NAME

ng_vjc - Van Jacobson compression netgraph node type

SYNOPSIS

#include <sys/types.h>
#include <netinet/in.h>
#include <netinet/in_systm.h>
#include <netinet/ip.h>
#include <net/slcompress.h>
#include <netgraph/ng_vjc.h>

DESCRIPTION

The **vjc** node type performs Van Jacobson compression, which is used over PPP, SLIP, and other point-to-point IP connections to compress TCP packet headers. The ip hook represents the uncompressed side of the node, while the vjcomp, vjuncomp, and vjip hooks represent the compressed side of the node. Packets received on the ip will be compressed or passed through as appropriate. Packets received on the other three hooks will be uncompressed as appropriate. This node also supports "always pass through" mode in either direction.

Van Jacobson compression only applies to TCP packets. Only "normal" (i.e., common case) TCP packets are actually compressed. These are output on the vjcomp hook. Other TCP packets are run through the state machine but not compressed; these appear on the vjuncomp hook. Other non-TCP IP packets are forwarded unchanged to vjip.

When connecting to a ng_ppp(4) node, the ip, vjuncomp, vjcomp, and vjip hooks should be connected to the ng_ppp(4) node’s vjc_ip, vjc_vjcomp, vjc_vjuncomp, and vjc_ip hooks, respectively.

HOOKS

This node type supports the following hooks:

ip Upstream (uncompressed) IP packets.

vjcomp Downstream compressed TCP packets.

vjuncomp Downstream uncompressed TCP packets.

vjip Downstream uncompressed IP packets.

CONTROL MESSAGES

This node type supports the generic control messages, plus the following:

NGM_VJC_SET_CONFIG (**setconfig**)

This command resets the compression state and configures it according to the supplied struct ngm_vjc_config argument. This structure contains the following fields:

```
struct ngm_vjc_config {
  u_char   enableComp;    /* Enable compression */
  u_char   enableDecomp;  /* Enable decompression */
  u_char   maxChannel;    /* Number of outgoing channels - 1 */
  u_char   compressCID;   /* OK to compress outgoing CID's */
};
```

When enableComp is set to zero, all packets received on the ip hook are forwarded unchanged out the vjip hook. Similarly, when enableDecomp is set to zero, all packets received on the vjip hook are forwarded unchanged out the ip hook, and packets are not accepted on the vjcomp and vjuncomp hooks. When a node is first created, both compression and decompression are disabled and the node is therefore operating in bi-directional "pass through" mode.

When enabling compression, maxChannel should be set to the number of outgoing compression channels minus one, and is a value between 3 and 15, inclusive. The compressCID field indicates whether it is OK to compress the CID header field for outgoing compressed TCP packets. This value should be zero unless either (a) it is not possible for an outgoing frame to be lost, or (b) lost frames can be reliably detected and immediately reported to the peer's decompression engine (see NGM_VJC_RECV_ERROR below).

NGM_VJC_GET_STATE (**getstate**)

This command returns the node's current state described by the struct slcompress structure, which is defined in <*net/slcompress.h*>.

NGM_VJC_CLR_STATS (**clrstats**)

Clears the node statistics counters. Statistics are also cleared whenever the enableComp or enableDecomp fields are changed from zero to one by a NGM_VJC_SET_CONFIG control message.

NGM_VJC_RECV_ERROR (**recverror**)

When the peer has CID header field compression enabled, this message must be sent to the local **vjc** node immediately after detecting that a received frame has been lost, due to a bad checksum or for any other reason. Failing to do this can result in corrupted TCP stream data.

SHUTDOWN

This node shuts down upon receipt of a NGM_SHUTDOWN control message, or when all hooks have

been disconnected.

SEE ALSO

netgraph(4), ng_iface(4), ng_ppp(4), ngctl(8)

V. Jacobson, *Compressing TCP/IP Headers*, RFC 1144.

G. McGregor, *The PPP Internet Control Protocol (IPCP)*, RFC 1332.

HISTORY

The **ng_vjc** node type was implemented in FreeBSD 4.0.

AUTHORS

Archie Cobbs <*archie@FreeBSD.org*>

BUGS

As the initialization routine in the kernel implementation of Van Jacobson compression initializes both compression and decompression at once, this node does not allow compression and decompression to be enabled in separate operations. In order to enable one when the other is already enabled, first both must be disabled, then both enabled. This of course resets the node state. This restriction may be lifted in a later version.

When built as a loadable kernel module, this module includes the file *net/slcompress.c*. Although loading the module should fail if *net/slcompress.c* already exists in the kernel, currently it does not, and the duplicate copies of the file do not interfere. However, this may change in the future.