



OPEN NETWORKING
FOUNDATION

Copy-Field action Extension

Version 0.1

December 23, 2014



Disclaimer

THIS SPECIFICATION IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. Without limitation, ONF disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification and to the implementation of this specification, and ONF disclaims all liability for cost of procurement of substitute goods or services, lost profits, loss of use, loss of data or any incidental, consequential, direct, indirect, or special damages, whether under contract, tort, warranty or otherwise, arising in any way out of use or reliance upon this specification or any information herein.

No license, express or implied, by estoppel or otherwise, to any Open Networking Foundation or Open Networking Foundation member intellectual property rights is granted herein.

Except that a license is hereby granted by ONF to copy and reproduce this specification for internal use only.

Contact the Open Networking Foundation at <http://www.opennetworking.org> for information on specification licensing through membership agreements.

Any marks and brands contained herein are the property of their respective owners.

WITHOUT LIMITING THE DISCLAIMER ABOVE, THIS SPECIFICATION OF THE OPEN NETWORKING FOUNDATION ("ONF") IS SUBJECT TO THE ROYALTY FREE, REASONABLE AND NONDISCRIMINATORY ("RANDZ") LICENSING COMMITMENTS OF THE MEMBERS OF ONF PURSUANT TO THE ONF INTELLECTUAL PROPERTY RIGHTS POLICY. ONF DOES NOT WARRANT THAT ALL NECESSARY CLAIMS OF PATENT WHICH MAY BE IMPLICATED BY THE IMPLEMENTATION OF THIS SPECIFICATION ARE OWNED OR LICENSABLE BY ONF'S MEMBERS AND THEREFORE SUBJECT TO THE RANDZ COMMITMENT OF THE MEMBERS.

Contents

1	Introduction	3
2	How it works	3
3	Copy-Field Experimenter ID	3
4	Copy-Field action	3
5	Copy-Field table features properties	5

1 Introduction

This document describes an ONF extension for OpenFlow version 1.3.X that adds an action to copy OXM fields.

2 How it works

An new OpenFlow action is defined. The *Copy-Field* action may copy data between any header or pipeline fields. It is typically used to copy data from a header field to a packet register pipeline field or from a packet register pipeline field to a header field, and in some cases from a header field to another header field. A switch may not support all combinations of copies between header or pipeline fields.

The effect of *copy-field* actions in the action set is undefined due to race conditions, and therefore its implementation in the action set discouraged.

Two new OpenFlow table features properties are defined to expose which fields are supported as a destination of the *copy-field* action.

3 Copy-Field Experimenter ID

The Experimenter ID of this extension is:

```
ONF_EXPERIMENTER_ID = 0x4F4E4600
```

4 Copy-Field action

This extension defines the following experimenter action type:

```
/* Action types */
enum onf_act_exp_type {
    ONFACT_ET_COPY_FIELD = 3200,          /* Copy-Field action. */
};
```

The action ONFACT_ET_COPY_FIELD uses the following structure:

```
/* Action structure for ONFACT_ET_COPY_FIELD. */
struct onf_act_copy_field {
    uint16_t    type;           /* OFPAT_EXPERIMENTER. */
    uint16_t    len;           /* Length is padded to 64 bits. */
    uint32_t    experimenter;   /* ONF_EXPERIMENTER_ID. */
    uint16_t    exp_type;       /* ONFACT_ET_COPY_FIELD. */
    uint8_t     pad2[2];
    uint16_t    n_bits;         /* Number of bits to copy. */
    uint16_t    src_offset;      /* Starting bit offset in source. */
    uint16_t    dst_offset;      /* Starting bit offset in destination. */
    uint8_t     pad[2];         /* Align to 32 bits. */
    /* Followed by:
     * - Exactly 8, 12 or 16 bytes containing the oxm_ids, then
     * - Enough all-zero bytes (either 0 or 4) to make the action a whole
     *   multiple of 8 bytes in length */
    uint32_t    oxm_ids[0];     /* Source and destination OXM headers */
};
OFP_ASSERT(sizeof(struct ofp_action_copy_field) == 20);
```

The Copy-Field action copies `src_oxm_id[src_offset:src_offset+n_bits]` to `dst_oxm_id[dst_offset:dst_offset+n_bits]`, where `a[b:c]` denotes the bits within 'a' numbered 'b' through 'c' (not including bit 'c'). Bit numbering starts at 0 for the least-significant bit, 1 for the next most significant bit, and so on.

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

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

The `exp_type` field is set to `ONFACT_ET_COPY_FIELD`.

The `n_bits` field contains the number of bits to copy from `src_oxmid` into `dst_oxmid`.

The `src_offset` field indicates the bit offset in `src_oxmid` where bits should be read.

The `dst_offset` field indicates the bit offset in `dst_oxmid` where bits should be written.

The `oxm_ids` field is an list containing two OXM types. The first element of this list, `src_oxm_id`, identified the header or pipeline field where the value is copied from. The second element of this list, `dst_oxm_id`, identified the header or pipeline field where the value is copied to. The elements of that list are 32-bit OXM headers for non-experimenter OXM fields or 64-bit OXM headers for experimenter OXM fields, those OXM fields don't include any payload. The value of `oxm_hasmask` must be zero and no value or mask is included. If the *Copy Field* action is supported, it must support `src_oxm_id` or `dst_oxm_id` being one of the packet registers supported by the switch. Support for other pipeline fields and header fields is optional, for those fields this action must obey the same rules as the *Set Field* action, in particular with respect to prerequisite.

The switch must reject *Copy Field* actions for which `src_offset+n_bits` is greater than the width of `src_oxm_id` or `dst_offset+n_bits` is greater than the width of `dst_oxm_id` with error type `OFPET_BAD_ACTION`, code `OFPBAC_BAD_SET_ARGUMENT`. This action must behaves properly when `src_oxm_id` overlaps with `dst_oxm_id`, that is, it behaves as if `src_oxm_id` were copied out to a temporary buffer, then the temporary buffer copied to `dst_oxm_id`, if this is not possible the switch must reject the *Copy Field* action with error type `OFPET_BAD_ACTION`, code `OFPBAC_BAD_SET_TYPE`.

The effect of *copy-field* actions in the action set is undefined due to race conditions, and therefore its implementation in the action set is discouraged. The action ordering for including this action in the action set is undefined.

5 Copy-Field table features properties

This extension defines the following experimenter table feature properties types:

```
/* Action types */
enum onf_tfp_exp_type {
    ONFTFP_ET_WRITE_COPYFIELD = 3200,      /* Write Copy-Field property. */
    ONFTFP_ET_APPLY_COPYFIELD = 3201,      /* Apply Copy-Field property. */
};
```

The table feature properties ONFTFP_ET_WRITE_COPYFIELD and ONFTFP_ET_APPLY_COPYFIELD use the following structure:

```
/* Table Feature Property structure for ONFTFP_ET_WRITE_COPYFIELD
 * and ONFTFP_ET_APPLY_COPYFIELD. */
struct onf_tfp_copy_field {
    uint16_t      type;      /* One of OFPTFPT_EXPERIMENTER,
                             OFPTFPT_EXPERIMENTER_MISS. */

    uint16_t      length;    /* Length in bytes of this property. */
    uint32_t      experimenter; /* ONF_EXPERIMENTER_ID. */
    uint32_t      exp_type;   /* ONFTFP_ET_WRITE_COPYFIELD or
                             ONFTFP_ET_APPLY_COPYFIELD. */

    /* Followed by:
     * - Exactly (length - 4) bytes containing the oxm_ids, then
     * - Exactly (length + 7)/8*8 - (length) (between 0 and 7)
     *   bytes of all-zero bytes */
    uint32_t      oxm_ids[0]; /* Array of OXM headers */
};
OFP_ASSERT(sizeof(struct onf_tfp_copy_field) == 12);
```

The `type` field must be set to `OFPTFPT_EXPERIMENTER` or `OFPTFPT_EXPERIMENTER_MISS`. The properties using `OFPTFPT_EXPERIMENTER_MISS` describe the capabilities for the table-miss flow entry whereas the properties using `OFPTFPT_EXPERIMENTER` describe the capabilities for regular flow entry.

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

The `exp_type` field is set to `ONFTFP_ET_WRITE_COPYFIELD` or `ONFTFP_ET_APPLY_COPYFIELD`. The `ONFTFP_ET_WRITE_COPYFIELD` properties describe *Copy-Field* action types supported by the table using the `OFPT_WRITE_ACTIONS` instruction, whereas the `ONFTFP_ET_APPLY_COPYFIELD` properties describe *Copy-Field* action types supported by the table using the `OFPT_APPLY_ACTIONS` instruction.

The `oxm_ids` field is the list of OXM types for the feature. The elements of that list are 32-bit OXM headers for non-experimenter OXM fields or 64-bit OXM headers for experimenter OXM fields, those OXM fields don't include any payload. The `oxm_length` field in OXM headers must be the length value defined for the OXM field, i.e. the payload length if the OXM field had a payload. For experimenter

OXM fields with variable payload size, the `oxm_length` field must be the maximum length of the payload. In some cases, only the packet register with the highest type field need to be included in the list.