Getting Started [7]

This chapter describes the steps you need to take to get a guest system up-and-running in single-user mode. It is assumed that you have read the previous chapters of this document. If you want to go immediately to multiuser mode, read this chapter and then proceed to Chapter 8, page 45, which describes additional considerations for parameter and configuration files.

7.1 Preparing your host for a guest

This section describes the items that need to be in place or performed prior to booting a guest system.

7.1.1 General considerations

The following is a list of general considerations:

- Be certain that both your host and guest systems contain the UNICOS under UNICOS feature code. For information about supported systems, see Chapter 3, page 9, and Chapter 4, page 11.
- Evaluate your resource requirements

  General UNICOS System Administration, Cray Research publication SG–2301, contains information about main memory and mass storage requirements for the products associated with a particular UNICOS release. These requirements will vary, depending on the environment you configure for your site.
- Note that IOS-host-guest dumps can be much larger than host dumps (up to twice the size), so you may need to increase your dump partition. If taking dumps to an SWS, be sure that the file system on which dumps are placed has as much available space as possible (remove or compress old dumps).
- Evaluate mainframe memory requirements

  If you intend to run significant loads on both your host and guest systems and your current system is saturated, you may need additional memory. Memory is allocated at guest boot time and requires no changes to the host parameter file. The guest parameter file must not contain the `MEMORY` line.

Keep in mind that the amount of LDCHCORE (amount of mainframe memory used for ldcache) configured into your kernel or specified in the parameter file may be inappropriate (too large) for your guest system.

- Evaluate SSD requirements

  Unlike main memory, SSD is not dynamically allocated at guest boot time. If you want to allow the use of secondary data segments (SDS) and/or ldcache on the guest system, you will have to decrease the allocation of SSD at host boot time. This requires a change in your parameter file (param).

  You can still make use of the SSD slices on the host until the guest starts by maintaining them as scratch file systems. Make sure that these devices are not mounted (open) on the host at guest boot time.

  Note: SSD-T storage is not supported on the CRAY J90 and CRAY T90 series of systems.

- Check CPUS parameter

  The guest parameter file must not contain the CPUS parameter.

7.1.2 Guest file systems

When you are preparing your guest file systems, note the following information:

- A guest requires full root (/), /usr, and swap file system resources. A separate /tmp file system also is suggested. If you do not want to use your current back-up file systems, you will need a new guest root (/) and /usr file system.
- Any disk slice opened and/or mounted on a guest must **not** be opened and/or mounted on the host. Doing so will result in an error at device-open time.

7.1.3 IOS parameter files

This section describes IOS parameter file considerations.

Before running a guest, you may have to make some minor modifications to your current system's parameter file. The changes are discussed using excerpts from the parameter files. Changes also can be managed by the UNICOS Installation/Configuration Menu System.

Note: Although guest parameter files can be placed anywhere on the file system, it is recommended that you create a directory in `/usr/guest` to hold them.

Depending upon the number of physical slices comprising your guest's locally mounted (that is, non-NFS mounted) file systems, it may be necessary for you to add a `NGRT =` *nnn* specification to your host parameter file. This directive sets the number of GRT (Global Resource table) entries to the specified value. The GRT is present only in the host system.

In general, if the number of file system slices in the guest configuration exceeds one half that of the host, it would be advisable to increase the GRT count. The host's default GRT allocation will accommodate a guest having half the number of physical slices of the host. This should be sufficient for most sites.

If you decide to increase the GRT count, the following formula may be useful:

`NGRT =` *host_slices* `+` *guest_slices* `+` *N* `+` *T*

The factors in the expression are defined as follows:

host_slices	The number of disk and SSD physical slices configured in the host.
guest_slices	The number of disk and SSD physical slices configured in the guest.
N	The total number of network device logical paths configured in the guest.
T	The number of tape units configured in the host.

The host must be rebooted for the GRT size increase to take effect.

7.1.3.1 UNICOS (host) parameter files

The following are considerations for UNICOS (host) parameter files:

- Do you want to allocate SSD memory to the guest? Unlike mainframe memory, SSD memory cannot be allocated at guest boot time. Therefore, if you want the guest to use part of the SSD, you must change the host's IOS parameter file, removing usage of that portion of the SSD to be used by the guest, and reboot the host with the changed IOS parameter file.

  The following IOS model E example reserves two slices of SSD for the guest and requires that the SSD be repartitioned to create another slice.

The following is an SSD configuration before repartitioning:

```
SSD  ssd      {
length 128 Mwords;
pdd ssd_blk0     {minor    2; block        0; length  32768 blocks;}
pdd ssd_blk1     {minor    3; block    32768; length  32768 blocks;}
pdd ssd_blk2     {minor    4; block    65536; length  65536 blocks;}
pdd ssd_blk3     {minor    5; block   131072; length 131072 blocks;}
     }


ldd ssd_blk0     { minor    2; pdd ssd_blk0    ;}
ldd ssd_blk1     { minor    3; pdd ssd_blk1    ;}
ldd ssd_blk2     { minor    4; pdd ssd_blk2    ;}
ldd ssd_blk3     { minor    5; pdd ssd_blk3    ;}
```

The following is a sample configuration after making another slice available for the guest:

```
SSD  ssd      {
length 128 Mwords;
pdd ssd_blk0     {minor    2; block        0; length  32768 blocks;}
pdd ssd_blk1     {minor    3; block    32768; length  32768 blocks;}
pdd ssd_blk2     {minor    4; block    65536; length  32768 blocks;}
pdd ssd_blk2a    {minor    5; block    98304; length  32768 blocks;}
pdd ssd_blk3     {minor    6; block   131072; length 131072 blocks;}
     }


ldd ssd_blk0     { minor    2; pdd ssd_blk0    ;}
ldd ssd_blk1     { minor    3; pdd ssd_blk1    ;}
ldd ssd_blk2     { minor    4; pdd ssd_blk2    ;}
ldd ssd_blk2a    { minor    5; pdd ssd_blk2a   ;}
ldd ssd_blk3     { minor    6; pdd ssd_blk3    ;}
```

The following SSD slices are used on the host:

`ssd_blk0`	For `sdsdev`
`ssd_blk2`	For `swapdev`
`ssd_blk3`	For `/tmp`

The following slices are used by the guest:

`ssd_blk1`	For `swapdev`

`ssd_blk2a` For `sdsdevw`

- Will the guest be attempting to access a device not usually used on the host? If so, it should be configured for the host as well. Whenever possible, the host should be aware of all hardware to be used by both systems.

  In the following example, the host uses an NSC A130 for its network connection. The guest uses an NSC N130. The N130 could also have been shared between the host and guest by specifying different logical paths in `/etc/hycf.hyp`.

For example, logical path 5 has traditionally been used for TCP/IP on an N130. The adapter address for a system named `sn1703` is `0x86`. The current connection through logical path 5 was specified in `/etc/hycf.hyp`, as follows:

```
direct  sn1703-hyp    8605    ff00    0   16432;
```

The following line was added to allow another connection using logical path 7 for the guest:

```
direct  sn1703a-hyp  8607    ff00    0    16432;
```

Then the following line was added to the `/etc/hosts` file:

```
128.162.82.80   sn1703a sn1703a-hyp sn1703a.cray.com
```

As with any network connection information, these changes must be broadcasted to all machines that require access to the guest.

Note: The N130 and A130 adapter must be made aware of the new IP address by placing the address and its associated logical path in the NSC startup file. See your local network administrator or your NSC service representative for more information.

The network device configuration follows:

```
npdev 0 {
        iopath {cluster 0; eiop 0; channel 030;}
        np_spec S_FEI3FY;
        }
npdev 1 {
        iopath {cluster 0; eiop 0; channel 032;}
        np_spec S_N130;
}
npdev 2 {
        iopath {cluster 0; eiop 0; channel 034;}
        np_spec S_FEIVA;
```

```
}
npdev 3 {
        iopath {cluster 0; eiop 0; channel 036;}
        np_spec S_A130;
}
```

7.1.3.2 Guest parameter files

Like any UNICOS system, the guest requires a parameter file. Unlike a stand-alone system, the guest parameter file is not transferred to the operator workstation (OWS or SWS), but it remains on your mainframe disk. To prevent configuration overlaps, make a copy of the host parameter file on the mainframe.

To create a guest parameter file from a copy of the (new) host parameter file created in the previous section, make the following changes:

- Remove the following mainframe specification from the guest parameter file:

  nn `Mwords memory;`

  A guest can be allocated different amounts of memory at each boot. This parameter prevents a guest from starting with anything less than *nn* Mwords.

- Change `rootdev`, `swapdev`, and `sdsdev`:

  Old (host): `rootdev` is `ldd root.a;`

  `swapdev` is `ldd ssd_blk2;`

  `sdsdev` is `pdd ssd_blk0;`

  New (guest): `rootdev` is `ldd root.b;`

  `swapdev` is `ldd ssd_blk1;`

  `sdsdev` is `pdd ssd_blk2a;`

  Note: A guest root device mounted on the host at guest boot time will result in a guest system panic. This is due to a configuration overlap. File systems can be automatically unmounted at guest boot time by specifying the corresponding logical device in your guest.rc configuration file. For more information, see the `guest`(1) man page.

7.1.4 Booting the host system

After you edit your parameter file (as described in Section 6.2, page 15), you will need to reboot your system by using the new parameter file in order for the updated information to take effect.

7.1.5 Configuring the UNICOS under UNICOS feature

After you have considered the impact of using this feature on your host, you are ready to configure guest defaults for your system.

During the UNICOS installation procedure, a `/usr/guest` directory is created. The UNICOS under UNICOS feature configuration information is kept in that directory. Although it is not necessary, it is suggested that you make this a separate file system that can be mounted on any root when that root is running as the host.

This section describes how to use the UNICOS Installation/Configuration Menu System to update the guest configuration files. Excerpts from the resulting files are included if you choose to view or edit them manually. If you do edit them manually, you need to set the `GUEST` and/or `GUESTADMIN` permission bits in the users' `udb` entry of each validated user so that the user has the `/etc/guest`(1) command privileges required for booting their guest.

For information on guest permission under a UNICOS MLS system, see the `guest`(1) man page.

1. Invoke the UNICOS Installation/ Configuration Menu System by entering the following commands:

   ```
   cd /etc/install
   ./install
   ```

2. These commands get you into the main installation menu. Select the following from the menu:

   ```
   M-> Configure System
   ```

3. Select the following option:

   ```
         Asynchronous libraries configuration ==>
         Dumpsys utility configuration ==>
   M->   UNICOS under UNICOS (guest) configuration ==>
         Miscellaneous software configuration ==>
         Import the configuration ...
         Activate the configuration ...
   ```

From there, the `UNICOS under UNICOS (guest) Configuration` menu comes up. Select the following from the menu:

```
M-> Guest defaults
```

Initially, you will have defaults shown in the following screen example. In most cases, these defaults should be sufficient. If you want to make changes, refer to the context-sensitive help facility available with the UNICOS Installation/Configuration Menu System.

To see your location in the menu system, use the `Where am I` (`W`) option. The `Guest Defaults` location is shown as follows:

```
    UNICOS 7.0 Installation / Configuration Menu System
.   Configure System
.   .   UNICOS under UNICOS (guest) Configuration
.   .   .   Guest Defaults
```

In this section, the location of menu selections is prefaced with a similar location header.

```
                  Guest Defaults


            Miscellaneous Options

    Maximum number of guest systems allowed.
      A value of zero (0) will DISABLE all
      guest(8) command functions.                  1

    Enable GUEST/HOST kernel tracing?              NO
    Halt HOST when guest panics?                   NO
    Maximum TTY connnections per guest             1
    Create user directories below /usr/guest?      YES
    Remove non-valid directories in /usr/guest     NO

            Memory Size Options

    Minimum HOST memory in megawords               8
    Minimum GUEST memory in megawords              8
    Maximum GUEST memory in megawords              8

            CPU Percentage Options

    CPU allocation scheme                          MEMORY
      Minimum HOST CPU percentage
      Maximum GUEST CPU percentage

A-> Reset global guest defaults...
```

Upon activation, the information that you provided in the previous menu produces a file called `Defaults` in the `/usr/guest` directory. A sample is included in the following screen.

```
#
#         UNICOS under UNICOS (guest) Global Defaults
#
# This file contains global guest defaults, which may be
# overridden by user values set in the guest Users file.
#
# format:
#       'option' = 'value'
#
#
# Miscellaneous Options
#
MAX_GUESTS=1
GUEST_KERNEL_TRACING=NO
HALT_HOST_ON_GUEST_PANIC=NO
TTY_CONNECTIONS=1
#
# Memory Options
#
MIN_HOST_MEMORY=8
MIN_GUEST_MEMORY=8
MAX_GUEST_MEMORY=8
#
# CPU Percentage Options
#
# CPU_ALLOCATION_SCHEME=MEMORY
```

If your system is significantly larger than the 16 Mwords (such as 128 Mwords), it is probable that the kernel configured for your system will not boot successfully in 8 Mwords. Increase the maximum/minimum guest memory size to at least 16 Mwords and try again.

7.2 Validating a user

After you have configured the guest defaults on the host, a user can be validated to start guest systems. All users, including the administrator, must be validated before they may use the UNICOS under UNICOS feature for anything other than status. Usually, each valid user has a directory in the `/usr/guest` directory.

Note: On an MLS system, you will need to have the `SYSADM` permission (or greater) to activate your configuration.

To validate a user, use the UNICOS Installation/Configuration Menu System, and make the following menu selection:

```
    UNICOS 10.0 Installation / Configuration Menu System
.   Configure System
.   .   UNICOS under UNICOS (guest) Configuration
.   .   .   Guest User Validation
```

After making the selection, press the letter `N`. The following screen appears:

```
    User    Admin?  Guests  Memory  KTrace?  Halt?  Ttys
    -----   -----  ------  ------  ------  ------- -----
E->          NO      0       8        NO      NO    1
```

The minimum requirement for validating a single user is to enter in the user's login name and change the number of allowable guests to 1, as follows. Use the editing capabilities available with the UNICOS Installation/Configuration Menu System to validate users, as shown in the following screen:

```
    User    Admin?  Guests  Memory  KTrace?  Halt?  Ttys
    -----   -----  ------  ------  ------  ------- -----
E->  jqp     NO      1       8        NO      NO    1
```

Note: After you have made any changes, be sure to activate your guest configuration. Also, remind the validated user or users to log out and log back in for UDB changes to take effect.

The information that you provide in this menu updates the `Users` file in the `/usr/guest` directory and in the `/etc/udb` file; it also creates a directory for the user in the `/usr/guest` directory. The `Users` file contains the default settings for each validated user. A skeleton `guest.rc` file is created in the user's `/usr/guest` directory. Changes to this file are required by the user before they can successfully boot a guest. This file is analogous to a user's `.login` or `.profile` file and is not updated by future menu system activations.

7.2.1 Device validation

Now that there is at least one validated user on your system, you must identify the file systems (root (`/`), `/usr`, `/src`) that will be mounted on the guest. If these file systems are usually mounted on the host, guest users must be validated to unmount them at guest system startup.

To validate a user, make the following menu selection:

```
    UNICOS 7.0 Installation / Configuration Menu System
.   Configure System
.   .   UNICOS under UNICOS (guest) Configuration
.   .   .   Guest Logical Device Authorization
```

In the following example, user `jqp` has been validated for `root_d`, `usr_d`, and `src_d`:

```
Guest Logical Device Authorization

      Logical Device     Guest User
       -----------------  ----------
  E->  /dev/dsk/root_d    jqp
       /dev/dsk/usr_d     jqp
       /dev/dsk/src_d     jqp
```

Note: After you have made any changes, be sure to activate your guest configuration.

The following is an example of a resulting configuration file `/usr/guest`:

```
#         UNICOS under UNICOS (guest) User Defaults
#
# This file contains defaults specific to each authorized guest user.

# Options not specified here will default to the settings in
# /usr/guest/Defaults.
#
# format:
#       [user:[user:...]]'option' = 'value'
#
#
# Guest Options for User: jqp
#
jqp:MAX_GUESTS=1
jqp:MAX_GUEST_MEMORY=8
jqp:GUEST_KERNEL_TRACING=YES
jqp:HALT_HOST_ON_GUEST_PANIC=NO
jqp:TTY_CONNECTIONS=1

#
# Logical Devices for User: jqp
#
jqp:LOGICAL_DEVICES="/dev/dsk/root_d \
        /dev/dsk/src_d \
        /dev/dsk/usr_d"
```

Keep in mind that this also will update the UDB when activated.

Note: The user's `guest.rc` file will not be updated by the install tool. It is created for the user as a courtesy when the user is first validated. After that, the user has complete control over his or her `guest.rc` file. If additional file systems are added to the user's authorized list by the administrator, the user will be responsible for appropriately updating his or her `guest.rc` file.

7.2.2 `guest.rc` file

Once validated, you will have a file called `guest.rc` located in the `/usr/guest/` *login* directory.

Note: This file is created for the user only once by the initial activation, and then must be maintained manually by the user. Further install tool activations will not affect the user's `guest.rc` file.

The `guest.rc` file identifies the particular support files and program levels to use for your guest. Review this file, and ensure that the UNICOS version you plan to boot contains support for the UNICOS under UNICOS feature. Pay particular attention to the following entries:

- `NODE_NAME`

  Once again, your guest is a unique system that must have a unique node name. If your kernel was not built with a `uts` node name that is different than that of the host, you need to specify a name here.

- `CRASH`

  Your `crash`(8) level should match the level of your guest kernel.

- `KCOMPRESS`

  Your `kcompress`(8) level **must** match the level of your guest kernel.

- `PARAM`

  Designates the UNICOS parameter file to use for your guest.

- `PARAM_CHECKER`

  Designates the parameter file validation program appropriate for your system level (such as `/etc/econfig`).

- `LOGICAL_DEVICES`

  Denotes the list of logical devices that will be dedicated to your guest and that must be unmounted from the host at guest startup. Review the `/usr/guest/Users` file for your authorized list.

A `guest.rc` file is not required, but it is advisable to always have one.

The following screen shows an example `guest.rc` file for user `jqp`.

```
# This file provides a mechanism for configuring the actions of the guest(1)
# command.  Each available user option is listed and briefly explained. (See
# the guest(1) command man page for more detailed information.)
#
# The option (and comment) format are similar to UNIX shell scripts:
#       option = 'value'
#
# Available options include:
#
#                       Description/
# Option                (Default)
#-------------------------------------------------
# CRASH                 location of crash binary
#                       (/etc/crash)
#
# DUMP_DIRECTORY        directory below which systems dumps are written
#                       (./dump)
#
# KCOMPRESS             location of kernel (de)compression binary
#                       (/etc/kcompress)
#
# KERNEL                location of kernel binary
#                       (./unicos)
#
# LOGICAL_DEVICES       list of logical devices to unmount before startup
#                       (NO DEFAULT)
#
# MEMSIZE             requested memory size
#                     (value of MAX_GUEST_MEMORY from
#                       /usr/guest/Users)
#
# MIN_MEMSIZE         minimum memory size that you will accept
#                     (value of MIN_GUEST_MEMORY from
#                      /usr/guest/Defaults)
#
# NODE_NAME           node name
#                     (system name from the kernel binary)
#
# PARAM               location of system parameter file (./param)
#
# TTY_CONNECTIONS     number of tty connections requested (1)
#
```

```
#
# The administrator has preset your options as follows.  (See the
# /usr/guest/Defaults and /usr/guest/Users files for more information on
# your validation.)
#
NODE_NAME=galeg1
PARAM=/usr/guest/ccn/p.root_j
CRASH=/etc/crash
KCOMPRESS=/usr/guest/jqp/kcompress_90
KERNEL=/usr/guest/jqp/unicos_90
LOGICAL_DEVICES="/dev/dsk/root_d \n        /dev/dsk/src_d \n        /dev/dsk/usr_d"
MEMSIZE=8
MIN_MEMSIZE=4
```

7.2.3 Checking your configuration

It is recommended that you verify that the configuration in your `guest.rc` file is valid. To do so, simply use the `-a` option of the `guest`(1) command as follows:

```
guest -a
```

For more information about the `-a` option, see the `guest`(1) man page. Before continuing, correct any errors or warnings noted by this verification option.

7.3 Booting a guest to single-user mode

Now that you have verified your configuration, you are ready to boot your guest system.

Note: Before starting your guest, you should make sure that zip lines other than `0` (the host console) are not in use. The guest boot causes a disconnect to occur on any active zip lines that are needed by the guest.

You should also review your `guest.rc` file to determine whether locations of files (such as `crash`, `kcompress`, and `unicos`) are available in the specified locations. Run the command `guest -a` to authenticate your `guest.rc` configuration and address warnings or errors before proceeding.

7.3.1 Startup

To start your guest operating system, use the following command:

```
guest -s
```

7.3.2 Console connections

If successfully started , the `guest`(1) command reports the number of your guest system's console on the OWS or SWS. Log in to the OWS, and enter the following command:

```
zip #
```

On GigaRing based systems, log in to the SWS and enter the following command:

```
mfcon -l #
```

7.3.3 Start-up output example

The following is an example output for a guest start-up session:

```
gale jqp &lt;23> pwd
/usr/guest/jqp

gale jqp &lt;24> ls
crash_90      guest.rc      map_90
dumps         kcompress_90  unicos_90

gale jqp &lt;25> tail -14 guest.rc
#
NODE_NAME=galeg1

CRASH=./crash_90
KCOMPRESS=./kcompress_90
KERNEL=./unicos_90

PARAM=/usr/guest/ccn/p.root_j
LOGICAL_DEVICES=" /dev/dsk/root_j \
        /dev/dsk/src_j \
        /dev/dsk/usr_j "

MEMSIZE=8
MIN_MEMSIZE=8
```

```
gale jqp &lt;26> guest -s
gst-45 guest: Info
   The guest kernel binary appears to be compressed.
   Decompressing it with:
        ./kcompress_90

gst-46 guest: Info
   The guest kernel binary decompression was successfully completed.

gst-43 guest: Info
   Changing the guest system name from:
        gale
   to:
        galeg1
/gst90/usr/src is mounted.
        Attempting to unmount...done.

/gst90/usr is mounted.
        Attempting to unmount...done.

/gst90 is mounted.
        Attempting to unmount...done.

gst-53 guest: Info
   A 8 MW Guest system (galeg1) has been successfully started for
user:
        jqp

gst-55 guest: Info
   The guest (galeg1) system console is:
        zip 1
   on the host (gale) system's OWS.
```

7.3.4 Status

Use the following command to get the status of your guest system. The `-v` option gives you the tty number of your console on the IOS:

```
guest -v
```

```
gale jqp &lt;27> guest -v
gst-2 guest: Info
   guest Version 9.3.0ed, generated on 09/15/97, at 15:57:03.


              System  CPU       Address Size
Console Ord     Name  (%)       (words) (MW)  Status       Owner  Flags

 zip 0   0     gale   57             0   16  executing     HOST  M
 zip 1   1   galeg1   43      18300928   14  executing      jqp  S

Flag values:
        M (multi-user), S (single-user), F (frozen), H (halt on panic)
        B (breakpointed)

Number of CPUs available to guests:  2
Additional guest tracing is:         ENABLED
```

7.3.5 Dumping

To perform a trial dump of your guest system, use the following command:

```
guest -d
```

You do **not** have to stop the guest before dumping it. The guest system resumes execution once the dump has completed.

Keep in mind that dumps from your guest system can be quite large. You should set `DUMP_DIRECTORY=` *path* in your `guest.rc` file, where *path* is a file system with enough unused space to handle your particular requirements. For information on viewing the dump, see Appendix A.

7.3.6 Ending a session

When you have finished your guest session, execute the following command to end the session:

```
guest -q
```

This command stops the guest system (`guest -x`), releases system memory held by the guest (`guest -r`), and remounts any file systems unmounted at guest boot time.

The following is an example of the output for a guest system stop and release:

```
gale jqp &lt;40> guest -q
gst-15 guest: Info
   The guest options file:
        guest.rc
   was found in the following directory:
        /usr/guest/jqp/

gst-16 guest: Info
   Changing the current working directory to:
        /usr/guest/jqp/

gst-61 guest: Info
   The guest system (galeg1) has been successfully stopped.


gst-316 guest: Warning
   There is outstanding I/O on iopath:
        0032.0
   Waiting for I/O completion.

sys-16 guest: Info
   Device busy

gst-37 guest: Info
   The guest memory (14 MW) has been returned to host.


/etc/mfsck: Starting pass 1


/etc/mfsck: Starting pass 2
```

```
/usr_j: file system opened
/usr_j: super block fname usr_j, fpack sn1703
/src_j: file system opened
/src_j: super block fname src_j, fpack sn1703
/root_j: file system opened
/root_j: super block fname root_j, fpack sn1703
/usr_j: Phase 1 - Check Blocks and Sizes
/root_j: Phase 1 - Check Blocks and Sizes/src_j: Phase 1 - Check Blocks and Sizes
/root_j: Phase 2 - Visit Directories
/usr_j: Phase 2 - Visit Directories
/root_j: Phase 3 - Checking Directories
/root_j: Phase 4 - Checking Non-Directories and Link Counts
/root_j: Phase 5 - Verify Dynamic Information - (Ignored)
/root_j: Phase 6 - Rebuilding Dynamic Information
/root_j: file system summary
/root_j:        32768 total i-nodes (29073 free i-nodes)
/root_j:        137088 total blocks  (65395 free blocks)
/root_j:  ***** FILE SYSTEM WAS MODIFIED *****
/usr_j: Phase 3 - Checking Directories
/usr_j: Phase 4 - Checking Non-Directories and Link Counts
/usr_j: Phase 5 - Verify Dynamic Information - (Ignored)
/usr_j: Phase 6 - Rebuilding Dynamic Information
/usr_j: file system summary
/usr_j:         32768 total i-nodes (25704 free i-nodes)
/usr_j:         137088 total blocks  (39475 free blocks)
/usr_j:  ***** FILE SYSTEM WAS MODIFIED *****
/src_j: Phase 2 - Visit Directories
/src_j: Phase 3 - Checking Directories
/src_j: Phase 4 - Checking Non-Directories and Link Counts
/src_j: Phase 5 - Verify Dynamic Information - (Ignored)
/src_j: Phase 6 - Rebuilding Dynamic Information
/src_j: file system summary
/src_j:         85712 total i-nodes (45087 free i-nodes)
/src_j:         365040 total blocks  (102446 free blocks)
/src_j:  ***** FILE SYSTEM WAS MODIFIED *****
/etc/gencat: complete (9 secs.)

/gst90 is not mounted.
   Attempting to mount /dev/dsk/root_j on /gst90...done.

/gst90/usr is not mounted.
   Attempting to mount /dev/dsk/usr_j on /gst90/usr...done.

/gst90/usr/src is not mounted.
   Attempting to mount /dev/dsk/src_j on /gst90/usr/src...done.
```