Network Management for IBM Communications Controller for Linux on System z (CCL) Data Link Switching (DLSw)

June 2006

Table of Contents

1 Contributors

- Chris Chato (technical contact)
- Andy Richter
- Joe Mason
- Phil Trent
- Brian Baker
- Tom McSweeney
- Mike Law

2 Overview

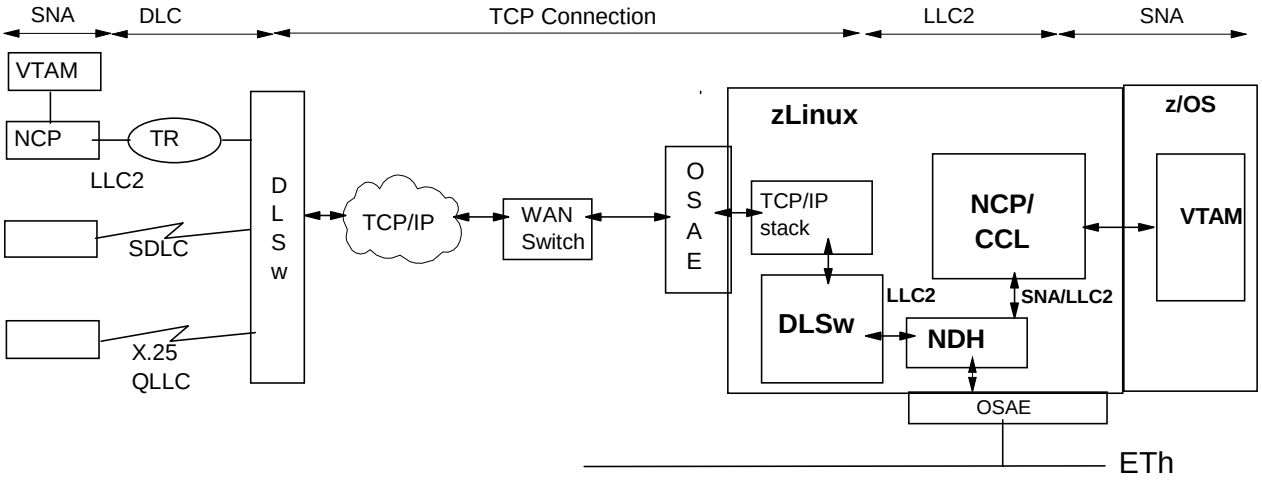
This document explains the network management strategy for Data Link Switching (DLSw) on the IBM Communications Controller for Linux on System z (CCL).

Unlike Cisco routers, CCL DLSw is not part of a router. Its sole purpose is to provide a transport across the TCP/IP cloud for the Network Control Program (NCP) and CCL SNA traffic. Data received by CCL DLSw from TCP/IP is passed by Network Device handler (NDH) to the appropriate CCL in the same Linux image. It is never routed out to an actual device. No bridging is ever involved.

The state of the SNA sessions running across CCL DLSw is of primary interest, rather than the state of CCL DLSw, itself. As long as sessions are running smoothly, examining the data at the SNA level is of most interest. It is only when a problem occurs that you may need to drill down into the lower levels to determine the source of the problem.

As is the case today, the rich set of SNA network management tools can be used to manage NCP SNA sessions. Similarly, the TCP/IP layer can be managed using the standard TCP/IP network management tools.

The following diagram illustrates the positioning of CCL DLSw in a network:



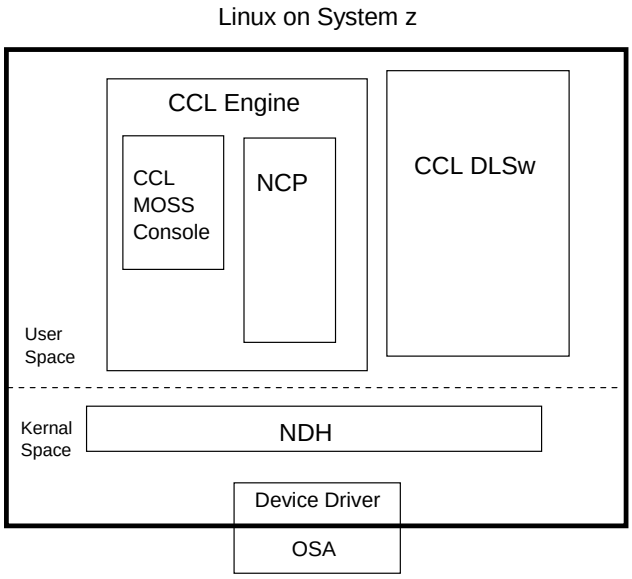
3 CCL DLSw

CCL DLSw is a user space application that runs in a Linux environment on a System z image. The only exit or entry points to CCL DLSw are the following:

- NDH socket
- Linux TCP/IP stack socket.

CCL DLSw always runs in the same Linux image with the NCPs and CCLs it serves.

Logical arrangement of CCL components



3.1 Monitoring and managing CCL and NDH

CCL DLSw requires both CCL and NDH. Neither CCL nor NDH have any specific awareness of DLSw. To CCL, DLSw is simply the opposite end of an LLC2 connection. The interface between NDH and DLSw is the NDH promiscuous socket. For information on monitoring and managing CCL and NDH, refer to the following:

- Communications Controller for Linux on System z Library (http://www-306.ibm.com/software/network/ccl/library/).

4 Networks Layers

In a CCL DLSw environment, the main layers from a network management perspective are:

- SNA
- DLSw
- TCP/IP.

The SNA layer has no direct knowledge of the DLSw sessions over which its SNA sessions run. Likewise, DLSw has no direct knowledge of the SNA sessions running on top of it.

4.1 Relationships between resources

The NCP and CCL resources which map to CCL DLSw sessions are NCP token-ring logical PUs (either TIC2 or TIC3). Network management tools identify NCP logical PUs by their PU names from the XCA Switched Major node definitions for BNN resources or from the NCP gen for INN resources. DLSw sessions are identified by their Data Link identifiers (DLIDs) which consist of the origin SMAC/SSAP and target DMAC/DSAP pairs.

When the connection is active, IBM NTuneMON can be used to correlate the NCP token-ring physical line with the underlying DLSw connection. The IBM NTuneMON token-ring SNA station panel (for NCP token-ring logical PUs) shows the local and remote MAC addresses and SAPs. These MAC/SAP pairs correspond to the DLID of the DLSw session.

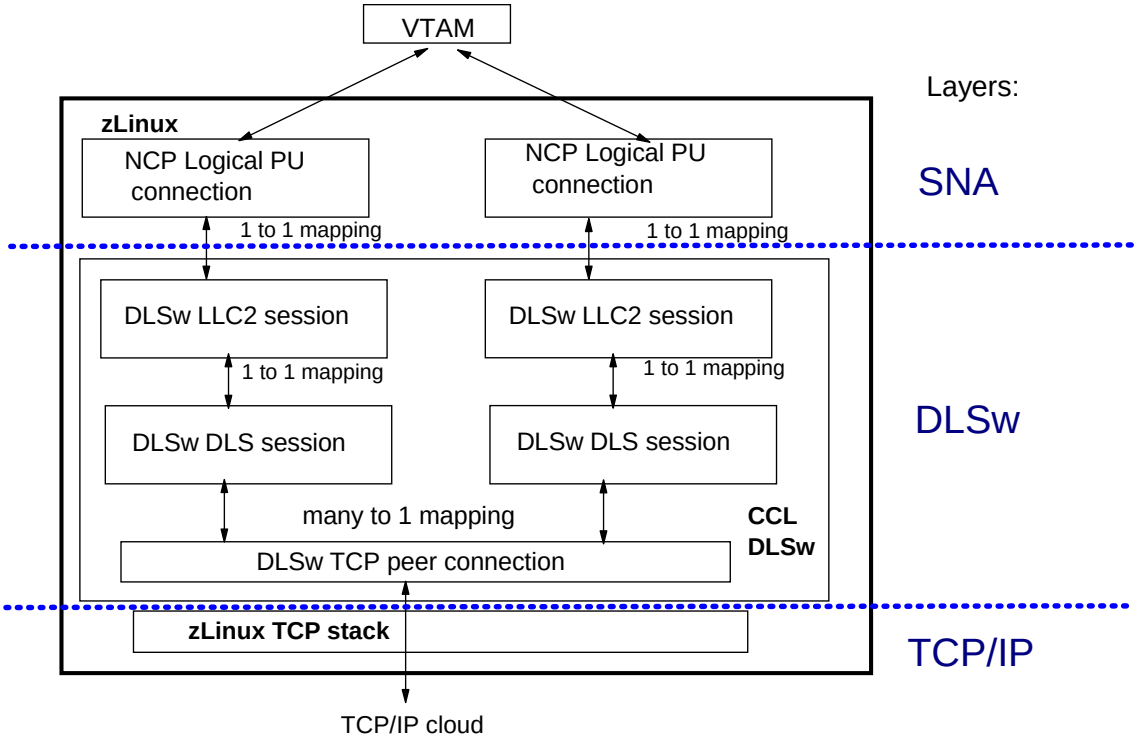
Common network management tools do not provide an explicit mapping between NCP token-ring logical PUs and DLSw DLIDs. This is not unique to CCL DLSw.

In general, the PU name to DLID mapping can be determined from a knowledge of the network topology and definitions. Remember that CCL DLSw is always in the same Linux image with the NCPs using it.

From a network management perspective, the important resources in CCL DLSw are the following:

- LLC2 sessions (1 to 1 with NCP logical PUs)
- DLS sessions (1 to 1 with LLC2 sessions and, thus, with NCP logical PUs)
- TCP peer connections (many DLS sessions per TCP peer connection).

The following diagram illustrates the relationship between resources:



4.2 SNA layer monitoring

As is the case today, you can still use the rich set of SNA network management tools to monitor and manage your NCP SNA sessions. Running over CCL DLSw does not change this.

The following are commonly used SNA network management tools:

- VTAM
  - Activate and deactivate SNA resources
  - Displays operational information such as connection status, outages, activation results, and so on.
- IBM Tivoli NetView for z/OS, particularly:
  - NetView Management Console
  - Network Logical Data Manager (NLDM)
  - Hardware monitor (NPDA)
    - Detects hardware and software problems by way of events
  - SNA Topology Manager

        - Provides SNA topology view
        - Shows status and connectivity
        - Provides detailed views of NCP/VTAM composite nodes
    - Network Performance Monitor (NPM)
        - Displays NCP performance statistics
- IBM NTuneMON
    - Runs on top of IBM Tivoli NetView for z/OS
    - Shows NCP resources and status
    - Shows local and remote MAC addresses and SAPs for token-ring logical PUs.
- Omegamon for Mainframe Networks
    - Follow-on product to IBM Tivoli NetView for z/OS

4.3 DLSw layer monitoring

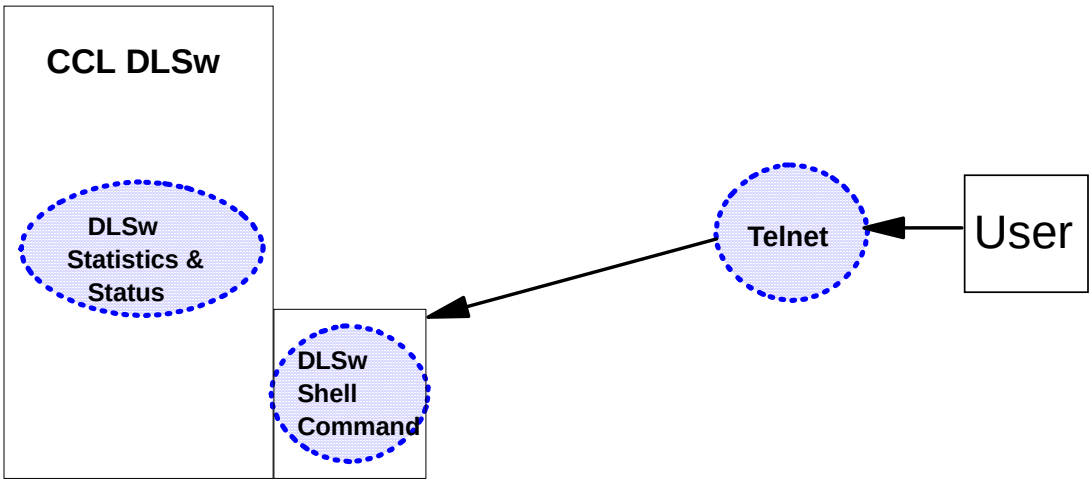
As mentioned previously, the state of the SNA sessions running across CCL DLSw is of primary interest, rather than the state of CCL DLSw, itself. It is only when a problem occurs that you may need to drill down into the lower levels to determine the source of the problem.

The main CCL DLSw resources to monitor are as follows:
- TCP peer connections (many DLS sessions per TCP peer connection)
- DLS sessions (1 to 1 with LLC2 sessions)
- LLC2 sessions (1 to 1 with NCP logical PUs).

Note: CCL DLSw does not support SNMP.

CCL DLSw console

The CCL DLSw console provides a set of predefined commands for displaying configured values, statistics, and status. Use telnet or ssh to access the console as illustrated in the following diagram:



Use the CCL DLSw console to display the following connections:
- TCP
- DLSw
- LLC2.

You can dynamically perform the following from the CCL DLSw console:
- Add or delete TCP peers
- Enable or disable dynamic peer support
- Enable or disable tracing.

CCL DLSw log files

The CCL DLSw log files are located in ccl_install_dir/logs/dlsw.logx. Errors are always logged. Enable tracing and debugging to capture additional data only when recommended by IBM service.

Cisco router console

If the other end of the CCL DLSw connection is a Cisco router, the Cisco router console can be used to obtain information about that end of the DLSw connection.

Use telnet, ssh, or the IBM Tivoli NetView for z/OS RUNCMD to Cisco's SNA service point to access the console.

Use the Cisco router console to display the following connections:
- TCP
- DLSw
- DLC.

Note: Since it is a router (supporting routing, bridging, vlans, and so on), the Cisco router console displays much more information in addition to information about DLSw.

Linux commands

You can also obtain information about CCL DLSw from the Linux on System z console using common Linux commands or tools such as:
- netstat to display the DLSw socket status and connectivity
- ping to test connectivity
- top to display DLSw process CPU and memory usage
- lsmod to confirm that NDH is installed and running
- Other standard Linux IP management commands, as appropriate
- ethereal to display traffic on the IP interface.

4.3.1 LLC2

Since the LLC2 connection between NCP, CCL, and CCL DLSw is just across the NDH socket, its statistics and configuration are not that crucial. Since they are always together in the same Linux image, they will not experience the normal networking problems that are experienced by LLC2 connections flowing over a real network.

However, the LLC2 configuration parameters must be compatible between the NCP gen and the CCL DLSw configuration or timeouts may occur.

4.3.2 Scripting and logging tools for capturing the CCL DLSw console output

Use the following tips to capture output from CCL DLSw console commands:

- Bring up the telnet or ssh session in a mode that logs the session output to a file. Note: If putty is used to connect, its "Copy all to clipboard" command can capture the output after the fact.
- Use the expect tool to write and run scripts to automate and save the results of CCL DLSw console commands (see "expect tool" in the Appendix).
- Use other Linux scripting tools.

4.4 TCP layer monitoring

When configured appropriately, the Linux on System z TCP/IP stack appears as a router in the IP network and can be managed using the standard IP network management tools.

The following are commonly used TCP/IP network management tools:

- Linux console provides commands and tools such as:
    - netstat to display the DLSw socket status and connectivity
    - ping to test connectivity
    - top to show process CPU and memory usage
    - Other standard Linux IP management commands
    - ethereal tool to display traffic on the IP interface
- IBM Tivoli Monitor (ITM)
    - Centralized console for viewing router information
    - Similar to information available with the Linux console
- IBM Tivoli NetView
    - IP topology
    - Status and connectivity
- IBM Tivoli NetView for z/OS, particularly:
    - NetView Topology Management:
        - IP view
        - Status and connectivity
- Omegamon for Mainframe Networks
    - Follow-on product to IBM Tivoli NetView for z/OS

5 Appendices

5.1 expect tool

expect is an open source tool, written by Don Libes, for automating text tasks. You must install it. Then, issue the command, ***man expect***, for details on how to use it.

Use expect to automate the issuing of CCL DLSw console commands and to capture the output of commands. You can also use it to analyze the data in the CCL DLSw log files.

The following sample expect file (dlscmd.exp) performs the following tasks:
1. Takes a DLSw console command as input
2. Logs onto the CCL DLSw console (using **dlsw_password**)
3. Issues the input DLSw console command
4. Appends the output to the following file:
   /opt/ibm/Communication_Controller_for_Linux/logs/dlscmd.log
5. Returns the output to the regular Linux console from which the expect file was run.

Sample dlscmd.exp file:

```
#!/usr/bin/expect
#
# dlscmd Expect Script
#  This script will log on to the local DLSW and issue one command
#  the command is the argument to the script
#  ex. './dlscmd.exp show interfaces'
#

# Set a short timeout.  If DLSw server not around or hung don't wait forever
set timeout 3

# dlspass is the password for the DLSw console
set dlspass dlsw_password

# dlsport is the port number for DLSw console logins
set dlsport 2002

log_user 0
if { "x$argv" == "x" } {
send_user "\nPlease enter a valid DLSw console command to run.\n"
send_user "\n for example:  '$argv0 show interfaces' \n"
exit 1 }

# log output. Appends to existing file
log_file /opt/ibm/Communication_Controller_for_Linux/logs/dlscmd.log
# or overlay log file every time
#log_file -noappend dlscmd.log

eval spawn telnet 127.0.0.1 $dlsport
expect {
       "word: >" { send "$dlspass\n" }
       timeout   { send_user "\nDLSw Server did not respond on port $dlsport within timeout.\n"
```

```
                exit 1 }
        eof     { send_user "\nDLSw is not listening on port $dlsport \n"
                exit 1 }
}
expect "DLSw>"
send_user "#### [clock format [clock seconds] -format %Y-%m-%d,%T] ####\nDLSw>"
send "$argv\n"
expect "DLSw>"
set output1 [string trimright $expect_out(buffer) "DLSw>"]
send_user "$output1"
send "quit\n"
exit
```

Sample run

```
linux45:/home/smith/expect # ./dlscmd.exp show int
#### 2006-05-12,09:31:20 ####
DLSw>show int
Interfaces:
Name      Mac Address            MTU
--------     -----------------         ------
lo          00:00:00:00:00:00     16436
sit0        00:00:00:00:00:00      1480
eth0        02:00:00:00:00:17      1500
*ndh0       00:00:00:00:00:00      16384
```

This output is also appended to:

- /opt/ibm/Communication_Controller_for_Linux/logs/dlscmd.log.

5.2 DLSw TCP peer connection problems

If DLSw TCP peer connections are not becoming active, verify a few very basic configuration settings prior to embarking on complicated TCP/IP analysis. For example:

- Is CCL DLSw coded with "Dynamic peer" enabled?
    - If not, CCL DLSw only accepts TCP peer connections with configured peers.
- If the other end of the DLSw TCP connection is a Cisco router, is the Cisco router coded with "promiscuous" to allow dynamic peers?
- If a DLSw peer is explicitly coded at each end of the DLSw connection, is it configured as “active” on at least one end?

5.3 Sample MAC/SAP pair session correlation

As previously explained, there is a one-to-one mapping between NCP Logical PU connections, DLSw LLC2 sessions, and DLSw DLS sessions. The following section provides screen captures showing the correlation between these resources based on the MAC/SAP pairs that uniquely define the resource. In the following example, the local

MAC and SAP are highlighted in blue and the remote ones in red.

5.3.1 IBM NTuneMON displays

The following Token-Ring Adapter List panel illustrates the local MAC address for the adapters:

```
 ATUTC  F72LAN1         Token-Ring Adapter List           NTuneMON V3R2  11:36
 Total Logical Lines Active= 4    Total Defined= 14   User Alarm=  80%
 PORT LINE     MAC      LOGICAL LINES
 ADD  ADDR   ADDRESS    ACTIVE  TOTAL  T/R MXTSL RCVBF TYPE TIC  STATUS

____|____|____________|_____________|___|_____|_____|____|____|_______________
  ANY   -       -          -  0    0     -    -     -    -    -     -
  92  1092 400072C4D3E2 100% 1    1    16M 16732 32000 ANY  TIC2 INACTIVE
  93  1093 420937450330 100% 1    1    16M 16732 32000 ANY  TIC2 INACTIVE
  ANY   -       -          -  0    0     -    -     -    -   TIC3   -
  80  2080 40000F722080   0% 0    2    16M 16732   -   INN  TIC3 INACTIVE
  0   2240 400037450280  16% 1    6    16M  2012   -   ANY  TIC3 ACTIVE
  72  2272 420937450330   0% 0    3    16M  2012   -   ANY  TIC3 ACTIVE
  68  2368 430072C4D3E2 100% 1    1    16M  2012   -   ANY  TIC3 ACTIVE
  -     -       -          -  -    -     -    -     -    -    -     -
 _____________________________________________________________________________
 | FOR DIRECT FIND ENTER ONE:    | STATUS DEFAULT   | VIEW DEFAULT   |
 | XID IDBLK-IDNUM=              | ACTIVE  INACTIVE | MAC SUB CPNAME |
 | T/R MAC ADDRESS=              | ALL (ACT/INACT)  | IDBLK   LINE   |
 | PU-T2.1 CPNAME =              |                  | QSTAT  (DUAL)  |
 | NCP LINE NAME  =              | STATUS= ACTIVE   | SUB            |
 -------------------------------------------------------------------------
 =>
F1=HELP F2=SUBAREA F3=RETURN 4=BNN IDBLK-IDNUM F5=MAC ADDR F6=ROLL F7=BACK
 ENTER=PHYSICAL TIC F8=FWD F9=ALL TICS F10=HEX F11=FUNC F12=REFRESH PA2=LOG
```

The following Token-Ring SNA station panel shows the details for a particular NCP token-ring logical PU, including the local and remote MACs and SAPs:

```
 ATUTL  F72LAN1       Token-Ring SNA Station   D05FVT5    NTuneMON V3R2  11:37
 Route through 0   Bridges    LLC Status: Link Opened             BDGLF =  -
                                                                  T1LOC = 3.1
                                                                  T1REM = 4.1
                                                                  T2LOC = 0.2
                                                                  T2REM = 0.2
                                                                  N3GEN = 3
  CONGESTION = NONE       STATION = NORMAL                  #PIUS/BLOCK = 0
     MODE    =                      DYNWIND(NW,DW,DWC)=  04     1      0
 |     NCP TOKEN-RING TIC          | DESTINATION STATION Subarea Station
|
 | Line Addr= 2368  TIC3  16M     | IDBLK-IDNUM= -             LINE= L2368L1
|
 | T/R Addr = 430072C4D3E2        | T/R Address= 400005C4D3E2  DSAP = 24
|
 | SSAP = 24                      | UNIQUE     = YES    PreDefineRoute= N/A
|
 | PHYPORT  = 68       ACTIVE     | CPNAME     = -             ACTIVE
|
 | OUTBOUND QUE LENGTH =  N/A     | TG=  1      SUBAREA=  44  NETID= NETX
|
 | ______________________________ |_____________ _____________________________
|
```

```
 | FRAME STATISTICS:      TRANSMIT       RECEIVE |         CUR:       GEN:
|
 | MAXTSL/RCVBUFC       =  2012   <---->    -    |MAXDATA  = 2012     0
|
 | Frames Count         =  -      --><--  -      |MAXOUT   = 24       24
|
 | WW / Outs. Frame ct  =  -/-                   |BLOCK    = 0        0
|
 | N3 / I-Frame Rcv ct  =                -/-     |PIUCHECK =  NO       NO
|
 =>
 F1=HELP  F2=DISG STATION PU  F3=RETURN                    F5=MODIFY  F6=ROLL
 ENTER=TG DETAILS(SUBA ONLY)               F10=HEXF12=REFRESH PA1=EXIT PA2=LOG
```

5.3.2 CCL DLSw console displays

The following output from the CCL DLSw console commands shows the resources corresponding to the NCP token-ring Logical PU displayed in the IBM NTuneMON panels above.

TCP peer connections

```
DLSw>show tcp peer
   Multicast
   IP Address      IP Address      Conn State     CST Version  ActSes
SesCreates
   --------------- --------------- -------------- --- -------- ------ ---------
-
1                  9.42.103.144    ESTABLISHED     a  AIW V2R0      1
1
DLSw>
```

DLS sessions

```
DLSw>show dls sessions
         Source         Destination      State     Flags    Dest IP Addr    Id
     --------------  --------------  ---------  -------  --------------  ----
   1 430072c4d3e2 24  400005c4d3e2 24  CONNECTED           9.42.103.144       0


DLSw>show dls llc-sessions
        SAP Int Name   Remote Addr   Local Addr    State         RIF
    1.  24  ndh0       400005c4d3e2  430072c4d3e2  CONTACTED
```

LLC2 sessions

```
DLSw>show llc2 sap 24
SAP:                            24
Interface:                      12,ndh0
Reply Timer(T1):                3 sec
Receive ACK Timer(T2):          3 100milisec (note: not used when N3=1)
Inactivity Timer(Ti):           30 sec
MAX Retry Value(N2):            8
MAX I-Field Size(N1):           16384
Rcvd I-frames before Ack(N3):   1
Transmit Window Size(Tw):       7
Acks Needed to Inc Ww(Nw):      1

Frame Type          Xmt      Rcvd
```

```
UI-frames:            0         0
TEST-frames:          0         2
XID-frames:           0         2
I-frames:             1         1
RR-frames:           87        88
RNR-frames:           1         0
REJ-frames:           0         0
SABME-frames:         0         1
UA-frames:            1         0
DISC-frames:          0         0
DM-frames:            0         0
FRMR-frames:          0         0
I-frames Discarded by LLC:      0
I-frames Refused by LLC user:   0

Cumulative number of sessions:  1
Number of active sessions:      1
Session ID                                          Local
(int-sap-id)  Remote MAC          Local MAC         SAP     State
0000-24-0000  40:00:05:c4:d3:e2  43:00:72:c4:d3:e2  24      LINK_OPENED


DLSw>show llc2 session 0000-24-0000
Session ID:                     0000-24-0000
Interface:                      12,ndh0
Local MAC addr:                43:00:72:c4:d3:e2
Remote MAC addr:                40:00:05:c4:d3:e2
Local SAP:                       24
Remote SAP:                      24
RIF:                            None
Access Priority:                0
State:                          LINK_OPENED
Reply Timer(T1):                3 sec
Receive ACK Timer(T2):          3 100milisec (note: not used when N3=1)
Inactivity Timer(Ti):           30 sec
MAX I-Field Size(N1):           16384
MAX retry Value(N2):            8
Rcvd I-frames before Ack(N3):   1
Transmit Window Size(Tw):       7
Working Transmit Size(Ww):      7
Acks Needed to Inc Ww(Nw):      1
Current Send Seq (Vs):          1
Current Rcv Seq (Vr):           1
Last ACK'd sent frame(Va):      1
No. of frames in ACK pend q:    0
No. of frames in Tx pend q:     0
Local Busy:                     NO
Remote Busy:                    NO
Poll Retry count:               8
Appl output flow stopped:       NO
Send process running:           YES

Frame Type          Xmt      Rcvd
I-frames:             1         1
RR-frames:           90        91
RNR-frames:           1         0
REJ-frames:           0         0
I-frames Discarded by LLC:      0
I-frames Refused by LLC user:   0
```

5.3.3 NDH displays

The following output from NDH commands may be of interest when dealing with CCL DLSw to show which MAC-SAP pairs NDH knows about and whether traffic is following.

```
cat /proc/net/ndh/socklist
NDH9700I SOCKLIST - Revision:1.78.1.8.bootleg
ReadSock-Inode   WriteSock-Inode  UID      PROTO STATE         MAC-SAP Pairs
32343            32344            0        NDH-TR CONNECTED
                                                              430072c4d3e2-04
                                                              430072c4d3e2-24
32325            32325            0        NDH-TR CONNECTED
                                                              000000000000-24
32204            32205            0        NDH-TR CONNECTED
                                                              4290eca2c00c-04
                                                              4290eca2c00c-0c
                                                              4290eca2c00c-14
32170            32171            0        NDH-TR CONNECTED
                                                              400037450280-04
                                                              400037450280-0c


cat /proc/net/ndh/statistics
NDH Statistics - Revision:1.78.1.8.bootleg
GLOBAL:
  5 Minute Statistics:
     Inbound Network-To-NDH Rate
          0 byte/sec   0 packets/sec
     Outbound NDH-To-Network Rate
          1 byte/sec   0 packets/sec
  Cumulative Statistics:
     Inbound Network-To-NDH
          64390361 packets         2388965835 byte
          18856990 packets discard 173553857 byte discard 0 erro
     Outbound NDH-To-Network
          45500712 packets         2242309393 byte
          0 packets discard 0 byte discard
     Packing:
        Inbound User-To-NDH
          8269988 packed pkts     6474841 nonpacked pkts
        Outbound NDH-To-User
          0 packed pkts     45500714 nonpacked pkts
TUNNEL SPECIFIC:  Cumulative Statistics:
SOCKETPAIR: 32343 32344  Name: NA  MAC: 43:00:72:C4:D3:E2
SAPS: 04 24
     Inbound User-To-NDH
          59 packets         1227 byte
          0 packets discard 0 byte discard
          0 packed pkts     59 nonpacked pkts
     Outbound NDH-To-User
          68 packets         1938 byte
               0 packed pkts     1106 nonpacked pkts
     Outbound NDH-To-User
          715 packets         37712 byte
          0 packets discard 0 byte discard
          12 packed pkts     687 nonpacked pkts
```