PSARC 2007/429 Brussels Framework Interfaces Specification
Revision : 1.0

Sowmini Varadhan, Raymond Li, Artem Kachitchkine
{sowmini.varadhan, raymond.li, artem.katcitchkine}@sun.com

October 3, 2007

Contents

1 Introduction

This document presents the interfaces introduced/modified by PSARC/2007/429 ("Brussels - enhanced network driver config via dladm").

The components of the Brussels project are described in the Brussels Umbrella document ([1])). A description of the Brussels architecture can be found in the Brussels Design Specification [2]. PSARC/2007/429 will deliver the changes pertaining to Framework delivery in the kernel and user-space described in Section 3 of the Design Specification.

2 MAC/Driver Interface Changes

2.1 MC_SETPROP, MC_GETPROP

If the `MC_SETPROP` (`MC_GETPROP`) flag is set in the `mac_callbacks_t` passed by the driver to `mac_register()`, the the driver is passing a non-null callback function for setting (getting) properties in `mc_setprop` (`mc_getprop`)

2.2 mc_setprop

```
typedef int   (*mac_spropf_t)(void *driver, const char *prname,
      int propnum, uint_t valsize, const void *val);
mac_propf_t     mc_setprop;
```

The `mc_setprop` is an optional entry point provided by the driver to set a property value. The targetted property is identified by the string `prname`, and the value to be assigned to it is available in `val`. The `driver` argument contains a pointer to the `mi_driver` field that was provided by the driver during `mac_register()`.

For Public properties (see [2]), `prnum` additionally contains the encoded numeric mapping corresponding to `prname`. The valid set of name to encoded property numbers is defined in `<sys/dld.h>`.

2.3 mc_getprop

```
typedef int   (*mac_gpropf_t)(void *driver, const char *prname,
      int propnum, uint_t valsize, void *val);
mac_propf_t     mc_getprop;
```

The `mc_getprop` is an optional entry point provided by the driver to obtain the current value of a property identified by the string `prname`. The value is to be returned by the driver in the buffer `val`, whose size is `valsize` . The `driver` argument contains a pointer to the `mi_driver` field that was provided by the driver during `mac_register()`.

For Public properties (see [2]), `prnum` additionally contains the encoded numeric mapping corresponding to `prname`. The valid set of name to encoded property numbers is defined in `<sys/dld.h>`.

3 libdladm modifications

3.1 DLDIOCSETPROP/DLDIOCGETPROP

The DLDIOCSETPROP and DLDIOCGETPROP ioctls are issued by libdladm and intercepted in the kernel dld module to set/get properties using available `mc_setprop()` and `mc_getprop` callback functions that the targetted driver has (optionally provided).

3.2 dld_ioc_prop_val_t

Data passed between the user-space and the kernel for the DLDIOCSETPROP/DLDIOCGETPROP ioctl is formated as a a `dld_ioc_prop_val_t` structure defined in `<sys/dld.h>` as follows:

```
#define DLD_LINKPROP_NAME_MAX 256
typedef struct dld_ioc_prop_val_s {
        int     pr_version;
        uint_t  pr_flags;               /* private to libdladm */
        char    pr_linkname[LIFNAMSIZ];   /* interface name */
        uint_t  pr_num;                 /* <sys/dld.h>
        char    pr_name[DLD_LINKPROP_NAME_MAX];
        uint_t  pr_valsize;             /* sizeof pr_val */
        void    *pr_val;
} dld_ioc_prop_val_t;
```

Values passed in `pr_num` will be obtained from `<sys/dld.h>`, and will be constants prefixed with the symbol `DLD_PROP_`.

3.3 dladm_is_wlan_prop

Input: `prop_name`
Return **TRUE** if `prop_name` is the name of a WiFi property that can be intercepted and handled by the net80211 module, **FALSE** otherwise.

3.4 dladm_get_single_mac_stat

The functions `get_single_mac_stat()` privately defined in `/sbin/dladm` has been extracted into libdladm due to its general usefulness in dladm sub-commands for displaying link state for various link types.

```
dladm_get_single_mac_stat(char *link, char *stat, uint8_t type, void *retval)
```

Look up the `mac` kstat `stat` for `link`. Thus, the invocation
`dladm_get_single_mac_stat("link0","my_kstat",..)` would return the equivalent of the shell command `/usr/bin/kstat link:0:mac:my_kstat`. Returned value is in `*retval`, whose type is specified by `type`

4 dladm modifications

4.1 show-ether

`dladm show-ether [-px][-o <field>, ...] <link>`

Administrators will be able to monitor link-state of ethernet links using the `dladm show-ether` subcommoand. If the `link` argument is specified, `show-ether` will display information for the specified link, otherwise it will display information for all ethernet links.

In keeping with the conventions followed by `dladm` subcommands (see CR 6515065) the `-p` option will specify that the information should be displayed in a machine parseable format. Similarly, the `-o` flag will allow the administrator to provide a case-insensitive, comma-separated subset of the list of output fields that may be displayed, with the "all" keyword indicating that all available fields are to be displayed. Available fields that may be displayed by the `show-ether` sub-command are:

LINK	the name of the link
PTYPE	the type of parameter; one of `current, capable, adv, peeradv`
STATE	the current up/down state of the link
AUTO	yes/no indicating whether/not autonegotiation is permitted
SPEED-DUPLEX	speed/duplex permutations available for the parameter type
PAUSE	flow-control information (see Appendix C.16 of [2])
REM_FAULT	fault/none indicating presence/absence of fault

By default, the `LINK, PTYPE, STATE, AUTO, SPEED-DUPLEX, PAUSE` fields are displayed for the value of `PTYPE=CURRENT` to show the current state of the link. The `-x` option can be used with the `show-ether` subcommand to display extended state information, including information about MII capabilities advertised by the local and remote ethernet endpoints.

4.2 dladm sub-command modifications

Some of the anomalies flagged in CR 6515065 ("dladm output subcomands need uniform support for -o -p and output format") will be addressed by PSARC/2007/429. Specifically

- Support for the `-o` will be provided for the `show-dev`, `show-linkprop`, `show-secobj` subcommand.

  The Clearview UV project (PSARC/2006/499) is adding/modifying/enhancing sub-commands that display link/device status. This project will co-ordinate code-delivery schedule so that the final code that is delivered into Nevada will have a consistent output for `-o` and `-p` options for all of the `show-` sub-commands.

- the undocumented "-d" flag, used to print hexadecimal values for WiFi keys to assist in debugging, will be removed from the `show-secobj` sub-command. Instead, the value of stored

keys will be printed as part of the `-o all` or the `-o value` options to `show-secobj` after checking permissions for the uid of the process executing the `show-secobj` sub-command.

5 Supported Public properties

This section describes public properties that will be implemented in the driver(s) to be delivered as part of PSARC/2007/429.

5.1 Flow Control

```
dladm set-linkprop -p flowctrl=["no", "tx", "rx", "bi"] [link]
dladm get-linkprop -p flowctrl [link]
```

Establishes/retrieves flow-control modes that will be advertised by the device. Valid input is one of

no	no flow control enabled
rx	receive side of flow control is enabled (we can receive, and act upon, pause frames)
tx	send side of flow control enabled (can send pause frames, but ignore received pause frames
bi	bi-directional flow control

Note that the dladm support is intended to supercede the existing confusing semantics surrounding for `*_cap_pause` and `*_cap_asmpause` defintions in the `ieee802.3(5)` man page. Thus, the existing support (where it exists) for pause parameters defined in `ieee802.3(5)`, namely:

```
adv_cap_pause, adv_cap_amspause, lp_cap_pause, lp_cap_asmpause,
        cap_pause, cap-asmpause, link_pause, link_asmpause
```

will be made read-only, and accesible via kstat/ndd.

5.1.1 link_flowctrl_t

The string values `"no", "tx", "rx", "bi"` passed to libdladm as rvals for the `flowctrl` property are mapped to the following enumerated integers and passed down to the driver.

```
/* defined in <sys/mac.h> */
typedef enum {
        LINK_FLOWCTRL_NONE,
        LINK_FLOWCTRL_TX,
        LINK_FLOWCTRL_RX,
        LINK_FLOWCTRL_BIDIR
} link_flowctrl_t;
```

5.2 MII Properties

The following MII properties, whose semantics are described in `ieee802.3(5)`, will be supported via dladm:

```
ifspeed
link_duplex
link_up
adv_autoneg_cap
adv_1000fdx_cap
adv_1000hdx_cap
adv_100fdx_cap
adv_100hdx_cap
adv_10fdx_cap
adv_10hdx_cap
```

5.3 default_mtu

```
dladm set-linkprop -p default_mtu=[value] [link]
dladm get-linkprop -p default_mtu [link]
```

Set/get the maximum client SDU (Send Data Unit) for the `link`. Valid range for `value` is [68-65536]

5.4 ifspeed

```
dladm get-linkprop -p ifspeed [link]
```

Set/get the interface speed (in Mbps) for the `link`. See also Section 4.1.

References

[1] "Brussels - enhanced network driver configuration via dladm; Umbrella Specification". http://opensolaris.org/os/community/arc/caselog/2007/429/inception-materials/brussels-umbrella-txt.

[2] "Brussels - NIC configuration Design Specification". http://opensolaris.org/project/brussels/files/brussels.pdf.