OpenFlow Intelligence API - Legacy (OFIA-TCP)

Andreas Schmidt, Saarland University

June 3, 2014

Object and Type Notation

The following appendix chapters define the schemas in a way that an object type has a name and consist of different properties. The object type's name is given in the table heading and if there is a further type given after a colon, the latter is the base type of which all properties are inherited. Each of the properties is a table row and has a key and a type, where the type is either an atomic JavaScript type (Number, String, Boolean), a composition of those (e.g. arrays) or an object type which is also specified in the appropriate section. Furthermore, there are a few types that are for example normal strings, but require a specific format:

- *DeviceID*: This represents either the *OpenFlow* node's datapath or the MAC address of a client. Therefore the format is `01:23:45:67:89:ab`, so six pairs of hexadecimal digits.
- *IP Address*: This is the usual format for IPv4 addresses, e.g. `192.168.1.10`. Optional CIDR masks can be given in form of a number for specifying subnets, e.g. `192.168.1.0/24`
- *FlowID* is a string that is based on hashing the flow's parameters, which results in equal flows having the same id.
- *Color*: A standard HTML RGB color in string representation using the following format `#000000`;
- *Email Address*: All valid email addresses can be stored in this string variable.
- *FixedNumber*: This type is used to transfer fixed precision numbers. As Numbers in JavaScript use as many decimals as available, this type uses a string representation of a floating point number.

The presented API has the version number **v0.9.1**. As the TCP interface is considered as a obsolete solution, the development of this API is discontinued. For more information on the latest API have a look at the *OFIA-HTTP* document.

1 Controller Interface

In order to provide an interface for querying information, the controller has to open a TCP socket on an arbitrary port (this can be configured via the web interface).

The query sent by the client contains serialized JSON, which has the following format:

```
{ type:  "ofcontrol", command:  "v2/" + command }
```

The commands can be: `getInfos`, `getLinks` or `getFlows`. The respective return formats are given in 2.2.

2 Data Schema

This schema covers the data sources that the controller exposes. It describes which network information is given by which source and what fields the different objects have.

2.1 Basic Types

There are a few basic types:

ControllerStatus		
Key	**Type**	**Description**
`monitoredNetworks`	*IP Address*[]	List of networks that are monitored by this controller
`controllerType`	String	Type identification (e.g. NOX)

Client		
Key	**Type**	**Description**
`id`	*DeviceID*	Identification (MAC address)
`ip`	*IP Address*	Configured IP address
`gw`	*DeviceID*	Gateway (connected OpenFlow node)
`port`	Number	Physical gateway port number

Flow		
Key	**Type**	**Description**
`id`	*FlowId*	Identification
`dlSrc`	Number	Source MAC address
`dlDst`	Number	Destination MAC address
`dlType`	Number	Link layer type identifier (e.g. Ethernet II)
`dlVlan`	Number	Vlan tag
`dlVlanPcp`	Number	Vlan priority
`nwSrc`	Number	Source IP address
`nwDst`	Number	Destination IP address
`nwToS`	Number	Type of service
`nwProtocol`	Number	Network layer protocol
`tpSrc`	Number	Source port number
`tpDst`	Number	Destination port number

Link		
Key	**Type**	**Description**
`src`	*DeviceID*	Source ID
`srcPort`	Number	Physical source port number
`dst`	*DeviceID*	Destination ID
`dstPort`	Number	Physical destination port number
`delay`	*FixedNumber*	Delay
`drRx`	*FixedNumber*	Receive data rate
`drTx`	*FixedNumber*	Transmit data rate
`plr`	*FixedNumber*	Packet loss rate

2.2 Data Sources

These basic types together make up the following data sources:

2.2.1 General Information (`getInfos`)

Infos		
Property Name	**Type**	**Description**
`status`	ControllerStatus	
`switches`	*DeviceID*[]	Switches' datapaths
`clients`	Client[]	Clients of the network

2.2.2 Link Information (`getLinks`)

The source **Links** is just a list of *Link* elements.

2.2.3 Flow Information (`getFlows`)

Flows		
Property Name	**Type**	**Description**
`flows`	Flow[]	List of flows in the network
`flowsByDevice`	*FlowID*[][]	List of flow IDs grouped by *DeviceIDs*