Layer 2 on a Native LPAR

Sample Definitions for Communications Controller for Linux on System z9 and zSeries

Target Audience

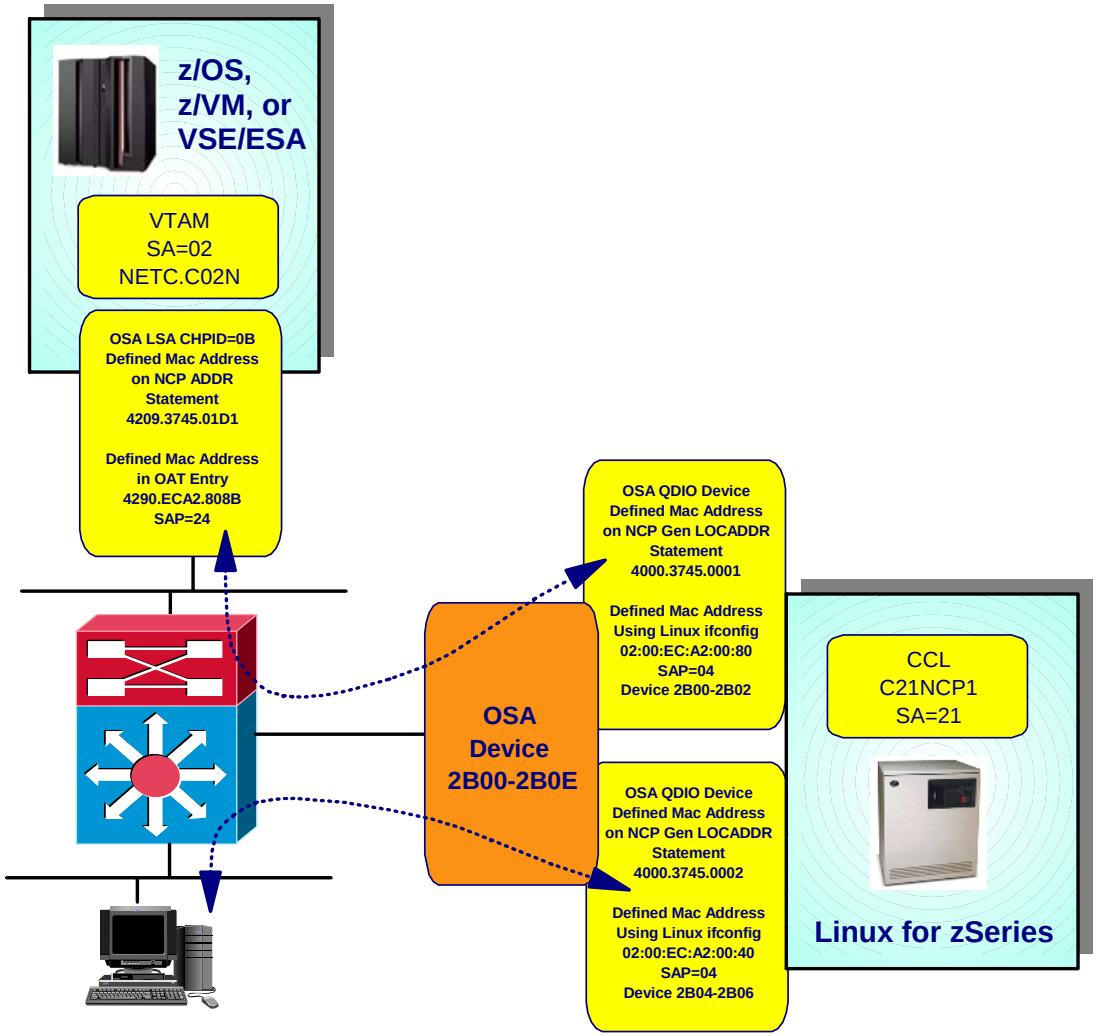
- Customers who wish to reduce the number of copper OSA adapters when using Communication Controller for Linux on System z9 and zSeries (CCL). QDIO in Layer 2 mode will support both INN and BNN connections.

Purpose of this Paper

The intent of this paper is to provide a sample solution for customers during the migration from 3745/3746-900 FEPs to Communication Controller for Linux on System z9 and zSeries (CCL). This document will provide working examples of the following:

- VTAM XCA Major Node – VTAM to CCL
- NCP Physical and Logical lines – CCL to 3745
  - VTAM to Communication Controller for Linux on System z9 and zSeries
  - 3745 to Communication Controller for Linux on System z9 and zSeries
  - Commands to define OSA QDIO adapters in Layer 2 mode

Test Configuration



Resources Used for Solution Verification

- One z/OS Communications Server
- One Linux ID running in a LPAR
  - 2048 mb of memory
  - 2 Real CPs
  - 2 3390-3 DASD volumes
- One OSA Fast Ethernet adapter
  - Required for LSA connection in z/OS.
- One QDIO OSA Adapter
- Layer 2 or Layer 3 Ethernet Switches

Define the Layer2 Interface to Linux -- Hardware

Create the script hwcfg-qeth-bus-ccw-0.0.2b00 in the /etc/sysconfig/hardware directory

```
#!/bin/sh
#
STARTMODE='auto'
MODULE='qeth'
MODULE_OPTIONS=''
MODULE_UNLOAD='yes'

SCRIPTUP='hwup-ccw'
SCRIPTUP_ccw='hwup-ccw'
SCRIPTUP_ccwgroup='hwup-qeth'
SCRIPTDOWN='hwdown-ccw'

# CCW_CHAN_IDS are the device addresses
CCW_CHAN_IDS='0.0.2b00 0.0.2b01 0.0.2b02'

# CCW_CHAN_NUM set the number of channels for this device
CCW_CHAN_NUM='3'

# CCW_CHAN_MODE sets the port name for an OSA-Express device
CCW_CHAN_MODE='GIGE2B00'

# QETH_LAYER2_SUPPORT enables Layer2 support for this device.
QETH_LAYER2_SUPPORT=1
```

Define the Layer2 Interface to Linux -- Network

Create this script ifcfg-qeth-bus-ccw-0.0.2b00 in the /etc/sysconfig/network directory

```
LLADDR='02:00:ec:a2:00:80‘
BOOTPROTO='none'
STARTMODE='onboot'
UNIQUE=''‘
```

- By using LLADDR, we can set the MAC address to any value necessary. This keyword may be different in Red Hat releases.
- The two scripts will need to be replicated for device address 2B00. Device 2B00 will have an LLADDR='02:00:ec:a2:00:80‘
- MAC Address defined on the NCP LOCADDR statement is the non-canonical version of this address – 4000.3745.0001

Define the Layer2 Interface to Linux -- Hardware

Create the script hwcfg-qeth-bus-ccw-0.0.2b04 in the /etc/sysconfig/hardware directory

```
#!/bin/sh
#
STARTMODE='auto'
MODULE='qeth'
MODULE_OPTIONS=''
MODULE_UNLOAD='yes'

SCRIPTUP='hwup-ccw'
SCRIPTUP_ccw='hwup-ccw'
SCRIPTUP_ccwgroup='hwup-qeth'
SCRIPTDOWN='hwdown-ccw'

# CCW_CHAN_IDS are the device addresses
CCW_CHAN_IDS='0.0.2b04 0.0.2b05 0.0.2b06'

# CCW_CHAN_NUM set the number of channels for this device
CCW_CHAN_NUM='3'

# CCW_CHAN_MODE sets the port name for an OSA-Express device
CCW_CHAN_MODE='GIGE2B00'

# QETH_LAYER2_SUPPORT enables Layer2 support for this device.
QETH_LAYER2_SUPPORT=1
```

Define the Layer2 Interface to Linux -- Network

Create this script ifcfg-qeth-bus-ccw-0.0.2b04 in the /etc/sysconfig/network directory

```
LLADDR='02:00:ec:a2:00:40‘
BOOTPROTO='none'
STARTMODE='onboot'
UNIQUE=''‘
```

- By using LLADDR, we can set the MAC address to any value necessary. This keyword may be different in Red Hat releases.
- The two scripts will need to be replicated for device address 2B04. Device 2B04 will have an LLADDR='02:00:ec:a2:00:40‘
- MAC Address defined on the NCP LOCADDR statement is the non-canonical version of this address – 4000.3745.0002

C02XCA – XCA Major Node Definitions

```
C02XCA   VBUILD  TYPE=XCA
*
C02ETHPT PORT    MEDIUM=CSMACD,ADAPNO=0,SAPADDR=24,CUADDR=2EBA,          X
                 TIMER=100
C02ETHGP GROUP   DIAL=NO,ISTATUS=ACTIVE
*
C02ETHL2 LINE    USER=SNA,ISTATUS=ACTIVE
C02ETHP2 PU      MACADDR=0200ECA20080,PUTYPE=5,SUBAREA=21,TGN=1,         X
                 SAPADDR=04,ALLOWACT=YES
```

C21NCP1 – NTRI Physical Line Definitions

```
*********************************************************************
* Physical NTRI Lines
*********************************************************************
*
C21PTRG1 GROUP ECLTYPE=(PHY,ANY),ADAPTER=TIC2,ANS=CONT,MAXTSL=16732,    X
               RCVBUFC=32000,USSTAB=AUSSTAB,ISTATUS=ACTIVE,XID=NO,      X
               RETRIES=(20,5,5),NPACOLL=(YES,EXTENDED)
*
C21TR88  LINE  ADDRESS=(1088,FULL),TRSPEED=16,PORTADD=88,               X
               LOCADD=400037450001,NPACOLL=YES
C21PU88A PU
*
C21TR89  LINE  ADDRESS=(1089,FULL),TRSPEED=16,PORTADD=89,               X
               LOCADD=400037450002,NPACOLL=YES
C21PU89A PU
```

C21NCP1 – NTRI Logical Lines – INN and BNN

```
*********************************************************************
* LOGICAL BNN Lines                                                 *
*********************************************************************
*
C21BNNG1 GROUP ECLTYPE=LOGICAL,ANS=CONTINUE,AUTOGEN=250,CALL=INOUT,    X
               ISTATUS=ACTIVE,PHYSRSC=C21PU89A,                        X
               USSTAB=AUSSTAB,RETRIES=(10,10,10,20),XMITDLY=NONE,      X
               MODETAB=AMODETAB,NPACOLL=YES

*********************************************************************
*     NTRI INN LOGICAL LINES FOR TOKEN RING PORT 1088               *
*********************************************************************
*
C21INNG1 GROUP ECLTYPE=(LOGICAL,SUBAREA),ANS=CONT,PHYSRSC=C21PU88A,    X
               LOCALTO=13.5,REMOTTO=18.2,T2TIMER=(0.2,0.2,3),          X
               ISTATUS=ACTIVE,SDLCST=(C21PRI,C21SEC),NPACOLL=YES,      X
               MONLINK=CONT
*
C21LG2A  LINE  TGN=1,TGCONF=SINGLE
C21PG2A  PU    ADDR=184209374501D1,SSAP=(04,H)
```

Starting CCL from Linux – CCLV1R1

- From the Linux console, change to the CCL directory:
  - cd /opt/ibm/Communication_Controller_for_Linux/
- Load the CCL kernel module
  - ./load_ndh.sh
    - You will receive the message :

      NDH kernel modules loaded. You are now able to run the cclengine
- Start the CCL engine
  - nohup ./cclengine –mC21NCP1 -p2021 SVTC21 &
    - If you use telnet or ssh into the Linux host you will want to preface the command with “nohup” so that the process will remain active even after the telnet/ssh session is terminated.

Starting CCL from Linux – CCLV1R2

- From the Linux console, change to the CCL directory:
  - cd /opt/ibm/ndh
- Load the CCL kernel module
  - ./load_ndh.sh
    - You will receive the message :
      NDH kernel modules loaded. You are now able to run the cclengine
- From the Linux console, change to the CCL directory:
  - cd /opt/ibm/Communication_Controller_for_Linux/
- Start the CCL engine
  - nohup ./cclengine –mC21NCP1 -p2021 SVTC21 &
    - If you use telnet or ssh into the Linux host you will want to preface the command with “nohup” so that the process will remain active even after the telnet/ssh session is terminated.

Activating NCP using XCA from NETC.C02N

- **From NETC.C02N activate the XCA major node**

```
V NET,ACT,ALL,ID=C02XCA
IST097I VARY ACCEPTED
IST093I C02XCA ACTIVE
IST464I LINK STATION C02ETHP2 HAS CONTACTED C21NCP1 SA 21
IST093I C02ETHP2 ACTIVE
```

- **From NETC.C02N activate the NCP**

```
V NET,ACT,ID=C21NCP1,ALL
IST097I VARY ACCEPTED
IST093I C21NCP1 ACTIVE
IST093I C21PU88A ACTIVE
IST093I C21PU89A ACTIVE
IST093I C21NPPU ACTIVE
IST464I LINK STATION C21PG2A HAS CONTACTED C02NPU SA 2
IST093I C21PG2A ACTIVE
```

Reference Documentation

Networking Overview for Linux on zSeries

- http://www.redbooks.ibm.com/redpapers/pdfs/redp3901.pdf