OPEN NETWORKING FOUNDATION

Role Status Extension

Version 0.1

December 23, 2014

Disclaimer

Contents

List of Tables

List of Figures

1 Introduction

This document describes an ONF extension for OpenFlow version 1.3.X that report controller role changes through a new status message.

2 How it works

A controller can request its role to be changed to OFPCR_ROLE_MASTER. If the switch must change the role of another controller from OFPCR_ROLE_MASTER to OFPCR_ROLE_SLAVE, the switch must generate a controller role status event for this controller informing it of its new state (in many cases that controller is no longer reachable, and switch may not be able to transmit that event).

3 Controller role status Experimenter ID

The Experimenter ID of this extension is:

ONF_EXPERIMENTER_ID = 0x4F4E4600

4 Controller role status Message

The following message types are defined by this extension.

```
/* Message types */
enum onf_exp_type {
  ONF_ET_ROLE_STATUS = 1911,       /* Role status - Async message */
};
```

When a controller has its role changed by the switch, and not directly changed by that controller using a OFPT_ROLE_REQUEST message, the corresponding controller must be informed with a ONF_ET_ROLE_STATUS message. The switch may generate ONF_ET_ROLE_STATUS messages at other times, for example when experimenter data change, this is outside the scope of this specification.

The ONF_ET_ROLE_STATUS message uses the following structure and fields:

```
/* Role status message. */
struct onf_message_role_status {
  struct ofp_header     header;
  uint32_t              experimenter;      /* ONF_EXPERIMENTER_ID. */
  uint32_t              exp_type;          /* ONF_ST_ROLE_STATUS. */
  uint32_t              role;              /* One of OFPCR_ROLE_*. */
  uint8_t               reason;            /* One of ONFCRR_*. */
  uint8_t               pad[3];            /* Align to 64 bits. */
  uint64_t              generation_id;     /* Master Election Generation Id */

  /* Role Property list */
```



```
  struct onf_role_prop_header properties[0];
};
OFP_ASSERT(sizeof(struct onf_message_role_status) == sizeof(struct ofp_experimenter_header) + 16);
```

The `type` field must be set to `OFPET_EXPERIMENTER`.

The `experimenter` field is the Experimenter ID (see 3).

The `reason` field can be one of the following values:

```
/* What changed about the controller role */
enum onf_controller_role_reason {
    ONFCRR_MASTER_REQUEST = 0,  /* Another controller asked to be master. */
    ONFCRR_CONFIG         = 1,  /* Configuration changed on the switch. */
    ONFCRR_EXPERIMENTER   = 2,  /* Experimenter data changed. */
};
```

The `role` field is the new role of the controller. The `generation_id` is the generation ID that was included in the role request message that triggered the role change.

The `properties` field is a list of role properties, describing dynamic parameters of table configuration.

The list of role property types that are currently defined are:

```
/* Role property types.
 */
enum onf_role_prop_type {
    ONFRPT_EXPERIMENTER           = 0xFFFF, /* Experimenter property. */
};
```

A property definition contains the property type, length, and any associated data:

```
/* Common header for all Role Properties */
struct onf_role_prop_header {
    uint16_t         type;    /* One of ONFRPT_*. */
    uint16_t         length;  /* Length in bytes of this property. */
};
OFP_ASSERT(sizeof(struct onf_role_prop_header) == 4);
```

The `ONFTMPT_EXPERIMENTER` property uses the following structure and fields:

```
/* Experimenter role property */
struct onf_role_prop_experimenter {
    uint16_t         type;    /* One of ONFRPT_EXPERIMENTER. */
    uint16_t         length;  /* Length in bytes of this property. */
    uint32_t         experimenter;  /* Experimenter ID which takes the same
                                       form as in struct
                                       ofp_experimenter_header. */
    uint32_t         exp_type;      /* Experimenter defined. */
    /* Followed by:
     *   - Exactly (length - 12) bytes containing the experimenter data, then
     *   - Exactly (length + 7)/8*8 - (length) (between 0 and 7)
```



```
     *     bytes of all-zero bytes */
    uint32_t         experimenter_data[0];
};
OFP_ASSERT(sizeof(struct onf_role_prop_experimenter) == 12);
```

The `experimenter` field is the Experimenter ID, which takes the same form as in struct `ofp_experimenter`.