NAME

ieee802.3 -- IEEE 802.3 Ethernet parameters and statistics

DESCRIPTION

The IEEE 802.3 standard specifies the details for Ethernet networking. This page describes the various statistics and tunables that device drivers supporting Ethernet commonly offer. Note that not every device or driver supports every one of these values, and many devices offer additional statistics and tunables that are specific to that hardware. See the device driver's documentation for those specific details.

Values that are statistics are visible kstat(1M), whereas properties are visible using the dladm(1M) **show-linkprop** subcommand. Tunables are properties that can be changed using the dladm(1M) **set-linkprop** subcommand. A more useful summary of current operational state can be seen with the dladm(1M) **show-ether** subcommand.

Statistics

The following statistics are accessible with kstat(1M). Note that some statistics are available in both 32- and 64-bit counters, in which case the name of the 64 bit statistic will be the same as the 32-bit, but with ‘‘**64**’’ appended". For example, **ipackets64** is the 64-bit version of the **ipackets** statistic. These are indicated with the special suffix [**64**] in the table below.

adv_cap_1000fdx
: Advertises 1000 Mbps full-duplex support.

adv_cap_1000hdx
: Advertises 1000 Mbps half-duplex support.

adv_cap_100fdx Advertises 100 Mbps full-duplex support.

adv_cap_100hdx Advertises 100 Mbps half-duplex support.

adv_cap_100T4 Advertises 100BASE-T4 support.

adv_cap_10fdx Advertises 10 Mbps full-duplex support.

adv_cap_10gfdx Advertises 10 Gbps support.

adv_cap_10hdx Advertises 10 Mbps half-duplex support.

adv_cap_autoneg Advertises auto-negotiation support.

adv_cap_asmpause
Advertises asymmetric flow control support.

adv_cap_pause Advertises flow control support.

adv_rem_fault Remote fault status sent to peer.

align_errors Mis-aligned frames received.

brdcstrcv Broadcast frames received.

brdcstxmt Broadcast frames transmitted.

cap_1000fdx Device supports 1000 Mbps full-duplex.

cap_1000hdx Device supports 1000 Mbps half-duplex.

cap_100fdx Device supports 100 Mbps full-duplex.

cap_100hdx Device supports 100 Mbps half-duplex.

cap_100T4 Device supports 100BASE-T4.

cap_10fdx Device supports 10 Mbps full-duplex.

cap_10gfdx Device supports 10 Gpbs.

cap_10hdx Device supports 10 Mbps half-duplex.

cap_asmpause Device supports asymmetric flow control.

cap_autoneg Device supports auto-negotiation.

cap_pause Device supports symmetric flow control.

cap_rem_fault Device supports remote fault notification.

carrier_errors Frames dropped due to loss of link.

collisions Collisions.

defer_xmts Transmits deferred due to link activity.

ex_collisions Frames dropped due to too many collisions.

fcs_errors Frames received with bad frame checksum.

first_collisions Frames with at least one collision.

ierrors Receive errors.

ifspeed Link speed in bits per second.

ipackets[**64**] Frames received successfully.

jabber_errors Jabber errors.

link_asmpause Asymmetric flow control; works together with **link_pause**. See the the description for it below.

link_autoneg Link was auto-negotiated.

link_duplex Link duplex status, values as follows:

0 Unknown.
1 Half-duplex.
2 Full-duplex.

link_pause Link flow control available; works together with **link_asmpause**. The meanings of these bits are:

pause	**asmpause**	**meaning**
0	0	No flow control.
1	0	Symmetric flow control.
0	1	Honor received pause frames.
1	1	Send pause frames when congested.

link_state Link state; 0 for down, 1 for up.

link_up Link is up if 1.

lp_cap_1000fdx Peer supports 1000 Mbps full-duplex.

lp_cap_1000hdx Peer supports 1000 Mbps half-duplex.

lp_cap_100fdx Peer supports 100 Mbps full-duplex.

lp_cap_100hdx Peer supports 100 Mbps half-duplex.

lp_cap_100T4 Peer supports 100BASE-T4.

lp_cap_10fdx Peer supports 10 Mbps full-duplex.

lp_cap_10gfdx Peer supports 10 Gbps.

lp_cap_10hdx Peer supports 10 Mbps half-duplex.

lp_cap_asmpause Peer supports asymmetric flow control.

lp_cap_autoneg Peer supports auto-negotiation.

lp_cap_pause Peer advertises flow control support.

lp_rem_fault Peer announces a remote fault.

macrv_errors Generic receive errors.

macxmt_errors Generic transmit errors.

multi_collisions Frames with more than one collision.

multircv Multicast frames received.

multixmt Multicast frames transmitted.

norcvbuf Receive frames dropped due to lack of resources.

noxmtbuf Transmit frames dropped due to lack of resources.

obytes[**64**] Bytes (octets) transmitted successfully.

oerrors Transmit errors.

oflo Overflow errors.

opackets[**64**] Frames successfully transmitted.

promisc Interface is in promiscuous mode.

rbytes[**64**] Bytes (octets) received successfully.

runt_errors Frames received that were too short.

sqe_errors Squelch errors.

toolong_errors Frames received that were too long.

tx_late_collisions Late collisions on transmit.

uflo Underflow errors.

unknowns Frames received with no local recipient.

xcvr_addr Transceiver address.

xcvr_id Transceiver vendor and device ID.

xcvr_inuse Identifies the type of transceiver in use. Values are as follows:

0 Unknown or undefined.
1 None.
2 10 Mbps
3 100BASE-T4
4 100BASE-X
5 100BASE-T2
6 1000BASE-X
7 1000BASE-T

Properties

The following parameters are accessible with dladm(1M). Some of these are normally only read-only. Other properties that are not specific to IEEE 802.3 / Ethernet links are also available via dladm(1M),

and are documented in its man page rather than here.

speed Link speed, in Mbps per second (dladm only).

duplex Link duplex, either "full" or "half".

state Link state, either "up" or "down".

mtu Maximum link frame size in bytes. See *Jumbo Frames*.

flowctrl Flow control setting, one of "no", "tx", "rx", or "bi". See *Flow Control*.

adv_10gfdx_cap Advertising 10 Gbps support.

en_10gfdx_cap Enable 10 Gbps support.

adv_1000fdx_cap Advertising 1000 Mbps full-duplex support.

en_1000fdx_cap Enable 1000 Mbps full-duplex.

adv_1000hdx_cap Advertising 1000 Mbps half-duplex support.

en_1000hdx_cap Enable 1000 Mbps half-duplex.

adv_100fdx_cap Advertising 100 Mbps full-duplex support.

en_100fdx_cap Enable 100 Mbps full-duplex.

adv_100hdx_cap Advertising 100 Mbps half-duplex support.

en_100hdx_cap Enable 100 Mbps half-duplex.

adv_10fdx_cap Advertising 10 Mbps full-duplex support.

en_10fhdx_cap Enable 100 Mbps full-duplex.

adv_10hdx_cap Advertising 10 Mbps half-duplex support.

en_10fhdx_cap Enable 10 Mbps half-duplex.

Auto-negotiation

With modern devices, auto-negotiation is normally handled automatically. With 10 Gbps and 1000 Gbps, it is mandatory. (10GBASE-T also requires full-duplex operation.) It is also *strongly* recommended for use whenever possible; without auto-negotiation the link will usually not operate unless both partners are configured to use the same link mode.

Auto-negotiation, when enabled, takes place by comparing the local capabilities that have been advertised (which must also be supported by the local device), with the capabilities that have been advertised by the link partner (peer). The first of the following modes that is supported by both partners is selected as the link negotiation result:

- 10 Gbps (10gfdx)
- 1000 Mbps full-duplex (1000fdx)
- 1000 Mbps half-duplex (1000hdx)
- 100 Mbps full-duplex (100fdx)
- 100BASE-T4 (100T4)
- 100 Mbps half-duplex (100hdx)
- 10 Mbps full-duplex (10fdx)
- 10 Mbps half-duplex (10hdx)

Advertisement of these modes can be enabled or disabled by setting the appropriate **en_** property in dladm(1M).

Auto-negotation may also be disabled, by setting the **adv_autoneg_cap** property to 0. In this case, the highest enabled link mode (using the above list) is ‘‘forced’’ for the link.

Flow Control

Link layer flow control is available on many modern devices, and is mandatory for operation at 10 Gbps. It requires that the link be auto-negotiated, and that the link be full-duplex, in order to function.

Flow control is applied when a receiver becomes congested. In this case the receiver can send a special frame, called a pause frame, to request its partner cease transmitting for a short period of time.

Flow control can be said to be either symmetric, in which case both partners can send and honor pause frames, or asymmetric, in which case one partner may not transmit pause frames.

The flow control mode used is driven by the **flowctrl** property. It has the following meanings:

"no" Neither send, nor honor pause frames.
"tx" Send pause frames, provided that the peer can support them, but do not honor them.

"rx" Receive and honor pause frames.
"bi" Both send and receive (and honor) pause frames.

The statistics for flow control (**adv_cap_pause**, **adv_cap_asmpause**, **lp_cap_pause**, **lp_cap_asmpause**, **link_pause**, and **link_asmpause**) are based on the properties exchanged in the auto-negotiation and are confusing as a result. Administrators are advised to use the **flowctrl** property instead.

Jumbo Frames

The IEEE 802.3 standard specifies a standard frame size of 1518 bytes, which includes a 4-byte frame checksum, a 14-byte header, and 1500 bytes of payload. Most devices support larger frame sizes than this, and when all possible parties on the same local network can do so, it may be advantageous to choose a larger frame size; 9000 bytes is the most common option, as it allows a transport layer to convey 8 KB (8192) of data, while leaving room for various link, network, and transport layer headers.

Note that the use of frames carrying more than 1500 bytes of payload is not standardized, even though it is common practice.

The **mtu** property is used to configure the frame size. Note that this is the size of the payload, and excludes the preamble, checksum, and header. It also excludes the tag for devices that support tagging (see *Virtual LANs* below).

Care must be taken to ensure that all communication parties agree on the same size, or communication may cease to function properly.

Note that the **mtu** property refers to the link layer property. It may be necessary to configure upper layer protocols such as IP to use a different size when this changes. See ifconfig(1M).

Virtual LANs

Most devices support virtual LANs (and also priority control tagging) though the use of a 4-byte tag inserted between the frame header and payload. The details of configuration of this are covered in the dladm(1M) manual.

Data Link Provider Interface (DLPI) Details

The correct method for applications to access Ethernet devices directly is to use the DLPI. See dlpi(7P) and libdlpi(3LIB) for further information.

The following DLPI parameters are presented to applications.

Maximum SDU	1500 (or larger, as determined by the **mtu** property.)
Minimum SDU	0

Address length	6
MAC type	DL_ETHER
SAP length	-2
Service mode	DL_CLDLS
Broadcast address	ff:ff:ff:ff:ff:ff (6 bytes with all bits set)

Note that if the application binds to SAP of 0, then standard IEEE 802.3 mode is assumed and the frame length is stored in place of the Ethernet type. Frames that arrive with the type field set to 1500 or less, are delivered to applications that bind to SAP 0.

Ethernet drivers on the support both DLPI style 1 and style 2 operation. Additionally, it is possible to configure provide ‘‘vanity’’ names to interfaces using the dladm(1M) **rename-link** subcommand. Such vanity names are only accessible using DLPI style 1.

NOTES

There may be other mechanisms available to configure link layer properties. Historically the ndd(1M) command, and driver.conf(4) files could be used to do this. These methods are deprecated in favor of dladm(1M) properties.

INTERFACE STABILITY

When present, the statistics and properties presented here are **Committed**. However, note that not every Ethernet device supports all of these, and some devices may support additional statistics and properties.

The DLPI and IEEE 802.3 itself are **Standard**.

SEE ALSO

dladm(1M), ifconfig(1M), kstat(1M), netstat(1M), ndd(1M), libdlpi(3LIB), driver.conf(4), dlpi(7P)

IEEE 802.3: Ethernet, IEEE Standards Association.

Data Link Provider Interface (DLPI), The Open Group, 1997.

STREAMs Programming Guide, Sun Microsystems, Inc., January 2005.