March 2012

Early Experiences with z/VM 6.2 and Live Guest Relocation: moving from CSE to SSI

R. J. Brenneman
rjbrenn@us.ibm.com
IBM Platform Evaluation Test Laboratory
Poughkeepsie, NY

Table of Contents

Introduction

This paper describes the installation of z/VM® 6.2 and initial configuration of the SSI cluster, particularly in terms of how it is different from the familiar old z/VM installation process. The paper then compares the function offered by the older CSE technology to the new SSI technology, and describes the migration we undertook to move our systems from the old environment to the new one. Lastly, the paper covers the changes we made to our environment to take advantage of Live Guest Relocation (LGR).

This paper presumes that the reader is already familiar with the standard z/VM installation process used in previous releases, as well as the new function offered by z/VM 6.2. If not, http://www.vm.**ibm.com**/zvm620 and http://www.vm.**ibm.com**/ssi are an excellent place to start.

This paper is also not meant to replace the z/VM 6.2 product documentation! The z/VM 6.2 installation process is fully documented in the *z/VM 6.2 Installation Guide*, which is available at http://publib.boulder.**ibm.com**/infocenter/zvm/v6r2/topic/com.ibm.zvm.v620.zvminst/zvminst.htm. This paper provides an example of installing and configuring z/VM 6.2 in an existing z/VM production-like environment.

Installing z/VM 6.2

We are installing our new z/VM 6.2 SSI environment on top of an existing z/VM 6.1 and 5.4 CSE cluster. Because we already have a z/VM environment, we took advantage of the ability to install a new z/VM release as a guest of the current production z/VM environment.

We also decided to install a full four member SSI cluster during the z/VM installation process, rather than install a single SSI member and then clone it and configure more members later. We actually have seven members in our CSE cluster, but because SSI is limited to four members in the z/VM 6.2 release, we will configure a four member cluster right from the start. Like z/VM 6.1, the new z/VM 6.2 release requires an IBM System z10® or later processor, so we could not replace our three z/VM 5.4 LPARs, which are running on older processors anyway.

Our four member SSI cluster will have one LPAR running on each of an IBM System z10 Business Class™ (z10 BC™), an IBM System z10 Enterprise Class (z10 EC™), an IBM zEnterprise™ 196 (z196), and an IBM zEnterprise 114 (z114) processor in our test lab. The z/VM 6.2 installation process for a four member SSI cluster running on IBM 3390 mod9 DASD requires 14 devices. The z/VM 6.2 installation process all runs from a single z/VM guest, which has all 14 3390 mod9 devices linked or dedicated to it. SSI also requires full CTC connectivity between all the z/VM LPARs, but that connectivity is not required during the installation process. We already have this connectivity in our environment because we are using the CTCs to support our current CSE cluster.

Pre-installation planning and configuration

For our installation we created four new z/VM guests, one to run on each z/VM 6.1 CSE LPAR that would be migrated to the z/VM 6.2 release. Our first level z/VM 6.1 CSE systems were named: LTICVM2, LTICVM4, LTICVM5, and LTICVM9. We created guests named: 2LVLVM2, 2LVLVM4, 2LVLVM5, and 2LVLVM9, as well as LVL2VM. Each 2LVLVMx guest runs on the corresponding LTICVMx LPAR, which will provide the ability to test the cluster across multiple CPCs before putting the new z/VM 6.2 systems into production.

Step 1. The LVL2VM guest simply owns all 14 of the 3390 mod9 DASD devices for the new SSI cluster. The devices are defined as DEVNO minidisks in the LVL2VM directory as mode MWV devices, to enable virtual Reserve/Release push down and reserve the real device.

```
USER LVL2VM NOLOG
   MDISK 0191 3390 1141 5 VMPU01 MR
   MDISK 8290 3390 DEVNO 8290 MWV
   MDISK 8291 3390 DEVNO 8291 MWV
   MDISK 8292 3390 DEVNO 8292 MWV
   MDISK 8293 3390 DEVNO 8293 MWV
   MDISK 8294 3390 DEVNO 8294 MWV
   MDISK 8295 3390 DEVNO 8295 MWV
   MDISK 8296 3390 DEVNO 8296 MWV
   MDISK 8297 3390 DEVNO 8297 MWV
   MDISK 8298 3390 DEVNO 8298 MWV
   MDISK 838B 3390 DEVNO 838B MWV
   MDISK 838C 3390 DEVNO 838C MWV
   MDISK 838D 3390 DEVNO 838D MWV
   MDISK 838E 3390 DEVNO 838E MWV
   MDISK 838F 3390 DEVNO 838F MWV
```

Step 2. The 2LVLVMx guests in turn link the LVL2VM minidisks and dedicate a set of CTC devices that provide connectivity to the other LPARs. Each 2LVLVMx system has a 191 disk, and the guest for LTICVM9 where we will run the actual installation also has class B authority and another disk at address 2CF0. This disk is precisely 120 cylinders, as stated in the z/VM 6.2 installation planning documentation.

```
USER 2LVLVM2 2M4LKNHS 512M 1024M G
   INCLUDE IBMDFLT
   IPL CMS PARM AUTOCR
   MACHINE ESA
   LINK LVL2VM 8290 8290 MW
```

```
   LINK LVL2VM 8291 8291 MW
   LINK LVL2VM 8292 8292 MW
   LINK LVL2VM 8293 8293 MW
   LINK LVL2VM 8294 8294 MW
   LINK LVL2VM 8295 8295 MW
   LINK LVL2VM 8296 8296 MW
   LINK LVL2VM 8297 8297 MW
   LINK LVL2VM 8298 8298 MW
   LINK LVL2VM 838B 838B MW
   LINK LVL2VM 838C 838C MW
   LINK LVL2VM 838D 838D MW
   LINK LVL2VM 838E 838E MW
   LINK LVL2VM 838F 838F MW
   DEDICATE FA40 FA47
   DEDICATE FA48 FA4F
   DEDICATE F210 F217
   DEDICATE F218 F21F
   DEDICATE F310 F317
   DEDICATE F318 F31F
   MDISK 0191 3390 901 5 VMPU01 MR

USER 2LVLVM9 OQ9LNF2E 512M 1024M BG
   INCLUDE IBMDFLT
   IPL CMS PARM AUTOCR
   MACHINE ESA
   LINK LVL2VM 8290 8290 MW
   LINK LVL2VM 8291 8291 MW
   LINK LVL2VM 8292 8292 MW
   LINK LVL2VM 8293 8293 MW
   LINK LVL2VM 8294 8294 MW
   LINK LVL2VM 8295 8295 MW
   LINK LVL2VM 8296 8296 MW
    LINK LVL2VM 8297 8297 MW
    LINK LVL2VM 8298 8298 MW
    LINK LVL2VM 838B 838B MW
    LINK LVL2VM 838C 838C MW
    LINK LVL2VM 838D 838D MW
    LINK LVL2VM 838E 838E MW
    LINK LVL2VM 838F 838F MW
    DEDICATE FB40 FB47
```

```
        DEDICATE FB48 FB4F
        DEDICATE F210 F217
        DEDICATE F218 F21F
        DEDICATE F310 F317
        DEDICATE F318 F31F
        MDISK 0191 3390 906 5 VMPU01 MR
        MDISK 2CF0 3390 2448 120 VMPU01 W

  USER 2LVLVM5 HTJI886U 512M 1024M G
     INCLUDE IBMDFLT
     IPL CMS PARM AUTOCR
     MACHINE ESA
     LINK LVL2VM 8290 8290 MW
     LINK LVL2VM 8291 8291 MW
     LINK LVL2VM 8292 8292 MW
     LINK LVL2VM 8293 8293 MW
     LINK LVL2VM 8294 8294 MW
     LINK LVL2VM 8295 8295 MW
     LINK LVL2VM 8296 8296 MW
     LINK LVL2VM 8297 8297 MW
     LINK LVL2VM 8298 8298 MW
     LINK LVL2VM 838B 838B MW
     LINK LVL2VM 838C 838C MW
     LINK LVL2VM 838D 838D MW
     LINK LVL2VM 838E 838E MW
     LINK LVL2VM 838F 838F MW
     DEDICATE FB40 FB47
     DEDICATE FB48 FB4F
     DEDICATE FA40 FA47
    DEDICATE FA48 FA4F
    DEDICATE F310 F317
    DEDICATE F318 F31F
    MDISK 0191 3390 911 5 VMPU01 MR

USER 2LVLVM4 DXU576OG 512M 1024M G
   INCLUDE IBMDFLT
   IPL CMS PARM AUTOCR
   MACHINE ESA
   LINK LVL2VM 8290 8290 MW
```

```
LINK LVL2VM 8291 8291 MW
LINK LVL2VM 8292 8292 MW
LINK LVL2VM 8293 8293 MW
LINK LVL2VM 8294 8294 MW
LINK LVL2VM 8295 8295 MW
LINK LVL2VM 8296 8296 MW
LINK LVL2VM 8297 8297 MW
LINK LVL2VM 8298 8298 MW
LINK LVL2VM 838B 838B MW
LINK LVL2VM 838C 838C MW
LINK LVL2VM 838D 838D MW
LINK LVL2VM 838E 838E MW
LINK LVL2VM 838F 838F MW
DEDICATE FB40 FB47
DEDICATE FB48 FB4F
DEDICATE FA40 FA47
DEDICATE FA48 FA4F
DEDICATE F210 F217
DEDICATE F218 F21F
MDISK 0191 3390 916 5 VMPU01 MR
```

We decided to run the installation process on guest 2LVLVM9 on system LTICVM9, because that system has access to stand alone 3590 tape drives.

Step 3. After logging on to 2LVLVM9, we made sure that all 14 3390 mod9 DASD devices were linked correctly in RW mode, and also that the 2CF0 and 191 minidisks, which were required by the second level z/VM installation process, were in RW mode as well. We used **q v dasd** to verify this.

```
q v dasd
DASD 0190 3390 VM9RS2 R/O        214 CYL ON DASD  8180 SUBCHANNEL = 001B
DASD 0191 3390 VMPU01 R/W          5 CYL ON DASD  8304 SUBCHANNEL = 0014
DASD 019D 3390 VM9RS2 R/O        292 CYL ON DASD  8180 SUBCHANNEL = 001C
DASD 019E 3390 VM9RS2 R/O        500 CYL ON DASD  8180 SUBCHANNEL = 001D
DASD 0401 3390 VM9RS2 R/O        292 CYL ON DASD  8180 SUBCHANNEL = 001F
DASD 0402 3390 VM9RS2 R/O        292 CYL ON DASD  8180 SUBCHANNEL = 001E
DASD 2CF0 3390 VMPU01 R/W        120 CYL ON DASD  8304 SUBCHANNEL = 0015
DASD 8290 3390 S1COM1 R/W      10017 CYL ON DASD  8290 SUBCHANNEL = 0006
DASD 8291 3390 620RL1 R/W      10017 CYL ON DASD  8291 SUBCHANNEL = 0007
DASD 8292 3390 VM2RES R/W      10017 CYL ON DASD  8292 SUBCHANNEL = 0008
DASD 8293 3390 VM2S01 R/W      10017 CYL ON DASD  8293 SUBCHANNEL = 0009
DASD 8294 3390 VM2PG1 R/W      10017 CYL ON DASD  8294 SUBCHANNEL = 000A
DASD 8295 3390 VM9RES R/W      10017 CYL ON DASD  8295 SUBCHANNEL = 000B
DASD 8296 3390 VM9S01 R/W      10017 CYL ON DASD  8296 SUBCHANNEL = 000C
DASD 8297 3390 VM9PG1 R/W      10017 CYL ON DASD  8297 SUBCHANNEL = 000D
DASD 8298 3390 VM5RES R/W      10017 CYL ON DASD  8298 SUBCHANNEL = 000E
DASD 838B 3390 VMDRES R/W      10017 CYL ON DASD  838B SUBCHANNEL = 000F
DASD 838C 3390 VMBORC R/W      10017 CYL ON DASD  838C SUBCHANNEL = 0010
DASD 838D 3390 VMCRES R/W      10017 CYL ON DASD  838D SUBCHANNEL = 0011
DASD 838E 3390 VMFOR7 R/W      10017 CYL ON DASD  838E SUBCHANNEL = 0012
DASD 838F 3390 VMEU07 R/W      10017 CYL ON DASD  838F SUBCHANNEL = 0013
Ready; T=0.01/0.01 10:48:59
```

Step 4. We also used **q privclas** to verify that the guest had the correct BG privilege classes, as required by the second level z/VM installation process.

```
q privclas
Privilege classes for user 2LVLVM9
        Currently: BG
        Directory: BG
Ready; T=0.01/0.01 10:49:01
```

Step 5. We then used **q v store** to verify that the 2LVLVM9 guest met the minimum 64 MB memory requirement for the second level z/VM installation.

```
q v store
STORAGE = 512M
Ready; T=0.01/0.01 10:51:42
```

Step 6. We also checked that the CMS A disk had sufficient space available to run the installation with **q disk a**.

```
q disk a
LABEL  VDEV M  STAT   CYL TYPE BLKSZ   FILES  BLKS USED-(%) BLKS LEFT  BLK TOTAL
VM9191 191  A   R/W     5 3390 4096        1          8-01        892        900
Ready; T=0.01/0.01 10:59:12
```

Step 7. Finally we verified that the tape devices where the install media were mounted were attached correctly using **q tape**.

```
q tape
TAPE 0522 ATTACHED TO 2LVLVM9  0181 R/W
TAPE 0523 ATTACHED TO 2LVLVM9  0182 R/W
Ready; T=0.01/0.01 11:01:16
```

Copying installation materials from tape

After verifying that the guest had all the necessary devices attached and met all the installation requirements, we started the actual installation process.

Step 1. First we rewound the install tape on the 181 drive and used vmfplc2 to skip forward 7 tape marks.

```
rew 181
Rewind complete
Ready; T=0.01/0.01 11:02:59
vmfplc2 fsf 7
Ready; T=0.01/0.01 11:03:39
```

Step 2. Then, we used vmfplc2 again to copy the installation files from the tape onto the 2LVLVM9 guest's 191 DASD, which is accessed as the CMS A disk.

```
vmfplc2 load * * A
 Loading ...
 DDR      MODULE   A2
 ICKDSF   MODULE   A2
 INSTPLAN EXEC     A2
 INSTPXED XEDIT    A2
 INSTPLAN HELPINST A1
 INSTIIS  EXEC     A2
 INSTIXED XEDIT    A2
 INSTIIS1 HELPINST A1
 INSTHXED XEDIT    A2
 $ITEMCK$ $TABLE$  A1
 $ITEMFB$ $TABLE$  A1
 $INST$   $FILE$   A1
 $IDISK$  $TABLE$  A1
 SYSTEMCK CONFIG   A1
 SYSTEMFB CONFIG   A1
 ZVM620   COPYRITE A1
 IDISK90  DIRECT   A1
 DIRECTXA MODULE   A2
 SYSTEMCS CONFIG   A1
 INSTPLN2 HELPINST A1
 INSTIIS2 HELPINST A1
 FORMSSI  MODULE   A2
 CPFMTXA  EXEC     A2
 DISCLAIM HELPINST A1
 INSTPLN2 EXEC     A2
 INSTDISC HELPINST A1
 SSIFTRNM EXEC     A2
 INSTALL  EXEC     A2
 $INSTALL $FILE$   A1
 INSTIPL  EXEC     A2
 DISKMAP  EXEC     A2
 CPFMTXA  MODULE   A2
 INSSMAPI HELPINST A1
End-of-file or end-of-tape
Ready; T=0.01/0.05 11:04:01
```

Step 3. We then rewound the installation tape again before continuing to the next phase.

```
rew 181
Rewind complete
Ready; T=0.01/0.01 11:04:16
```

Setting up the installation using the instplan command

Step 1. Now we started the installation process by running **instplan tape** to lay out our z/VM 6.2 SSI installation.

```
instplan tape
```

< CMS clears the console and goes into full screen mode >

```
                *** z/VM INSTALLATION PLANNING ***


Mark the product(s) selected to be installed into the filepool with an
"F" and those selected to be installed to minidisks with an "M"
M         VM              F         OSA             M        PERFTK
F         VMHCD           M         RACF            M         DIRM
M         RSCS            M         ICKDSF          M         TCPIP

 Select a System Default Language.
       X AMENG         _ UCENG         _ KANJI


 Select a System DASD model.
     _ 3390 Mod 3          X 3390 Mod 9


 Enter the name of common service filepool.
     Filepool Name:     S1SVPOOL


Select a System Type: Non-SSI or SSI (SSI requires the SSI feature)
_ Non-SSI Install:    System Name ________
X SSI Install:        Number of Members 4     SSI Cluster Name LTICS1
```

We chose to install most of the products onto minidisks, with the exception of OSA/SF and HCD, which we installed completely into the filepool. Anyone familiar with previous z/VM installation procedures will notice two differences on this screen.

The first and most obvious difference is the section at the bottom, where we can decide to install a stand alone z/VM 6.2 system, or perform an SSI Install. The Non-SSI install option installs z/VM without enabling the SSI feature yet the z/VM directory is still installed as SSI-Ready. This means that even users who will not be exploiting SSI initially will have to migrate their directory from their old z/VM environment to the supplied SSI-Ready z/VM 6.2 directory. There are procedures in the *CP Planning and Administration Guide* to convert a

z/VM 6.2 system from Non-SSI to SSI configuration. For the SSI Install option, it is perfectly acceptable to install only one member in a SSI cluster, if that is what makes sense for your situation. We chose to go ahead and install all four members that we knew we would be running.

The second and more subtle difference in that panel is immediately above the SSI / Non-SSI selection. The filepool that is named during the installation is part of the new z/VM Service process. Even in the case where a z/VM 6.2 system is installed in a Non-SSI configuration, all of the service disks and components for all of the z/VM products are placed into a filepool. This provides the ability to migrate to an SSI configuration seamlessly, as well as manage service in the SSI cluster.

This change obviously has a serious impact on established service procedures in facilities that have been running IBM's z/VM family of products for a long time, or are maintaining an extensive list of modifications to the basic z/VM system. Fortunately we do not have a very sophisticated service process in the test lab, so this change does not cause major problems for us.

Sep 2. After processing the input on the first panel with F5, the SSI License displays. After thoroughly reading it and careful consideration, we accept it with F5 again to get to the next panel.

```
            *** z/VM INSTALLATION PLANNING PANEL 2 ***

N   Would you like to have your system automatically configured to be
managed by the Unified Resource Manager or some other SMAPI client for
system management? (Y/N)

Keep The Following in Mind:

If you say YES, you should not attempt to manage your system in any
other way.


If you'd like to manage your own system, or use a purchased
external security manager or a purchased directory manager say NO
```

Step 3. This panel has a single question on it, and it is a *very* important one. If you answer YES to this question, the z/VM installation process will automatically configure the Systems Management API and a minimal directory manager to enable the z/VM system to be managed by the new IBM zEnterprise Unified Resource Manager function provided by the HMC.

This automatic configuration cannot be rolled back after the installation is complete, and it will prevent you from using a full directory management product or an external security manager. You will not be able to enable the full DirMaint™ product or RACF®, nor will you be able to install and use CA Technologies' TopSecret or VM:Secure products.

This option to automatically configure z/VM for Unified Resource Manager management is meant for sniff testing the Unified Resource Manager functionality, not for production use. Remember that IBM recommends in the strongest possible terms that production systems run an external security manager, and answering YES to this question prevents that.

Step 4. We run in a production-like fashion, so we reply No to this option and press F5 to continue to the next panel.

```
              *** z/VM INSTALLATION PLANNING PANEL 3 ***

SSI Cluster Name:   LTICS1
After installation is complete, the SSI cluster will be IPLed:
    _    First-Level
    X    Second-Level

    SSI Member Name(s):

    SLOT #     MEMBER NAME    IPL LPAR/USERID
    ======     ===========    ===============
      1         LTICVM2_        2LVLVM2_
      2         LTICVM9_        2LVMVM9_
      3         LTICVM5_        2LVLVM5_
      4         LTICVM4_        2LVLVM4_
```

Step 5. In this panel we choose the name of the SSI cluster, and indicate whether it will be running first or second level after installation completes. We chose Second-Level because we will be running each SSI member as a guest of the first level LPAR that it will be running in when we put SSI into production. The z/VM installation process will automatically attempt to configure virtual CTC connections between all four second level guests as if they would all be running under the same z/VM host LPAR. We will fix that faulty assumption after the installation completes.

If we had chosen First-Level instead, there would be an additional panel to fill in with the hardware addresses of all the real CTC devices that will provide connectivity between the LPARs. We will make these changes in the SYSTEM CONFIG file manually after the installation process is complete.

Step 6. We also name each of the SSI cluster members here. By default, z/VM 6.2 will use either the LPAR name or the guest USERID to determine which member is which.

NOTE: If you inspect the response for our member 2 system (LTICVM9 on 2LVLVM9) you will see that we actually made a typo here. We accidentally said we would be running the LTICVM9 system on a guest named 2LV***M***VM9 rather than 2LV***L***VM9. This mistake actually left our fully installed member 2 unable to IPL. On startup, the z/VM System would immediately abend with a message stating that the system was configured as an SSI cluster member but did not have a member name assigned.

The fix was to IPL another cluster member and fix the SYSTEM CONFIG file to have the right System_Identifier statements. Alternately, we could have also used the DEFINE MDISK command from an authorized user to access the parm disk from the first level system and fix the SYSTEM CONFIG statements that way.

Step 7. After double and triple checking the member names and LPAR or USERIDs, we continued with the F5 key, and a couple screens of summary information are presented.

```
HCPIPX8475I THE PRODUCTS YOU SELECTED TO LOAD TO MINIDISK ARE:
  VM PERFTK RACF DIRM RSCS ICKDSF TCPIP

  THE PRODUCTS YOU SELECTED TO LOAD TO SFS ARE:
  OSA VMHCD

  THE SYSTEM DEFAULT LANGUAGE SELECTED:
  AMENG

  THE COMMON SERVICE FILEPOOL NAME IS:
  S1SVPOOL

  THE INSTALL TYPE YOU SELECTED IS:
  SSI

  THE SSI CLUSTER NAME IS:
  LTICS1

  THE NUMBER OF MEMBERS IS:
  4
MEMBER NAME 1: LTICVM2      IPL LPAR/USERID 1: 2LVLVM2
MEMBER NAME 2: LTICVM9      IPL LPAR/USERID 2: 2LVMVM9
MEMBER NAME 3: LTICVM5      IPL LPAR/USERID 3: 2LVLVM5
```

```
 MEMBER NAME 4: LTICVM4      IPL LPAR/USERID 4: 2LVLVM4

AFTER INSTALLATION IS COMPLETE, MEMBERS WILL BE IPLed FROM:
 Second-Level

THE DASD TYPE YOU SELECTED TO LOAD ON IS:
 3390 Mod 9

THE VOLUMES NEEDED TO LOAD z/VM ARE:
 COMMON:  VMCOM1
 RELEASE: 620RL1
 MEMBER1: M01RES M01S01 M01P01
 MEMBER2: M02RES M02S01 M02P01
 MEMBER3: M03RES M03S01 M03P01
 MEMBER4: M04RES M04S01 M04P01

 DO YOU WANT TO CONTINUE? (Y/N)
```

On the summary screen, we can see why we need 14 volumes. Each member will have three volumes:

1. One IPL and system residence volume
2. One page volume
3. One spool volume.

The IPL volume will contain the IPL text and SAL, the CP Nucleus for that system, and any read write minidisks that the z/VM system and products require in the course of normal operation. Things such as the RACF audit logs, Operator's 191 disk, Maint's 191 disk, and generally any minidisk which is unique to a z/VM system and cannot be shared will be on the system residence volume. The page and spool volumes are the same as in previous z/VM releases, every z/VM system needs at least one of each. For our four members, this makes a total of 12 volumes.

The other two volumes are the Common Volume and the Release Volume. The Release Volume contains all the product code and test disks for this particular z/VM release. The Common Volume contains the SSI Cluster descriptor block, the shared CF0 parm disk where the SYSTEM CONFIG file resides, and the shared service filepool minidisks.

Step 8. Answering Y and pressing Enter bring up the DASD configuration panel.

```
          *** z/VM INSTALLATION VOLUME DEFINITION ***

  TYPE      LABEL       ADDRESS                        FORMAT (Y/N)
 =======  ========   =========                        ============
 COMMON     S1COM1       8290                               Y

 RELVOL     620RL1       8291


TYPE      LABEL       ADDRESS          TYPE      LABEL       ADDRESS
=======  ========   =========        =======  ========   =========
LTICVM2                               LTICVM9
RES        VM2RES       8292         RES        VM9RES       8295
SPOOL      VM2S01       8293         SPOOL      VM9S01       8296
PAGE       VM2P01       8294         PAGE       VM9P01       8297

LTICVM5                               LTICVM4
RES        VM5RES       8298         RES        VM4RES       838D
SPOOL      VM5S01       838B         SPOOL      VM4S01       838E
PAGE       VM5P01       838C         PAGE       VM4P01       838F
```

Step 9. Enter the device addresses of the DASD for each required volume. You can change the volume labels in this panel if the default volume labels do not match your installation's volume naming standard. Press F5 to continue.

```
HCPIIX8377R YOU HAVE SELECTED TO FORMAT YOUR DASD. DASD SELECTED ARE:
                S1COM1     8290
                620RL1     8291
                VM2RES     8292
                VM2S01     8293
                VM2P01     8294
                VM9RES     8295
                VM9S01     8296
                VM9P01     8297
                VM5RES     8298
```

```
                VM5S01     838B
                VM5P01     838C
                VM4RES     838D
                VM4S01     838E
                VM4P01     838F
HCPINP8392I INSTPLAN EXEC ENDED SUCCESSFULLY
Ready; T=0.07/0.08 11:16:46
```

After INSTPLAN finishes, we are ready to begin the actual installation. Up until this point no changes have been made to any of the DASD devices.

Performing the installation with the install command

Step 1. Run install to start formatting and labeling the DASD and installing the new z/VM code.

```
install
HCPIIS8381I CHECKING TAPE VOLUME NUMBER FOR DRIVE 181

CP SP CONS * RDR

CP SP CONS START
HCPIIS8490I NOW FORMATTING VOLUME 8290 (1 OF 14)
HCPIIS8490I NOW FORMATTING VOLUME 8291 (2 OF 14)
HCPIIS8490I NOW FORMATTING VOLUME 8292 (3 OF 14)
HCPIIS8490I NOW FORMATTING VOLUME 8293 (4 OF 14)
HCPIIS8490I NOW FORMATTING VOLUME 8294 (5 OF 14)
```

... lots and lots of output trimmed ...

```
HCPMLP8392I INSTALL EXEC ENDED SUCCESSFULLY
Ready; T=6.50/7.19 13:40:20
```

The install process proceeds normally as it reads data from the tape drives and writes the base z/VM system onto the DASD devices.

Automated Step 2. After the base z/VM system is installed for this member, the installation process proceeds to clone it to the other three members.

Automated Step 3. The installation process then IPLs each of the four cluster members on this virtual machine in sequence.

Automated Step 4. As each member is IPLed, the installation process automatically invokes PUT2PROD to copy the production code from the service file pool to the production minidisks on the member's residence volume.

When each member has been IPLed and the new code copied to production, the INSTALL exec ends with the last cluster member up and running on the virtual machine. In our case, we ended with LTICVM4's residence volume IPLed and running on the 2LVLVM9 guest, and the above “INSTALL EXEC ENDED SUCCESSFULLY” message was the last message on the guest's console.

Finalizing installation and first IPL of the new z/VM 6.2 systems

The final step in the install process is to update the SYSTEM CONFIG file to allow each member to be IPLed only in its respective guest virtual machine.

Step 1. We run instscid remove to put the default System_Identifier statements in place.

```
instscid remove
************************************************************
*     PROCESSING REMOVE FOR ALL MEMBERS
************************************************************
MSGPFX8404I SYSTEM CONFIG has been updated to allow all members
            to be IPL'ed ONLY from the LPAR/USERid
            defined for each member at install time.
MSGPFX8392I INSTSCID EXEC ENDED SUCCESSFULLY
MSGPFX8342E THE COMMAND ' PIPE CMS COPYFILE INSTSCID COPY C
      INSTALL $MSGLOG$ =
(APPEND ' FAILED WITH RC=28
Ready; T=0.01/0.01 14:27:51
```

The failure message we saw here indicated that the install message log was not successfully copied to another minidisk for safe keeping. We copied it to the guest's 191 disk manually.

Step 2. Next, we shut down the second level SSI member so that we could IPL the proper member on this virtual machine. Note the new SSI shutdown messages.

```
SHUTDOWN
14:28:41 HCPWRP963I SHUTDOWN STEP PLMLV - LEAVE THE SSI CLUSTER
14:28:41 HCPWRP963I SHUTDOWN STEP USOAC - JOURNAL USER TERMINATION
14:28:42 HCPWRP963I SHUTDOWN STEP MFRSD - TERMINATE HARDWARE LOADER
14:28:42 HCPWRP963I SHUTDOWN STEP APISD - TERMINATE OTHER PROCESSORS
14:28:43 HCPWRP963I SHUTDOWN STEP ENASD - DISABLE TERMINAL DEVICES
14:28:43 HCPWRP963I SHUTDOWN STEP PLMDN - CHANGE SSI STATE TO DOWN
14:28:44 HCPWRP963I SHUTDOWN STEP KCBSD - PERFORM ISFC SHUTDOWN TASKS
14:28:45 HCPWRP963I SHUTDOWN STEP ISHDN - SHUT DOWN I/O SUBSYSTEM
14:28:45 HCPWRP963I SHUTDOWN STEP TTRAL - TERMINATE CONCURRENT COPY
SESSIONS
14:28:46 HCPWRP963I SHUTDOWN STEP SGPST - STOP OTHER PROCESSORS
14:28:46 HCPWRP959I LTICVM4  SYSTEM TERMINATION IN PROGRESS ON 2011-11-
29
```

```
14:28:46 HCPWRP963I SHUTDOWN STEP TXTDS - TERMINATE DATA TRACES
14:28:47 HCPWRP963I SHUTDOWN STEP SVACV - ACTIVATE TERMINATION SAVE
AREAS
14:28:47 HCPWRP963I SHUTDOWN STEP CHMOF - DISABLE CHANNEL MEASUREMENT
14:28:48 HCPWRP963I SHUTDOWN STEP ISHDA - DISABLE ALL DEVICES
14:28:48 HCPWRP963I SHUTDOWN STEP CKPSH - TAKE A CHECKPOINT
14:28:49 HCPWRP963I SHUTDOWN STEP OPRCK - SAVE OPERATOR CONSOLE LIST
14:28:49 HCPWRP963I SHUTDOWN STEP MCWMD - DETERMINE MACHINE CHECK
STATUS
14:28:50 HCPWRP963I SHUTDOWN STEP SDVRS - RESET IBM DASD CU
CHARACTERISTICS
14:28:50 HCPWRP962I VM SHUTDOWN COMPLETED IN 11 SEC
14:28:50 HCPWRP963I SHUTDOWN STEP SVADV - DEACTIVATE TERMINATION SAVE
AREAS
14:28:51 HCPWRP961W SYSTEM SHUTDOWN COMPLETE FOR LTICVM4  ON 2011-11-29
HCPGIR450W CP entered; disabled wait PSW 00020000 00000000 00000000
00000961
i cms
z/VM V6.1.0    2011-09-23 13:04


DMSSTT002E File SYN SYNONYM A1 not found
Ready; T=0.01/0.01 14:29:19
```

Step 3. At this point, we discovered our error in the z/VM Installation Planning panel 2 as we attempted to IPL the LTICVM9 system on guest 2LVLVM9. Our error was relatively easy to fix by IPLing LTICVM2 on 2LVMVM2 and fixing the error in SYSTEM CONFIG on the CF0 parm disk.

If we had made an error on the member name field, it would have been easier to re-run the instplan and install steps over again rather than attempt to fix it, because the member name is used in more config files than the LPAR or guest name is.

Step 4. When we fixed the SYSTEM CONFIG errors, our z/VM IPL ran nicely, and generated a couple of new messages.

```
11:04:57 z/VM  V6 R2.0  SERVICE LEVEL 1101 (64-BIT)
11:04:57 SYSTEM NUCLEUS CREATED ON 2011-11-02 AT 18:54:31, LOADED FROM
VM9RES
11:04:57
11:04:57 ************************************************************
11:04:57 * LICENSED MATERIALS - PROPERTY OF IBM*
```

```
11:04:57 *
11:04:57 * 5741-A07 (C) COPYRIGHT IBM CORP. 1983, 2011. ALL RIGHTS
11:04:57 * RESERVED. US GOVERNMENT USERS RESTRICTED RIGHTS - USE,
11:04:57 * DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP SCHEDULE
11:04:57 * CONTRACT WITH IBM CORP.
11:04:57 *
11:04:57 * * TRADEMARK OF INTERNATIONAL BUSINESS MACHINES.
11:04:57 ************************************************************
11:04:57
11:04:57 ************************************************************
11:04:57 * IBM z/VM Single System Image Feature is enabled and active.
11:04:57 ************************************************************
11:04:57
11:04:57 HCPZCO6718I Using parm disk 1 on volume S1COM1 (device 8290).
11:04:57 HCPZCO6718I Parm disk resides on cylinders 1 through 120.
11:04:57
11:10:53 HCPMLM3016I Management by the Unified Resource Manager is not
available for this system.
11:10:53 Start ((Warm|Force|COLD|CLEAN) (DRain) (DIsable)  (NODIRect)
11:10:53       (NOAUTOlog)) or (SHUTDOWN)
11:10:56
11:10:56 NOW 11:10:56 EST TUESDAY 2011-12-13
11:10:56 Change TOD clock (Yes|No)
11:10:56
11:10:56 The directory on volume VM9RES at address 8295 has been
brought online.
11:10:57 HCPWRS2513I
11:10:57 HCPWRS2513I Spool files available       56
11:10:58 HCPWRS2512I Spooling initialization is complete.
11:10:58 DASD 8296 dump unit CP IPL pages 10200
11:10:58 HCPAAU2700I System gateway LTICVM9 identified.
11:10:59 HCPNET3010I Virtual machine network device configuration
changes are permitted
11:10:59 HCPPLM1697I The state of SSI system LTICVM9 has changed from
DOWN to JOINED
11:10:59 HCPPLM1698I The mode of the SSI cluster is STABLE
11:10:59 z/VM Version 6 Release 2.0, Service Level 1101 (64-bit),
11:10:59 built on IBM Virtualization Technology
11:10:59 There is no logmsg data
11:10:59 FILES:   NO RDR, 0004 PRT,   NO PUN
```

```
11:10:59 LOGON AT 11:10:59 EST TUESDAY 12/13/11

11:10:59 GRAF  0009 LOGON  AS  OPERATOR USERS = 1

11:10:59 HCPIOP952I 0512M system storage

11:10:59 FILES: 0000018 RDR, 0000017 PRT,      NO PUN

11:10:59 HCPCRC8082I Accounting records are accumulating for userid
DISKACNT.

11:10:59 XAUTOLOG EREP

11:10:59 Command accepted

11:10:59 XAUTOLOG DISKACNT

11:10:59 Command accepted

11:10:59 XAUTOLOG AUTOLOG1

11:10:59 AUTO LOGON  ***       DISKACNT USERS = 2     BY OPERATOR

11:10:59 AUTO LOGON  ***       EREP     USERS = 3     BY OPERATOR

11:10:59 HCPCLS6056I XAUTOLOG information for DISKACNT: The IPL command
is verified by the IPL command processor.

11:10:59 HCPCLS6056I XAUTOLOG information for EREP: The IPL command is
verified by the IPL command processor.

11:10:59 Command accepted

11:10:59 XAUTOLOG OPERSYMP

11:10:59 Command accepted

11:10:59 AUTO LOGON  ***       AUTOLOG1 USERS = 4     BY OPERATOR

11:10:59 AUTO LOGON  ***       OPERSYMP USERS = 5     BY OPERATOR

11:10:59 HCPCLS6056I XAUTOLOG information for AUTOLOG1: The IPL command
is verified by the IPL command processor.

11:10:59 HCPCLS6056I XAUTOLOG information for OPERSYMP: The IPL command
is verified by the IPL command processor.

11:10:59 * MSG FROM EREP: 0 RECORDING FILE(S), 0 RECORDS, A DISK 03 %
FULL

11:10:59 * MSG FROM OPERSYMP: 0 RECORDING FILE(S), 0 RECORDS, A DISK 01
% FULL

11:10:59 HCPCRC8064I Recording data retrieval has been started;
recording *SYMPT

OM for userid OPERSYMP.

11:10:59 HCPCRC8064I Recording data retrieval has been started;
recording *LOGRE

C for userid EREP.

11:10:59 AUTO LOGON  ***       VMSERVS  USERS = 6     BY AUTOLOG1

11:10:59 AUTO LOGON  ***       VMSERVU  USERS = 7     BY AUTOLOG1

11:10:59 AUTO LOGON  ***       VMSERVR  USERS = 8     BY AUTOLOG1

11:10:59 AUTO LOGON  ***       VMSERVP  USERS = 9     BY AUTOLOG1

11:10:59 * MSG FROM DISKACNT: 4 RECORDING FILE(S), 68 RECORDS, A DISK
07 % FULL
```

```
11:10:59 HCPCRC8064I Recording data retrieval has been started;
recording *ACCOU
NT for userid DISKACNT.
11:10:59 AUTO LOGON  ***       DTCVSW1  USERS = 10    BY AUTOLOG1
11:10:59 AUTO LOGON  ***       DTCVSW2  USERS = 11    BY AUTOLOG1
11:11:29 USER DSC   LOGOFF AS  AUTOLOG1 USERS = 10
11:11:29 AUTO LOGON  ***       DIRMAINT USERS = 11    BY AUTOLOG1
 DVHPRO2008I ROLE = DIRMAINT
 DVHSHU2194T Automatic shutdown/restart
 DVHSHU2194T initiated.  Machine= DIRMAINT,
 DVHSHU2194T caller= DVHBEGIN, reason= 2119
 DVHSHU2196I The failing command will be retried.
 DVHSHU2197I The DIRMAINT machine is attempting to
 DVHSHU2197I re-IPL and restart.
 DVHPRO2008I ROLE = DIRMAINT
 DVHSHU2194T Automatic shutdown/restart
 DVHSHU2194T initiated.  Machine= DIRMAINT,
 DVHSHU2194T caller= DVHBEGIN, reason= 2119
 DVHSHU2195I The failing command has been purged.
 DVHSHU2197I The DIRMAINT machine is attempting to
 DVHSHU2197I re-IPL and restart.
 DVHPRO2008I ROLE = DIRMAINT
 DVHSHU2194T Automatic shutdown/restart
 DVHSHU2194T initiated.  Machine= DIRMAINT,
 DVHSHU2194T caller= DVHBEGIN, reason= 2119
 DVHSHU2196I The failing command will be retried.
 DVHSHU2197I The DIRMAINT machine is attempting to
 DVHSHU2197I re-IPL and restart.
 DVHPRO2008I ROLE = DIRMAINT
 DVHSHU2194T Automatic shutdown/restart
 DVHSHU2194T initiated.  Machine= DIRMAINT,
 DVHSHU2194T caller= DVHBEGIN, reason= 2119
 DVHSHU2195I The failing command has been purged.
 DVHSHU2198A The DIRMAINT service machine is logging
 DVHSHU2198A off.
11:11:32 USER DSC   LOGOFF AS  DIRMAINT USERS = 10
```

The DirMaint failure messages are expected at this point, because we have not yet configured it. By default, z/VM 6.2 starts DirMaint using autolog1's profile exec, in case it is required to support Unified Resource Manager.

Our installation is pre-configured by the installer to run all the SSI cluster members on a single z/VM image with virtual CTCs connecting them. We are actually going to run each member on a different CPC as guests of our existing CSE clustered z/VM systems. Now we are ready to do a little SSI Cluster configuration.

SSI cluster configuration

Step 1. Log on to the new second level z/VM 6.2 system as MAINT. We see one of the next major changes from previous releases of z/VM and release 6.2:

```
**************************************************************
THE MAINT620 USER ID **MUST** BE USED INSTEAD OF MAINT
WHEN INSTALLING SERVICE.
**************************************************************
```

The new z/VM 6.2 release introduces new maintenance IDs that are specific to the z/VM release. The MAINT user still exists, but any z/VM SES work must be performed from the release specific ID. There is also a new ID named PMAINT, which owns the cluster minidisks on the common volume. One of the cluster minidisks is the CF0 disk, which contains the shared SYSTEM CONFIG file that all the SSI cluster members use at IPL time.

Step 2. We can link and access the pmaint cf0 disk normally, and view the SYSTEM CONFIG file that it contains. The following are the relevant sections that pertain to the new SSI functionality.

The default install includes a System_Identifier section like this:

```
/************************************************************/
/*                  System_Identifier Information           */
/************************************************************/
     System_Identifier LPAR 2LVLVM2 LTICVM2
     System_Identifier LPAR 2LVLVM9 LTICVM9
     System_Identifier LPAR 2LVLVM5 LTICVM5
     System_Identifier LPAR 2LVLVM4 LTICVM4

/*   System_Identifier * * LTICVM4   */
```

The System Identifier must be set correctly for every SSI member. Each member has a statement in the file, and additionally there is the commented out statement that was used by the installer during the cloning process. The **instscid remove** command, which we ran right after the install finished, edited this section to uncomment the block of specific System_Identifier statements, and commented out the generic statement that this system config is only used for LTICVM4.

This is also the section that was configured incorrectly when we made an error on the input in the installation planning panel. We will add more Identifiers so that our system can be IPLed either second level on the indicated IDs or first level on native LPARs.

Step 3. Our modified System_Identifier section looks like this:

```
System_Identifier LPAR VMLX01 LTICVM2
System_Identifier 2097 %4zzzz LTICVM9
System_Identifier 2098 %1zzzz LTICVM5
System_Identifier 2818 %8zzzz LTICVM4
System_Identifier LPAR 2LVLVM2 LTICVM2
System_Identifier LPAR 2LVLVM9 LTICVM9
System_Identifier LPAR 2LVLVM5 LTICVM5
System_Identifier LPAR 2LVLVM4 LTICVM4
System_Identifier_Default ZVMSYS
```

We added four more identifiers for our production LPARs, and a system identifier default statement so that we can write automation later to handle the situation if a z/VM system is IPLed in the wrong place. We removed the commented out statement because it is no longer useful.

Step 4. Next we have the SSI statement:

```
/*********************************************************************/
/*         SSI Statement required for VMSSI feature                  */
/*********************************************************************/
  SSI LTICS1  PDR_VOLUME S1COM1,
       SLOT 1 LTICVM2,
       SLOT 2 LTICVM9,
       SLOT 3 LTICVM5,
       SLOT 4 LTICVM4
```

The SSI statement declares what the cluster name is, which volume contains Persistent Data Record, and what the names and slot numbers are for each cluster member. We do not need to make any changes to the SSI statement.

Step 5. The Checkpoint and Warmstart section contains configuration information for each cluster member. Because the SYSTEM CONFIG file is shared by all SSI cluster members, it uses the <system>: prefix to limit the scope of each System Residence statement to the system that it applies to.

```
/*********************************************************************/
/*               Checkpoint and Warmstart Information                */
/*********************************************************************/
 LTICVM2:     System_Residence,
              Checkpoint  Volid VM2RES   From CYL 21  For 9 ,
              Warmstart   Volid VM2RES   From CYL 30  For 9
 LTICVM9:     System_Residence,
              Checkpoint  Volid VM9RES   From CYL 21  For 9 ,
              Warmstart   Volid VM9RES   From CYL 30  For 9
 LTICVM5:     System_Residence,
              Checkpoint  Volid VM5RES   From CYL 21  For 9 ,
              Warmstart   Volid VM5RES   From CYL 30  For 9
 LTICVM4:     System_Residence,
              Checkpoint  Volid VM4RES   From CYL 21  For 9 ,
              Warmstart   Volid VM4RES   From CYL 30  For 9
```

Step 6. The CP_Owned volume statements are also different. There are some nice comments in the file, which describe how the CP_Owned list should be managed for a SSI cluster.

```
    LTICVM2:    CP_Owned   Slot   1  VM2RES
    LTICVM9:    CP_Owned   Slot   1  VM9RES
    LTICVM5:    CP_Owned   Slot   1  VM5RES
    LTICVM4:    CP_Owned   Slot   1  VM4RES
                CP_Owned   Slot   5  S1COM1
                CP_Owned   Slot  10  VM2S01
                CP_Owned   Slot  11  VM9S01
                CP_Owned   Slot  12  VM5S01
                CP_Owned   Slot  13  VM4S01
```

```
LTICVM2:    BEGIN
            CP_Owned   Slot 255  VM2P01
LTICVM2:    END
LTICVM9:    BEGIN
            CP_Owned   Slot 255  VM9P01
LTICVM9:    END
LTICVM5:    BEGIN
            CP_Owned   Slot 255  VM5P01
LTICVM5:    END
LTICVM4:    BEGIN
            CP_Owned   Slot 255  VM4P01
LTICVM4:    END
```

Notice that there are four slot 1 definitions, one for each system, and four slot 255 definitions, one for each system. The System Residence volumes are unique for each cluster member, and are not shared at all. The Paging volumes are also unique for each cluster member. The Common volume and the Spool volumes *are* shared among all the cluster members however.

What this set of statements sets up is an easy to understand model for handling system DASD for the cluster members. The RES volume is always in slot 1, and is not shared. The Common volume is at slot 5, and is shared. The Spool volumes start in slot 10 and count up from there, and are shared across all cluster members. As systems require additional spool space, add them to the config starting at slot 14 and count *up* from there, and make sure that they are configured on every SSI member. The Page volumes start at the other end of the list, at slot 255, and count *down* towards the Spool volumes. The Page volumes are not shared, and we can see a new construct in the SYSTEM CONFIG file. In addition to the <system>: memonic, which limits a single statement to a particular system, z/VM 6.2 adds the <system>: BEGIN and END statements to limit a block of statements to a particular system.

Step 7. Further down in System Config, we find the ISLINK statements that set up the SSI CTC connectivity:

```
/*****************************************************************/
/*         Activate ISLINK statements                            */
/*****************************************************************/
  LTICVM2:     ACTIVATE ISLINK A2B1 A2B2  NODE LTICVM9
 LTICVM2:     ACTIVATE ISLINK A2C1 A2C2  NODE LTICVM5
 LTICVM2:     ACTIVATE ISLINK A2D1 A2D2  NODE LTICVM4
```

```
LTICVM9:     ACTIVATE ISLINK B2A1 B2A2  NODE LTICVM2
LTICVM9:     ACTIVATE ISLINK B2C1 B2C2  NODE LTICVM5
LTICVM9:     ACTIVATE ISLINK B2D1 B2D2  NODE LTICVM4
LTICVM5:     ACTIVATE ISLINK C2A1 C2A2  NODE LTICVM2
LTICVM5:     ACTIVATE ISLINK C2B1 C2B2  NODE LTICVM9
LTICVM5:     ACTIVATE ISLINK C2D1 C2D2  NODE LTICVM4
LTICVM4:     ACTIVATE ISLINK D2A1 D2A2  NODE LTICVM2
LTICVM4:     ACTIVATE ISLINK D2B1 D2B2  NODE LTICVM9
LTICVM4:     ACTIVATE ISLINK D2C1 D2C2  NODE LTICVM5
```

SSI requires full connectivity between each SSI member: every member must have a direct link to every other member. Our choice to run the SSI cluster second level after installation allowed the installer to pre-configure this section with the virtual CTC addresses we were told to configure on to our z/VM guests in the *z/VM 6.2 Installation Guide*. We are not going to run our entire SSI cluster on a single z/VM host, however, so we must change this section to use our real CTC addresses, which we have dedicated in our second level z/VM guest's directory entries.

Step 8. Our updated ISLINK section looks like this:

```
/*********************************************************************/
/*         Activate ISLINK statements                                */
/*********************************************************************/

LTICVM2:     ACTIVATE ISLINK FA40 FA48  NODE LTICVM9
LTICVM2:     ACTIVATE ISLINK F210 F218  NODE LTICVM5
LTICVM2:     ACTIVATE ISLINK F310 F318  NODE LTICVM4
LTICVM9:     ACTIVATE ISLINK FB40 FB48  NODE LTICVM2
LTICVM9:     ACTIVATE ISLINK F210 F218  NODE LTICVM5
LTICVM9:     ACTIVATE ISLINK F310 F318  NODE LTICVM4
LTICVM5:     ACTIVATE ISLINK FB40 FB48  NODE LTICVM2
LTICVM5:     ACTIVATE ISLINK FA40 FA48  NODE LTICVM9
LTICVM5:     ACTIVATE ISLINK F310 F318  NODE LTICVM4
LTICVM4:     ACTIVATE ISLINK FB40 FB48  NODE LTICVM2
LTICVM4:     ACTIVATE ISLINK FA40 FA48  NODE LTICVM9
LTICVM4:     ACTIVATE ISLINK F210 F218  NODE LTICVM5
```

Step 9. If you go back and look at the directory entries for the 2LVLVMx guests, you can see we dedicate real CTC devices with consistent virtual addresses so that when we move our z/VM members down to first level, they will be able to use the first level CTC devices.

Step 10. When we made the changes to the ISLINK and System Identifier sections, we shutdown and reIPL z/VM to pick up the ISLINK changes.

IPLing additional SSI cluster members

Now we can attempt to IPL another SSI cluster member on another CPC to get the rest of the cluster up and running.

```
14:50:14 z/VM  V6 R2.0  SERVICE LEVEL 1101 (64-BIT)
14:50:15 SYSTEM NUCLEUS CREATED ON 2011-11-02 AT 18:54:31, LOADED FROM
VM2RES
14:50:15
14:50:15 ************************************************************
14:50:15 * LICENSED MATERIALS - PROPERTY OF IBM*                      *
14:50:15 *                                                           *
14:50:15 * 5741-A07 (C) COPYRIGHT IBM CORP. 1983, 2011. ALL RIGHTS    *
14:50:15 * RESERVED. US GOVERNMENT USERS RESTRICTED RIGHTS - USE,     *
14:50:15 * DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP SCHEDULE   *
14:50:15 * CONTRACT WITH IBM CORP.                                    *
14:50:15 *                                                           *
14:50:15 * * TRADEMARK OF INTERNATIONAL BUSINESS MACHINES.            *
14:50:15 ************************************************************
14:50:15
14:50:15 ************************************************************
14:50:15 * IBM z/VM Single System Image Feature is enabled and active.
14:50:15 ************************************************************
14:50:15
 14:50:15 HCPZCO6718I Using parm disk 1 on volume S1COM1 (device 8290).
 14:50:15 HCPZCO6718I Parm disk resides on cylinders 1 through 120.
 14:50:15
14:50:33 HCPMLM3016I Management by the Unified Resource Manager is not
available for this system.
14:50:33 Start ((Warm|Force|COLD|CLEAN) (DRain) (DIsable)  (NODIRect)
14:50:33       (NOAUTOlog)) or (SHUTDOWN)
14:50:36
14:50:36 NOW 14:50:36 EST TUESDAY 2011-12-13
14:50:36 Change TOD clock (Yes|No)
14:50:36
14:50:36 The directory on volume VM2RES at address 8292 has been
brought online.
 14:50:38 HCPWRS2513I
 14:50:38 HCPWRS2513I Spool files available       40
 14:50:39 HCPWRS2512I Spooling initialization is complete.
14:50:39 DASD 8293 dump unit CP IPL pages 10189
14:50:39 HCPAAU2700I System gateway LTICVM2 identified.
14:50:39 HCPPLM1644I The following is the current status of the SSI
member
14:50:39 HCPPLM1644I systems according to the PDR:
```

```
14:50:39 SSI Name: LTICS1
14:50:39 SSI Persistent Data Record (PDR) device: S1COM1 on 8290
14:50:39 SLOT SYSTEMID STATE     CONNECT TYPE          HOPS
14:50:39    1 LTICVM2  Down      Local                   -
14:50:39    2 LTICVM9  Joined    Not connected           -
14:50:39    3 LTICVM5  Down      Not connected           -
14:50:39    4 LTICVM4  Down      Not connected           -
14:50:39 HCPPLM1669I Waiting for ISFC connectivity in order to join the
SSI cluster.
14:50:39 HCPFCA2706I Link LTICVM9 activated by user SYSTEM.
14:50:39 HCPKCL2714I Link device FA40 added to link LTICVM9.
14:50:39 HCPKCL2714I Link device FA48 added to link LTICVM9.
14:50:40 HCPALN2702I Link LTICVM9 came up.
14:50:40 HCPACQ2704I Node LTICVM9 added to collection.
14:50:40 HCPPLM1697I The state of SSI system LTICVM2 has changed from
DOWN to JOINING
14:50:40 HCPPLM1698I The mode of the SSI cluster is IN-FLUX
14:50:40 HCPXHC1147I Spool synchronization with member LTICVM9
initiated.
14:50:40 HCPPLM1697I The state of SSI system LTICVM2 has changed from
JOINING to JOINED
14:50:40 HCPPLM1698I The mode of the SSI cluster is IN-FLUX
14:50:40 HCPXHC1147I Spool synchronization with member LTICVM9
completed.
14:50:41 HCPNET3010I Virtual machine network device configuration
changes are permitted
14:50:41 HCPPLM1698I The mode of the SSI cluster is STABLE
14:50:41 z/VM Version 6 Release 2.0, Service Level 1101 (64-bit),
14:50:41 built on IBM Virtualization Technology
14:50:41 There is no logmsg data
14:50:41 FILES:   NO RDR, 0001 PRT,   NO PUN
14:50:41 LOGON AT 14:50:41 EST TUESDAY 12/13/11
14:50:41 GRAF  0009 LOGON  AS  OPERATOR USERS = 1
 14:50:41 HCPIIO6284I Device 0401 cannot be varied online due to
 conflicting device classification information.
 14:50:41 HCPIIO6284I Device 0402 cannot be varied online due to
 conflicting device classification information.
14:50:41 HCPIOP952I 0512M system storage
14:50:41 FILES: 0000009 RDR, 0000018 PRT,      NO PUN
14:50:41 HCPCRC8082I Accounting records are accumulating for userid
DISKACNT.
14:50:41 XAUTOLOG EREP
14:50:41 Command accepted
14:50:41 XAUTOLOG DISKACNT
14:50:41 Command accepted
```

```
14:50:41 XAUTOLOG AUTOLOG1
14:50:41 Command accepted
14:50:41 XAUTOLOG OPERSYMP
14:50:41 Command accepted
14:50:41 AUTO LOGON  ***       EREP     USERS = 2     BY OPERATOR
14:50:41 HCPCLS6056I XAUTOLOG information for EREP: The IPL command is
verified by the IPL command processor.
14:50:41 AUTO LOGON  ***       DISKACNT USERS = 3     BY OPERATOR
14:50:41 HCPCLS6056I XAUTOLOG information for DISKACNT: The IPL command
is verified by the IPL command processor.
14:50:41 AUTO LOGON  ***       AUTOLOG1 USERS = 4     BY OPERATOR
14:50:41 AUTO LOGON  ***       OPERSYMP USERS = 5     BY OPERATOR
14:50:41 HCPCLS6056I XAUTOLOG information for AUTOLOG1: The IPL command
is verified by the IPL command processor.
14:50:41 HCPCLS6056I XAUTOLOG information for OPERSYMP: The IPL command
is verified by the IPL command processor.
 14:50:41  * MSG FROM EREP: 0 RECORDING FILE(S), 0 RECORDS, A DISK 03 %
 FULL
 14:50:41  * MSG FROM OPERSYMP: 0 RECORDING FILE(S), 0 RECORDS, A DISK
 01 % FULL
14:50:41 HCPCRC8064I Recording data retrieval has been started;
recording *LOGREC for userid EREP.
14:50:41 HCPCRC8064I Recording data retrieval has been started;
recording *SYMPTOM for userid OPERSYMP.
 14:50:41  * MSG FROM DISKACNT: 7 RECORDING FILE(S), 179 RECORDS, A
 DISK 10 % FULL
14:50:41 HCPCRC8064I Recording data retrieval has been started;
recording *ACCOUNT for userid DISKACNT.
14:50:42 AUTO LOGON  ***       VMSERVS  USERS = 6     BY AUTOLOG1
14:50:42 AUTO LOGON  ***       VMSERVU  USERS = 7     BY AUTOLOG1
14:50:42 AUTO LOGON  ***       VMSERVR  USERS = 8     BY AUTOLOG1
14:50:42 AUTO LOGON  ***       DTCVSW1  USERS = 9     BY AUTOLOG1
14:50:42 AUTO LOGON  ***       DTCVSW2  USERS = 10    BY AUTOLOG1
14:51:12 USER DSC   LOGOFF AS  AUTOLOG1 USERS = 9
14:51:12 AUTO LOGON  ***       DIRMAINT USERS = 10    BY AUTOLOG1
 DVHPRO2008I ROLE = DIRMAINT
 DVHSHU2194T Automatic shutdown/restart
 DVHSHU2194T initiated.  Machine= DIRMAINT,
 DVHSHU2194T caller= DVHBEGIN, reason= 2119
 DVHSHU2196I The failing command will be retried.
 DVHSHU2198A The DIRMAINT service machine is logging
 DVHSHU2198A off.
14:51:12 USER DSC   LOGOFF AS  DIRMAINT USERS = 9
```

Note that as this member comes up, it joins the SSI cluster and the cluster transitions through several states to keep track of the joining process. The CTC connectivity must be working correctly to allow each member to IPL and join the cluster. If the CTC connections are not working, the z/VM system will wait forever for the communications to start up with the other cluster members. We have usually been able to fix a balky CTC device by issuing DEACT ISLINK <devno> and varying the devices offline at the remote end, then varying them back online and issuing ACT ISLINK <devno> again to get communications going.

At this point we have both SSI members up and running on two CPCs second level. We will start the other two members up and get all four running before proceeding with enabling DirMaint, configuring RACF, or any of our other customization steps. We do not discuss these steps in this paper, because they are fairly similar to in the process used in previous releases. The only exception is enabling and configuring RACF, but the configuration steps for RACF in a SSI cluster are documented in *RACF Security Server System Programmer's Guide* section 4.4.3 Sharing RACF Databases in a z/VM Single System Image Cluster.

CSE functions versus SSI functions

Now that we have an operational z/VM 6.2 SSI cluster, we can compare and contrast the function offered by SSI to the old clustering technology for z/VM 6.1 and older releases, called CSE.

CSE provided three basic clustering functions, named XLINK, XSPOOL, and XMSG.

CSE XLINK functionality

XLINK provided protection for minidisks, such that the RR, MR, MW minidisk access semantics were enforced across LPARs. If a guest had a RW link to a minidisk on LPAR 1 then a guest on LPAR 2 would not be able to get a RW link unless both guests accessed the minidisk as MW, just as if they were running on the same z/VM image. XLINK allowed guests to be defined on multiple z/VM systems, yet their minidisks would be protected from multiple users attempting to write to the same device.

CSE's XLINK support required that DASD devices that would be protected by XLINK have a CSE area formatted on them using the XLINK utility on MAINT's 193 disk. The CSE area could take up to 9 cylinders on a 3390 mod9 or larger device. This area had to be protected with a minidisk overlay definition in $ALLOC$, to prevent a minidisk from being placed over it.

CSE XSPOOL functionality

CSE's XSPOOL support provided the illusion that a user's spool files would be available on every z/VM system participating in the XSPOOL environment. Each z/VM system had all the spool volumes of all the z/VM systems in their CP_Owned list, and they would read spool files off whichever spool volume they happened to reside on.

The CSE XSPOOL support required a communications path between every system that participated in XSPOOL, and used the PVM product to provide this communications path. PVM in turn was given CTC devices to each of the other z/VM systems, and ran the CSECOM task to enable the z/VM systems to exchange spool descriptor files.

One side effect of XSPOOL was that a user configured to participate in XSPOOL was not able to log on to more than one z/VM system at a time. This required that certain users such as MAINT, TCPIP, and other service virtual machines be added to the XSPOOL exclusion list, so that they would be able to run simultaneously on all the CSE cluster members.

CSE XMSG functionality

Lastly, CSE's XMSG support enabled users to send messages across system boundaries using SMSG and similar commands. This provided the ability to expand automation across the cluster, and required the same PVM infrastructure as XSPOOL support. The requirement for PVM to support cluster communications in CSE is a problem for customers running z/VM on an IFL, because PVM requires a special bid process to be licensed on an IFL.

SSI functions equivalent to CSE functions

SSI provides functionality similar to XLINK, XSPOOL, XMSG, and more. Additionally, it does not require an external program such as PVM to enable any of these capabilities. The shared minidisk support does not require any control blocks on the DASD devices such as the CSE area. All the minidisk access controls are handled by the z/VM systems directly. The XSPOOL and XMSG equivalent function do not require PVM or any other licensed software, z/VM can do it all by itself.

Similar to CSE's XSPOOL support, SSI also limits a user to logging on to one cluster member at a time, and the user sees all their spool files no matter which z/VM system they were created on. The ability to log on to multiple cluster members at once for users such as MAINT or TCPIP is controlled through the users directory configuration, rather than as a byproduct of a System Config statement, however.

Furthermore, SSI offers the ability to relocate a guest from one z/VM system to another – we will cover that in the last section.

In practical terms, the only limitation of implementing SSI as compared to CSE is the hard limit of four cluster members. SSI does not support more than four members in the cluster, whereas CSE can support more than that limit. If one chose to implement only XLINK, as many as 56 systems could participate in the cluster.

On the other hand, SSI is more robust in its ability to recover from failures. SSI is more careful about how it joins new systems into the cluster, and when the cluster is unstable it prevents any new users from logging on, and prevents minidisk linking activity.

We have had several situations where a CSE cluster member was IPLing and a Linux® guest that was logged on by startup automation managed to start on multiple CSE cluster members at once, despite the XSPOOL protection. In these cases, the guest's disks had to be recovered from tape backup, because they were written to from two systems at once and corrupted.

Migrating from CSE to SSI

Our CSE configuration included seven members. Only four of which ran on processors that were capable of running z/VM 6.2 due to the Architecture Level Set of an z10™ or later processor. Our LTICVM2, LTICVM9, LTICVM5, and LTICVM4 systems were eligible to move up to z/VM 6.2 SSI.

Converting CSE System Config statements to SSI equivalents

Our CSE clustered systems all ran using the same set of SYSTEM CONFIG statements, which were copied to each CSE system's config file on the CF1 parm disk. The CSE config statements are included here:

```
XLINK_System_Include Slot 1 LTICVM1
XLINK_System_Include Slot 2 LTICVM2
XLINK_System_Include Slot 3 LTICVM9
XLINK_System_Include Slot 4 LTICVM4
XLINK_System_Include Slot 5 LTICVM5
XLINK_System_Include Slot 6 LTICVM6
XLINK_System_Include Slot 7 LTICVM7
XLINK_System_Exclude LVL2VM
```

This first block of statements for XLINK_System_Include defines how the CSE area on XLINK protected devices is laid out. There are seven slots defined in the CSE area, one for each system.

These config statements are replaced by the SSI statement, which defines the PDR area on the common volume in z/VM 6.2, which is already in our SSI System Config by default.

Following the XLINK_System_Include statements are the XLINK_Volume_Include statements:

```
XLINK_Volume_Include VMP*
XLINK_Volume_Include VMPP00 Cylinder 10008
XLINK_Volume_Include VMN* Cylinder 10008
XLINK_Volume_Include VMO* Cylinder 10008
XLINK_Volume_Include VMQ*
XLINK_Volume_Include DT* Cylinder 63430
XLINK_Volume_Include IT* Cylinder 63430
XLINK_Volume_Include GP* Cylinder 63430
```

The XLINK_Volume_Include statements declare which volumes will be protected by CSE XLINK. Any volume who's Volume Label matches the patterns listed here is protected if it has a CSE area at the indicated cylinder. SSI, on the other hand, does not require a CSE area or any special formatting at all on the protected user volumes. All volumes are protected; there is no config statement in SSI equivalent to XLINK_Volume_Include, so we do not need to copy these statements over to our SSI System Config either.

After the CSE XLINK statements we have our CSE XSPOOL configuration statements:

```
XSPOOL_SYSTEM Slot 1 LTICVM1 Share_Spool NO
XSPOOL_SYSTEM Slot 2 LTICVM2 Share_Spool NO
XSPOOL_SYSTEM Slot 3 LTICVM9 Share_Spool NO
XSPOOL_SYSTEM Slot 4 LTICVM4 Share_Spool NO
XSPOOL_SYSTEM Slot 5 LTICVM5 Share_Spool NO
XSPOOL_SYSTEM Slot 6 LTICVM6 Share_Spool NO
XSPOOL_SYSTEM Slot 7 LTICVM7 Share_Spool NO
```

The XSPOOL_SYSTEM definitions define slots for Cross System Spooling support in CSE and specifies whether XSPOOL is active for that system. We were not utilizing the shared spool support provided by XSPOOL, but we were using the byproduct of these definitions: which is that users are prevented from logging on to multiple systems at once.

The slots for XSPOOL here are also handled by the SSI PDR record definition in the SYSTEM CONFIG file, so there is no equivalent statement in z/VM 6.2 SSI. Cross System Spooling support in SSI is always on when the system is a member of a SSI cluster. There is no option to turn it off, as there is in CSE.

After the CSE XSPOOL statements are the XSPOOL Exclusion List statements:

< xlist trimmed for brevity>

```
XSPOOL_XLIST_OUTPUT  VSMSERVE OPMGRM OPERATOR TCPIP MAINT
XSPOOL_XLIST_INPUT   VSMSERVE OPMGRM OPERATOR TCPIP MAINT
```

Step 5. The XSPOOL_XLIST_OUTPUT and INPUT statements exclude users from the XSPOOL configuration. Because we were not actually sharing spool in our CSE configuration we were really using these statements to control which users were allowed to log on to multiple z/VM systems in the CSE cluster. In SSI, this control does not exist explicitly but the effect can be achieved by converting a User to an Identity. We discuss the User versus Identity differences when we discuss DIRECTORY statements in the text that follows.

To summarize, the XLINK and XSPOOL statements are not allowed or needed in a SSI configuration, but it is good practice to examine the current CSE configuration to make sure that the SSI defaults will not cause problems for you.

The rest of your System Config statements can be moved to the SSI system as is, such as VSWITCH configurations and MAC address prefix settings.

After you have all the SYSTEM CONFIG changes for CSE to SSI in place, it is time to move on to the directory.

Converting CSE DIRECTORY statements to SSI equivalents

In our environment, when we move guests from one z/VM system to another we simply copy the user directory entries over and make sure the required DASD is attached to the system. When moving from CSE to SSI, some more attention is required.

The first difference to be aware of is the DIRECTORY statement itself. CSE requires one DIRECTORY statement for each system in the CSE cluster:

```
*CSE
   DIRECTORY 0123 3390 VM7IPL 11zzzz-2094 LTICVM7
   DIRECTORY 0123 3390 VM2IPL *4zzzz-2094 LTICVM2
   DIRECTORY 0123 3390 VM9IPL *4zzzz-2097 LTICVM9
   DIRECTORY 0123 3390 VM5IPL *1zzzz-2098 LTICVM5
   DIRECTORY 0123 3390 VM4IPL *8zzzz-2098 LTICVM4
   DIRECTORY 0123 3390 VM1IPL *0zzzz-2084 LTICVM1
   DIRECTORY 0123 3390 VM6IPL 01zzzz-2094 LTICVM6
```

SSI uses only one DIRECTORY statement that covers every system in the SSI Cluster:

```
*SSI
   DIRECTORY SSI 123 3390 VM2RES VM9RES VM5RES VM4RES
```

Notice that the CSE version lists the processor Serial Number-Machine Type (SN and MT) and system name, yet SSI does not require that information. These statements are used by DIRECTXA when it processes the directory to build the object directory on the system residence volume. On CSE systems, DIRECTXA looks at the processor SN and MT to decide which node it is running on, and then processes the directory accordingly. On a SSI system DIRECTXA does not need to be told explicitly and can figure out what to do on its own.

CSE also provided the ability to adapt a user's directory definition to the specifics of the z/VM systems where it could run. In the following directory entry for the EREP user, we can see that it had two options for a 191 disk, depending on which system it was logged on at.

```
*CSE
USER EREP EREP 32M 32M BFG
   ACCOUNT EREP IBMCE
   IPL 190
   IUCV *LOGREC
```

```
   MACH ESA
   XAUTOLOG AUTOLOG1 OP1 MAINT
   CONSOLE 001F 3215
   SPOOL 000C 2540 READER A
   SPOOL 000D 2540 PUNCH B
   SPOOL 000E 1403 A
   LINK MAINT 0190 0190 RR
   LINK MAINT 0201 0201 RR
SYSAFFIN LTICVM1 LTICVM2 LTICVM6 LTICVM7
   MDISK 0191 3390 7959 6 +VMRES MR READ WRITE MULTIPLE
SYSAFFIN LTICVM4 LTICVM5 LTICVM9
   MDISK 191 3390 2396 6 +VMRES  MR READ     WRITE    MULTIPLE
```

If EREP was logged on to systems LTICVM1, LTICVM2, LTICVM6 or LTICVM7, it got a 6 cylinder 191 disk on the system IPL volume starting at cylinder 7959. On the other hand, if the EREP guest was logged on to systems LTICVM4, LTICVM5, or LTICVM9 it got a 6 cylinder 191 disk on the IPL volume starting at cylinder 2396. This ability could also be applied to a dedicated device such as a FCP subchannel or a dedicated crypto domain.

For our Linux virtual machines, we used the SYSAFFIN statements to provide the illusion of an identical IODF configuration as we logged users off one CSE member and on to another. A guest could count on the OSA device at virtual addresses 600-602 always being available, irrespective of whether real devices 1000-1002 were dedicated as 600-602 on CPC 1 and real devices 9A0-9A2 were dedicated as 600-602 on CPC 2.

Users with SYSAFFIN statements were prevented from logging on to multiple systems at once due to the XSPOOL_XLIST_ settings in System Config.

On z/VM 6.2, SYSAFFIN is not supported for systems running in a SSI cluster or with a SSI-Ready directory, which is the style of directory installed by default. Instead, users that require unique directory changes on specific cluster members can be defined as an Identity, rather than as user, as shown in the z/VM 6.2 default definition for EREP as shown here:

```
*SSI
IDENTITY EREP     EREP       32M   32M BFG
 BUILD ON LTICVM2 USING SUBCONFIG EREP-1
 BUILD ON LTICVM9 USING SUBCONFIG EREP-2
 BUILD ON LTICVM5 USING SUBCONFIG EREP-3
 BUILD ON LTICVM4 USING SUBCONFIG EREP-4
 AUTOLOG AUTOLOG1 OP1 MAINT
 ACCOUNT EREP IBMCE
 MACH ESA
 IPL 190
 IUCV  *LOGREC
 CONSOLE 01F 3215
 SPOOL 00C 2540 READER A
 SPOOL 00D 2540 PUNCH B
 SPOOL 00E 1403 A

SUBCONFIG EREP-1
 LINK MAINT 190 190 RR
 LINK MAINT 201 201 RR
 MDISK 191 3390 2616 002 VM2RES MR READ     WRITE    MULTIPLE

SUBCONFIG EREP-2
 LINK MAINT 190 190 RR
 LINK MAINT 201 201 RR
 MDISK 191 3390 2616 002 VM9RES MR READ     WRITE    MULTIPLE

SUBCONFIG EREP-3
 LINK MAINT 190 190 RR
 LINK MAINT 201 201 RR
 MDISK 191 3390 2616 002 VM5RES MR READ     WRITE    MULTIPLE

SUBCONFIG EREP-4
 LINK MAINT 190 190 RR
 LINK MAINT 201 201 RR
 MDISK 191 3390 2616 002 VM4RES MR READ     WRITE    MULTIPLE*
```

It is important to note that those are essentially five user definitions. One Identity definition for the EREP user, and four Subconfig definitions that contain the statements unique to each cluster member. The Identity definition contains all the common statements that apply to all SSI cluster members. It also contains the BUILD ON statements that specify which SUBCONFIG is used for each cluster member.

The Subconfig definitions contain the statements that are unique to each cluster member. Mostly this is used to place a minidisk on the system residence volume, such as the 191 disk in this example. The Subconfig also includes any Link statements pointing to minidisks contained in a Subconfig of the target user. For example, the 190 disk owned by MAINT is actually in MAINT's Subconfig, so the link statement here for EREP also must be in the Subconfig.

One practical difference between using SYSAFFIN statements with CSE and Identity / Subconfigs with SSI is that the SYSAFFIN allows unique per cluster member configurations, without altering shared spooling behavior or the ability to log on to multiple systems at once.

The Identity / Subconfig construct in SSI allows unique per cluster member configurations, but it also implies that an Identity can log on to every cluster member at the same time, and is also excluded from the shared spool environment. This means that some things will just not be possible in SSI that are possible in CSE.

For example, a dedicated crypto queue could be different on each LPAR, and a Linux guest might need access to a dedicated crypro queue no matter which LPAR it is running on. With CSE, simply wrapping the CRYPTO APDED statements in SYSAFFIN permits that guest to access the correct crypto queue on each LPAR. If the guest must be able to move around between LPARs, it can access the queue while being prevented from logging on to more than one system at a time.

On SSI however, the unique LPAR specific statements must be placed into Subconfigs, and the Linux guest defined as an Identity user. The problem here is that the Linux system is not prevented from logging on to multiple systems at one time, because it is an Identity. The SSI minidisk protection mechanism would prevent the guest from obtaining multiple write links as long as the minidisk definitions are sensible and do not specify something such as MW. However, other havoc could be caused by the guest logging on to multiple systems simultaneously, such as disrupting system automation or the network if it is able to start a network interface.

Luckily, in our situation the only real directory configuration problem we had was with Linux guests that had dedicated crypto queues. Dedicated crypto queues also prevent a guest from being relocated with Live Guest Relocation, so our guests were essentially pinned to the one LPAR, where they usually ran anyway. We decided that some of the guests that were using dedicated crypto queues did not technically need them, and they were converted to use shared crypto queues with the CRYPTO APVIRT statement.

The remaining guests that truly required dedicated crypto queues were already members of a software High Availability cluster for whatever software required the crypto support. We therefore chose to do without Live Guest Relocation and the ability to run those guests on multiple LPARs, because there were other HA cluster members available to continue providing service in the event that these guests were not available.

To summarize, converting our directory from CSE to SSI format was fairly straightforward. We took the CSE directory and copied all the users that did not have SYSAFFIN statements into the new SSI cluster directory. Then we inspected the users with SYSAFFIN statements that were left over and revisited the decisions that lead to us using SYSAFFIN in the first place. In many instances, the process of setting up the user for LGR as described in the following section addressed the reasons why we were using SYSAFFIN.

Live Guest Relocation requirements and early experiences

In order to relocate a guest from one z/VM 6.2 system to another, the CP component must be able to determine which devices on the target system are equivalent to devices on the source system. For DASD devices, this is relatively easy. Each DASD volume has a label that is unique to the volume, as well as a unique identifier provided by the subsystem. Other types of devices, such as OSA and FCP, are not as straightforward. OSA and FCP devices with the same device numbers might not necessarily connect to the same networks.

SSI LGR Equivalence IDs

One of the new features in z/VM 6.2 is the concept of an Equivalence ID (EQID). Two devices labeled with the same Equivalence ID (EQID) means that "This Device over here and That Device over there get you to the same place." This is important when relocating a guest with active network sessions and some disk I/O in progress. By default, z/VM 6.2 will assign an EQID to every device on the system at IPL time, but the only EQIDs that are truly equivalent across LPARs by default are on the DASD devices.

For OSA, FCP, and HiperSockets™ devices, the Systems Programmer must explicitly assign EQIDs in the SYSTEM CONFIG file on PMAINT's CF0 disk.

Here is a section of our EQID statements:

```
/*********************************************************************/
/* Define EQID Classes                                               */
/*********************************************************************/
  LTICVM2: BEGIN
           RDEV 09E0-09E2 EQID 9DOTLAN TYPE OSA
           RDEV C100-C102 EQID TRUNK1  TYPE OSA
           RDEV C200-C202 EQID TRUNK1  TYPE OSA
           RDEV C300-C302 EQID TRUNK2  TYPE OSA
           RDEV C400-C402 EQID TRUNK2  TYPE OSA
           RDEV C508-C50A EQID HEARTBT TYPE OSA
           RDEV 0608-060A EQID HEARTBT TYPE OSA
           RDEV C50C-C50E EQID ADMIN   TYPE OSA
           RDEV 060C-060E EQID ADMIN   TYPE OSA
           RDEV 0100      EQID F0X00   TYPE FCP
           RDEV 0101      EQID F0X01   TYPE FCP
           RDEV 0102      EQID F0X02   TYPE FCP
           RDEV 0103      EQID F0X03   TYPE FCP
           RDEV 0C00      EQID F0Y00   TYPE FCP
           RDEV 0C01      EQID F0Y01   TYPE FCP
           RDEV 0C02      EQID F0Y02   TYPE FCP
           RDEV 0C03      EQID F0Y03   TYPE FCP
           RDEV 0300      EQID F1X00   TYPE FCP
           RDEV 0301      EQID F1X01   TYPE FCP
           RDEV 0302      EQID F1X02   TYPE FCP
           RDEV 0303      EQID F1X03   TYPE FCP
           RDEV 0400      EQID F1Y00   TYPE FCP
           RDEV 0401      EQID F1Y01   TYPE FCP
           RDEV 0402      EQID F1Y02   TYPE FCP
           RDEV 0403      EQID F1Y03   TYPE FCP
           RDEV E100-E101 EQID HSI10   TYPE HIPERS
           RDEV E116      EQID HSI10   TYPE HIPERS
           RDEV E102-E103 EQID HSI11   TYPE HIPERS
           RDEV E117      EQID HSI11   TYPE HIPERS
           RDEV E104-E105 EQID HSI12   TYPE HIPERS
           RDEV E118      EQID HSI12   TYPE HIPERS
  LTICVM2: END
  LTICVM9: BEGIN
           RDEV 06D4-06D6 EQID 9DOTLAN TYPE OSA
           RDEV 09C0-09C2 EQID TRUNK1  TYPE OSA
              RDEV 09E0-09E2 EQID TRUNK2  TYPE OSA
              RDEV 09CC-09CE EQID HEARTBT TYPE OSA
              RDEV 09EC-09EE EQID HEARTBT TYPE OSA
```

```
            RDEV 09C4-09C6 EQID ADMIN   TYPE OSA
            RDEV 09E4-09E6 EQID ADMIN   TYPE OSA
            RDEV 0100      EQID F0X00   TYPE FCP
            RDEV 0101      EQID F0X01   TYPE FCP
            RDEV 0102      EQID F0X02   TYPE FCP
            RDEV 0103      EQID F0X03   TYPE FCP
            RDEV 0200      EQID F0Y00   TYPE FCP
            RDEV 0201      EQID F0Y01   TYPE FCP
            RDEV 0202      EQID F0Y02   TYPE FCP
            RDEV 0203      EQID F0Y03   TYPE FCP
            RDEV 0400      EQID F1X00   TYPE FCP
            RDEV 0401      EQID F1X01   TYPE FCP
            RDEV 0402      EQID F1X02   TYPE FCP
            RDEV 0403      EQID F1X03   TYPE FCP
            RDEV 0A00      EQID F1Y00   TYPE FCP
            RDEV 0A01      EQID F1Y01   TYPE FCP
            RDEV 0A02      EQID F1Y02   TYPE FCP
            RDEV 0A03      EQID F1Y03   TYPE FCP
            RDEV E100-E101 EQID HSI10   TYPE HIPERS
            RDEV E116      EQID HSI10   TYPE HIPERS
            RDEV E102-E103 EQID HSI11   TYPE HIPERS
            RDEV E117      EQID HSI11   TYPE HIPERS
            RDEV E104-E105 EQID HSI12   TYPE HIPERS
            RDEV E118      EQID HSI12   TYPE HIPERS
LTICVM9: END
```

All the OSA devices are assigned an EQID that matches the name of the VSWITCH supported by that OSA. For example, the OSA devices with EQID TRUNK1 are all used by a VSWITCH named TRUNK1. In network terms, equivalent OSA devices are connected to the same Layer 2 broadcast domain. Two equivalent devices must be able to reach each other directly over the network, with no routers between them.

The FCP devices each get unique EQIDs – no two FCP devices have the same EQID within an LPAR. We are using NPIV in our SAN environment, so each FCP device really is unique, and they cannot be interchanged without careful planning. Our SAN zones contain both FCP devices that have the same EQID. For example, a guest that uses device 0C00 on system LTICVM2 for SAN access has a zone in the SAN switch with the world wide port name of both device 0C00 from LTICVM2 and device 0200 from LTICVM9 in it, in addition to the target disk enclosure world wide port names. In SAN terms, equivalent FCP devices are either separately zoned to the same target devices, or are in the same zone together with the target devices.

The HiperSockets devices are a special case. By definition, a HiperSockets device on CPC 1 cannot be equivalent to a HiperSockets device on CPC 2. HiperSockets do not span to different CPCs. It is impossible to ping a guest on one CPC from another CPC through HiperSockets without running the HiperSockets concentrator, or a software bridge connecting the two HiperSockets through real OSA devices. We do not bridge our HiperSockets together in this manner in our test environment, so by the strictest definition of equivalence we cannot say that our HiperSockets devices are equivalent between CPCs.

That said, it is possible to declare that one HiperSockets device is equivalent to another using EQIDs, but this requires very careful thought and planning, as well as some additional technology (such as OSPF dynamic routing) to make the combination of Live Guest Relocation and active HiperSockets utilization feasible. See the companion paper *z/VM Single System Image Cluster Live Guest Relocation with HiperSockets Exploitation via Dynamic Routing* for more details on this combination.

We do not have any guests using dedicated OSA devices on these z/VM systems, but if we did we would have labeled a set of devices for that guest to use similarly to how we have the HiperSockets devices labeled here.

With the above EQIDs in the SYSTEM CONFIG file, we have a sound I/O foundation that we can use as we start moving guests around the cluster.

SSI LGR Directory requirements

In order to exploit LGR our guests required a few additional z/VM directory changes in addition to what was required to move from CSE to SSI.

What follows is an example based on one of our WebSphere Application Servers and its profile. It has two 3390 minidisks as well as four dedicated FCP devices.

Before converting from our CSE configuration, we were using SYSAFFIN statements to attach the correct FCP devices on each CPC for cases when we had to manually shut the guest down and move it to the other LPAR.

```
*CSE
USER LITSWAS1 XXXXXXXX 1536M 1536M GZ
   INCLUDE LITPRO
   CPU 0
   CPU 1
   IPL CMS PARM AUTOCR
   MACHINE ESA 2
SYSAFFIN LTICVM9
   DEDICATE A000 0102
   DEDICATE A100 0202
   DEDICATE A200 0402
   DEDICATE A300 0A02
SYSAFFIN LTICVM2
   DEDICATE A000 0102
   DEDICATE A100 0C02
   DEDICATE A200 0302
   DEDICATE A300 0402
SYSAFFIN *
   MDISK 0200 FB-512 V-DISK 2048000 MR
   MDISK 0201 3390 56679 6678 IT0054 MR
   MDISK 0202 3390 23798 6678 IT0018 MR
*
   PROFILE LITPRO
   IUCV ALLOW
   OPTION APPLMON
   CRYPTO    APVIRT
   CONSOLE 0009 3215 T OPMGRM
   NICDEF 0F00 TYPE QDIO LAN SYSTEM TRUNK1
   NICDEF 0F10 TYPE QDIO LAN SYSTEM TRUNK2
   NICDEF 0F20 TYPE QDIO LAN SYSTEM ADMIN
   NICDEF 0F30 TYPE QDIO LAN SYSTEM HEARTBT
   SPOOL 000C 2540 READER
   SPOOL 000D 2540 PUNCH
   SPOOL 000E 1403 A
   LINK MAINT 0190 0190 RR
   LINK MAINT 019D 019D RR
   LINK MAINT 019E 019E RR
   LINK USER 0196 0196 RR
   LINK LITMSTR 0191 0191 RR
```

We were already using NPIV, and had already set up zones in the SAN so that whether this guest was running on LTICVM2 or LTICVM9, it would be able to reach its SAN disks. We had also already set up the disk subsystem to enable the guest to reach its LUNs from either CPC. It is important to note here that this access control had to be done in two places: both the SAN zoning in the switches, and the LUN masking on the disk subsystem.

After converting to z/VM 6.2, here is the SSI version of that user's directory entry, along with the required changes for LGR:

```
*SSI
USER LITSWAS1 LITSWAS1 1536M 1536M GZ
   INCLUDE LITPRO
   CPU 0
   CPU 1
   IPL CMS PARM AUTOCR
   MACHINE ESA 2
   DEDICATE A000 0102
   DEDICATE A100 0202
   DEDICATE A200 0402
   DEDICATE A300 0A02
   MDISK 0201 3390 56679 6678 IT0054 MR
   MDISK 0202 3390 23798 6678 IT0018 MR

PROFILE LITPRO
  CRYPTO    APVIRT
  IUCV ALLOW
  OPTION APPLMON CHPIDVIRTUALIZATION ONE
  CONSOLE 0009 3215 T OPMGRM
  NICDEF 0F00 TYPE QDIO LAN SYSTEM TRUNK1
  NICDEF 0F10 TYPE QDIO LAN SYSTEM TRUNK2
  NICDEF 0F20 TYPE QDIO LAN SYSTEM ADMIN
  NICDEF 0F30 TYPE QDIO LAN SYSTEM HEARTBT
  SPOOL 000C 2540 READER
  SPOOL 000D 2540 PUNCH
  SPOOL 000E 1403 A
  LINK MAINT 0190 0190 RR
  LINK MAINT 019D 019D RR
  LINK MAINT 019E 019E RR
  LINK USER 0196 0196 RR
  LINK LITMSTR 0191 0191 RR
```

There are two significant changes to note here.

The first change is that the SYSAFFIN statements and the set of DEDICATE statements for LTICVM2 are gone from the user directory entry. This is one of the cases where we decided that the SYSAFFIN statements were there only to enable guest mobility, and we would exploit LGR to move the guest to other SSI cluster members. Because we have EQIDs defined for the FCP devices, we can log the guest on to LTICVM9 and relocate it back and forth between LTICVM9 and LTICVM2.

Unfortunately, we do lose the ability to log this guest on and IPL Linux automatically starting at LTICVM2. The EQIDs do not provide that capability, unlike SYSAFFIN statements. If this guest logs on to LTICVM2 directly, it will be missing the FCP devices. This limitation would only be a problem during an extended outage on LTICVM9. We are not entirely sure how to address this restriction, other than to manually attach the equivalent devices after logging the guest on, but before attempting to IPL Linux.

The second change of note is in the profile: we added CHPIDVIRTUALIZATION ONE to the OPTION statement. That option turns on some required z/VM I/O virtualization functions to support LGR.

Exercising Live Guest Relocation

Once we made those directory changes and the SSI cluster was running well as reported by **q ssi**, we were ready to try an actual guest relocation.

```
q ssi
SSI Name: LTICS1
SSI Mode: Stable
Cross-System Timeouts: Enabled
SSI Persistent Data Record (PDR) device: S1COM2 on 8081
SLOT SYSTEMID STATE     PDR HEARTBEAT       RECEIVED HEARTBEAT
   1 LTICVM2  Joined    12/15/11   18:08:56 12/15/11   18:08:56
   2 LTICVM9  Joined    12/15/11   18:08:56 12/15/11   18:08:56
   3 LTICVM5  Joined    12/15/11   14:08:53 12/15/11   18:08:56
   4 LTICVM4  Down (shut down successfully)
Ready; T=0.01/0.01 18:09:26
```

Step 1. Our q ssi command has reported that the Cluster Mode is "Stable" so we are ready to test the relocate function with **vmrelo test**:

```
vmrelo test litswas1 lticvm2
HCPRLH1940E LITSWAS1 is not relocatable for the following reason(s):
HCPRLI1996I LITSWAS1: Virtual machine device 0190 is a link to a local
minidisk
HCPRLI1996I LITSWAS1: Virtual machine device 019D is a link to a local
minidisk
HCPRLI1996I LITSWAS1: Virtual machine device 019E is a link to a local
minidisk
Ready(01940); T=0.01/0.01 14:57:37
```

The vmrelo test has indicated we have a problem. A guest with links to local only minidisk is not relocatable. A local minidisk is any minidisk that is defined in a subconfig section in the directory. MAINT's 190, 19D, and 19E minidisks are defined a subconfig section because they reside on the System Residence volume for each z/VM LPAR, and are specific to the service level running on that LPAR.

Step 2. So, in order to relocate this guest, we must detach the CMS minidisks owned by MAINT. Luckily, our Linux config does not require these CMS disks while Linux is running, so we do not have a problem doing this. If your environment uses a CMS disk to hold Linux configuration data or application data, you will have to make sure that the disk is defined in the common section of the user who owns it, not in a subconfig.

```
for litswas1 cmd det 190-19f
LITSWAS1 : 0190-0191 DETACHED
LITSWAS1 : HCPDTV040E Device 0192 does not exist
LITSWAS1 : HCPDTV040E Device 0193 does not exist
LITSWAS1 : HCPDTV040E Device 0194 does not exist
LITSWAS1 : HCPDTV040E Device 0195 does not exist
LITSWAS1 : HCPDTV040E Device 0196 does not exist
LITSWAS1 : HCPDTV040E Device 0197 does not exist
LITSWAS1 : HCPDTV040E Device 0198 does not exist
LITSWAS1 : HCPDTV040E Device 0199 does not exist
LITSWAS1 : HCPDTV040E Device 019A does not exist
LITSWAS1 : HCPDTV040E Device 019B does not exist
LITSWAS1 : HCPDTV040E Device 019C does not exist
LITSWAS1 : 019D-019E DETACHED
LITSWAS1 : HCPDTV040E Device 019F does not exist
LITSWAS1 : HCPFOR069I Command Complete.  CP return code = 0040.
Ready; T=0.01/0.01 14:58:12
```

We have secondary user authority for this guest, so we were able to use the **for** command directly from the user we are issuing the vmrelo commands from.

Step 3. After those pesky CMS disks were detached, we tried the **vmrelo test** command again:

```
vmrelo test litswas1 lticvm2
User LITSWAS1 is eligible for relocation to LTICVM2
Ready; T=0.01/0.01 14:58:21
```

Step 4. That looks good, so we are ready to try actually moving a guest now with **vmrelo move**:

```
vmrelo move litswas1 lticvm2
Relocation of litswas1 from LTICVM9 to LTICVM2 started
User litswas1 has been relocated from LTICVM9 to LTICVM2
Ready; T=0.01/0.01 14:58:36
```

Step 5. And it is done! We were running a ping flood from an adjacent system against the guest while it was in the process of being relocated to see how many packets would be dropped during the process, and the results look promising:

```
# time ping -f -s 1400 litswas1
PING litswas1.pokprv.stglabs.ibm.com (10.20.7.57) 1400(1428) bytes of
data.
.................................
--- litswas1.pokprv.stglabs.ibm.com ping statistics ---
14200 packets transmitted, 14166 received, 0% packet loss, time 6948ms
rtt min/avg/max/mdev = 0.298/0.410/10.175/0.257 ms, ipg/ewma
0.489/0.352 ms


real 0m6.953s
user 0m0.110s
sys  0m0.494s
```

The total relocation time was less than seven seconds from start to finish. During those seven seconds, our ping flood transmitted 14200 echo requests, or 2.04 pings per millisecond. Counting the number of missing replies we get a total of 34 ping replies missing, which computes to a transit time of 16.66 milliseconds. That transit time is the interval between when the guest was stunned to sleep on the source system and when it woke up on the target and started replying to pings again.

Observations regarding Live Guest Relocation

There are a lot of factors involved in that transit time number. The CPU and memory load on both the source and destination z/VM systems are obvious factors, but also the Linux version itself and the complexity of the network environment must be considered.

When z/VM performs a live guest relocation, the first thing it does is to copy the entire memory address space of the guest from the source z/VM system to the target. When it completes that operation, it then examines the Linux guest's memory address space on the source system again to see how many pages changed between the last copy and now. z/VM then copies all the changed memory pages over to the target system. It iterates in this fashion several times, keeping track of how many changed pages it is copying every time.

When z/VM determines that the set of pages being copied is not getting any smaller, it then stuns the guest on the source system so that it stops changing pages. z/VM then copies the changed pages one last time, and then copies over the virtual CPU state as well as any other machine state. z/VM then attempts to start the virtual machine on the target system.

If all goes well, the guest wakes up on the target z/VM system, and z/VM sends the guest an interrupt to perform error recovery on all the devices, to instruct them to redrive their last I/O. When the guest is running normally on the target machine, z/VM then destroys the former virtual machine, which is still stunned on the source, and reports success.

If the virtual machine does not start up correctly on the target machine, z/VM has the option to simply delete everything on the target z/VM system, wake the guest back up on the source z/VM, and report the failure.

Obviously, if the guest has a larger memory configuration, it will take longer to copy the memory over from one machine to the other. This does not affect the transit time, or the interval when the guest is stunned and unresponsive on either the source or target systems, however. The transit time is directly related to the size of the very active working set of memory. z/VM will not attempt to move a guest that has such a large chunk of very active memory that the transit time will result in a noticeable outage.

We have observed some large guests (where large is approximately 6 GB of memory) take on the order of five minutes or more to move, from the time when vmrelo was entered to when z/VM reported success. The actual interval when the guest was unresponsive to input from an ssh session over the network was not apparent to us. The guest gave nearly instantaneous response to keyboard input the entire time.

The **vmrelo** command is synchronous by default, and z/VM Development recommends relocating only one guest at a time. There is the option to issue the command asynchronously and thus relocate many guests in parallel. In our experience this technique does not really get the z/VM LPAR evacuated any sooner than moving them one at a time, and we tended to see more relocation failures if we attempted to relocate guests in parallel.

Summary

The z/VM SSI clustering with LGR capability provides our test lab the ability to keep our Linux systems running longer, separating our Linux system availability from the z/VM host system that supports them. We are now able to schedule z/VM system maintenance windows that do not impact all the Linux virtual machines, which allows us to provide higher Linux system availability. Our Linux systems are now as available as permitted by the Linux kernel update schedule alone, not the sum of the Linux kernel update schedule and the z/VM CP update schedule.

We will be heavily exercising the z/VM 6.2 Live Guest Relocation function in the coming months and will be using it to keep Linux systems up and running even when their host z/VM LPAR needs an IPL to pick up z/VM service.

As always, we welcome your feedback and comments.

IBM Systems and Technology Group
Route 100
Somers, New York 10589
U.S.A.
Produced in the United States of America,
03/2012

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

ZSW03222-USEN-01