Extending the switch

Extending the switch

The following is a brief description of how the switch can be extended with additional functionality, by using the experimenter features of OpenFlow.

In the current implementation OpenFlow control channel processing is done in two distinct steps: first the wire format is converted to an internal representation (with syntax checking), and then this internal representation is interpreted by the soft switch code. The other direction is handled in a similar manner: the switch code generates a message in an internal representation, which is then converted into OpenFlow wire format

oflib

The OpenFlow library (oflib) defines the internal representations for each OpenFlow entity (messages, actions, etc.). It also provides functions for conversion between the OpenFlow wire format and those internal structures. Functions for conversions are also available for all entities, but most important of these are *ofl_msg_unpack* and *ofl_msg_pack*, which converts complete messages. Oflib provides this functionality for all OpenFlow 1.1 messages, and in both directions; so in fact it can be (re)used for both switch and controller implementations.

In order to process experimenter entities, each *oflib* function takes an extra argument (wherever appropriate), where the user can provide a number of callback functions. A group of functions can be defined for each experimenter entity class, all of which are collected into a single experimenter callback set for convenience. For details on the experimenters, one can refer to the *ofl.h* header file.

The task of the experimenter implementer is to define their own internal strucutres - reusing the already defined internal headers (e.g. ofl_*action_experimenter*), and provide the appropriate callback functions. Oflib itself will decide based on the wire format or internal headers, whether the encountered entity is standard or experimenter, and will so use its internal functions, or the provided callbacks.

The current implementation provides examples of such experimenter callbacks, found in the *oflib-exp* module. While experimenters could be part of the switch implementation itself, *oflib-exp* is separated as it is reused by both the switch and the datapath controller (*dpctl*) utility. The experimenter might simply replace these callbacks with their own, or can create a wrapper, which decides whether to use their own functions, or the *oflib-exp* ones -- similar to how *oflib-exp* itself dispatches to *ofl-exp-openflow* and *ofl-exp-nicira*.

udatapath

In the soft switch implementation control messages are processed in the *dp_control.c* file. At this point the received message is already in oflib internal format. The code here is responsible for dispatching the incoming control messages to the appropriate modules based on the message types. Messages of experimenter types are dispatched to the *dp_exp.c* code, where the stubs should be extended for new message types.

When sending responses, or messages from the switch to the controller, the implementer can construct the message in its internal format, with whatever experimenter structures involved. The sending functionality will use *oflib* to convert the message to the OpenFlow wire format, which in turn will use the provided experimenter callbacks for the conversion.