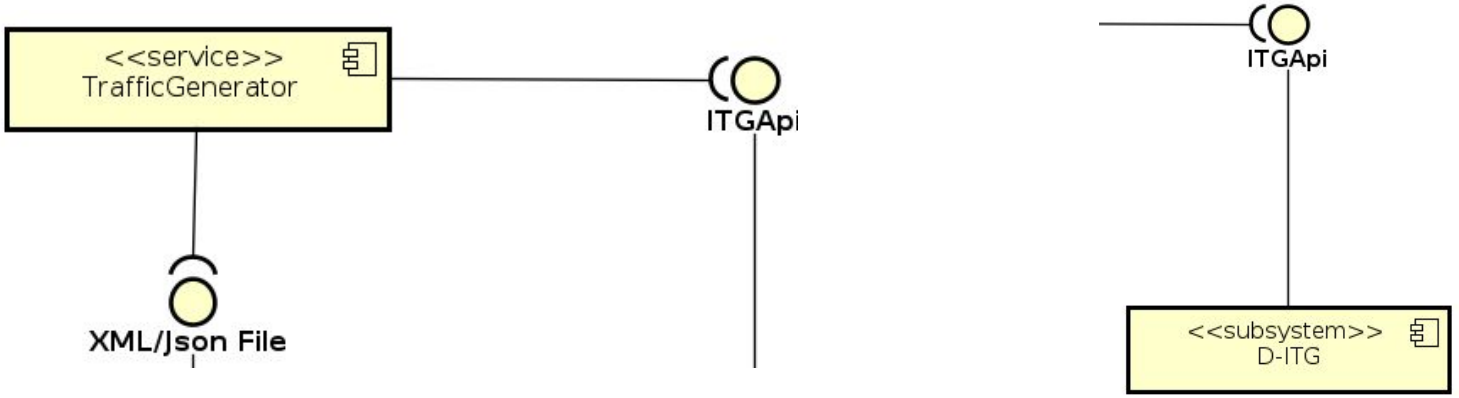
- Stochastic modesl for PS(packet size) and IDT (Inter Departure Time) should use some Machine Learning technic to define what sort of profile the flow fits.

- For exemple, a neural network is capable of classify somo profile of data into labels. These labels could be the **workload profile (PS and IDT)**.



Main components (v0)

- ➔ The traffic generator should be able to read the file created by the Machine Learning classifier (here called compact trace descriptor), and produce flows using these features.
- ➔ To implement this task is going to be used the D-ITG API.



Component implementation

➔ Now, some components implementations may be defined.

Flow-Spliter	not defined
FlowDb	SQLite
Machine Learning Classifier	not defined (Python or C++ API for machine learning)
Traffic Generator	C++ (First use D-ITG API, later it could be expanded)
CLI*	Klish(http://code.google.com/p/klish/) (C++ and XML)

(*) Item not mentioned before

Validation

- ➔ First, the machine learning component should have a reasonable acuracy (higher as possible).
- ➔ Compare syntethic trace shapes with real ones.
- ➔ Feedback a generated trace shape, and compare the compac traces descriptors (before and after feedback).
- ➔ **First validation**: capture a single flow produced by D-ITG, and implement single stochastic feature classification.

Next Workplan Preview

Activities	Week1	Week2	Week3	Week4	Week5	Week6
(1) Bibliography review and report (latex): art state in traffic classification, and stochastic models for workloads.	■	■	■	■		
(2) Flow Splitter implementation (v0) (for pcap files)					■	■
(3) Requiriments extraction for the Traffic Generator, and interface definition (Compact trace descriptor template).	■	■				
(4) Traffic Generator implementation (v1)			■	■	■	■
(5) Database Requeriment extraction and definition.				■	■	
(6) Database implementation (v0)					■	■
(7) Coursera: Practical Machine Learning	■	■	■	■	■	■
(8) Coursera: Machine Learning	■	■	■	■	■	■

Next Workplan Preview

- Observations
  - The conclusion (2) and (6) (v1 of the component) should stay for the next Working Plan
  - The definition of the Machine learning model and metodology depends of (1) and (3).
  - (5) depends on (1)
  - (4) enables network tests.
  - If everything works as planed, the first validation will be possible at the end of the next working plan.
  - More detail about tools will be defined during the nexto working plan: these are just the guidelines for the next cicle, and a starting point for the research as a whole.
  - New itens could be added, until the “release” of the next WP.