Process Resource Manager (PRM)

INDEX

Introduction

The HP Process Resource Manager (PRM) is a resource management tool used to control the amount of resources that processes use during peak system load (at 100% CPU, 100% memory, or 100% disk bandwidth utilization). PRM can guarantee a minimum allocation of system resources available to a group of processes through the use of PRM groups. A PRM group is a collection of users and applications that are joined together and assigned certain amounts of CPU, memory and disk bandwidth. The two types of PRM groups are *FSS groups* and *PSET groups*. An FSS group is the traditional PRM group, whose CPU entitlement is specified in shares. This group uses the Fair Share Scheduler (FSS) in the HP-UX kernel within the system's default processor set (PSET). A PSET group is a PRM group whose CPU entitlement is specified by assigning it a subset of the system's processors (PSET). Processes in a PSET have equal access to CPU cycles on their assigned CPUs through the HP-UX standard scheduler.

PRM's Resource Managers

PRM has four resource managers:

CPU (processor time)
Ensures that each PRM group is granted at least its allocation of CPU. Optionally for FSS PRM groups, this resource manager ensures no more than its capped amount of CPU. For PSET PRM groups, processes are capped on CPU availability by the number of processors assigned to the group.

APPL (application)
Ensures that specified applications and their child processes run in the appropriate PRM groups.

MEM (memory)
Ensures that each PRM group is granted at least its share, but (optionally) no more than its capped amount of real memory. Additionally, with memory resource groups (MRGs) which are available as of UX 11.11, you can specify memory shares be isolated so that a group's assigned memory shares cannot be loaned out to or borrowed from other groups.
Note that MRG gives the ability to logically partition private (i.e., user) memory into multiple MRGs. It does not, however, support the partitioning of the remaining user memory resource types: **shared memory**, shared-text, or shared library code. These resource types are managed by the global MRG group 0. Hence all shared memory types are allocated to the PRM_SYS group.

DISK (disk bandwidth)
Ensures that each FSS PRM group is granted at least its share of disk bandwidth. PRM disk bandwidth management can only control disks that are managed by Logical Volume Manager (LVM). PSET PRM groups are treated as part of PRM_SYS (PRMID 0) for disk bandwidth purposes.

The following **daemons** belong to the PRM subsystem:

prm0d	the old fashioned (pre UX 11.11) memory manager (MEM) (*)
prm2d	the new memory manager (MEM) available as of UX 11.11 (*)
prm1d	the application manager (APPL)
prm3d	enables remote management of PRM by the xprm GUI

CPU and DISK resource management is done in the kernel.

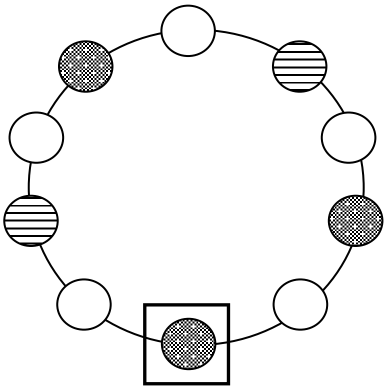
(*) **NOTE:**
HP-UX 11.11 supports memory resource groups (MRGs) and can directly enforce PRM memory allocations. Physical memory is allocated to memory groups based on the PRM configuration. When the system is paging and a memory group is exceeding its shares, under prm2d memory management the MRG kernel causes the process to page.

HP-UX 11.00 does not support memory groups, so the operating system cannot directly enforce PRM memory allocations. PRM indirectly controls memory allocation by suspending processes in groups that are exceeding their memory cap. Because suspended processes cannot touch their memory pages, they are the most likely to be paged out by the operating system. This is handled by the old fashioned prm0d memory manger.

How the FSS scheduler works

Consider the PRM CPU scheduler as the carousel below. The circles represent FSS PRM groups. There are 3 groups. The number of circles a group has in the carousel is determined by its CPU entitlement.

Group1 having 50% CPU entitlement, hence owning 5 of 10 circles.
Group2 having 30% CPU entitlement, hence owning 3 of 10 circles.
Group3 having 20% CPU entitlement, hence owning 2 of 10 circles.



With CPU capping off
If a group does not need the CPU, the scheduler checks the next group for work skipping the current.

With CPU capping on
The group with control of the CPU retains control, regardless of need, until the clock tick

ends.

When system use is low, CPU capping can cause the system to idle. This is because a PRM group is given its full entitlement of CPU, whether it uses it or not.
As of today CPU use can be capped for either **all** FSS PRM groups or **no** FSS PRM groups. For PSET PRM groups, capping is the result of the amount of CPUs assigned to the group.

Installation

Current PRM Version and Patches

PRM runs on servers as well as on workstations with HP-UX 10.20 or greater. For HP-UX 10.20 there are only versions <= C.01.07 available.

A full functional 90 day trial license can be downloaded from http://hp.com/go/prm.

The current version is C.02.03 available as of Application Release Sep 2004:

```
     B3835DA           C.02.03        HP Process Resource Manager
```

PRM version **C.02.xx** uses the following new OS features that were introduced with **UX 11.11:**

- in kernel memory control with memory resource groups (**MRG**)
- processor sets (**PSETS**)

More about PRM C.02.xx in the appendix.

The PRM bundle contains 3 products:

```
$ swlist -l product -s swserver :/depot/darts/64.Mar04/11.11 B3835DA
Proc-Resrc-Mgr  C.02.02  Process Resource Manager Proc-Resrc-Mgr product
PRM-Sw-Krn      C.01.01  Process Resource Manager PRM-Sw-Krn product
PRM-Sw-Lib      C.02.02  Process Resource Manager PRM-Sw-Lib product
```

NOTE: The installation of the libraries collection (and therefore of the whole bundle) requires a system reboot! Starting with PRM Version C.02.00.02, PRM includes a new product fileset PRM-Sw-Krn.PRM-KRN. This dummy fileset (simply contains a Readme file) prevents system reboots when you upgrade to future PRM versions. If the version of the fileset being installed is the same as the version of the fileset already installed, there will be no reboot.Without the reboot though, the prm3d daemon and the PRM SNMP agent (prmagt) do not restart and need to be started manually:

```
# /opt/prm/bin/prm3d
# /opt/prm/bin/prmagt
```

PRM Release History

	10.20	11.00	11.11	11.22	11.23	As of Appl Release
C.01.07	✓	✓				Dec 99
C.01.08		✓	✓			Jun 00
C.01.08.2		✓	✓			Dec 00
C.02.00		✓	✓			Sep 01
C.02.00.02		✓	✓			Jun 02
C.02.00.04				✓		
C.02.01		✓	✓			Jun 03
C.02.01.01					✓	Sep 03
C.02.02		✓	✓		✓	Mar 04
C.02.03			✓		✓	Sep 04

After the PRM product the following patches (or newer ones) including their dependencies need to be installed:

for UX 11.00:

PRM product patches (choose one, depending on your PRM version):
PHSS_24864 - 11.X PRM C.01.08.2 Cumulative Patch
PHSS_26845 - 11.X PRM C.02.00 Cumulative Patch
PHSS_26846 - 11.X PRM C.02.00.02 Cumulative Patch
PHSS_30985 - 11.X PRM C.02.02 Cumulative Patch

OS patches:
PHKL_27089 - 11.00 syscall, signal, umask cumulative patch *(important if >=24 cpus)*
PHSS_25277 - 11.00 Xserver cumulative patch
PHSS_25091 - 11.00 X Font Server SEP2001 Periodic Patch
PHSS_26490 - 11.00 CDE Runtime Periodic Patch
PHCO_26000 - 11.00 Pthread library cumulative patch
PHCO_25707 - 11.00 libc cumulative patch
PHKL_25775 - 11.00 LVM cumulative patch, performance upgrades
PHCO_26274 - 11.00 ps(1) cumulative patch
PHCO_23189 - 11.00 libnss_files cumulative patch

for UX 11.11 = UX 11i (PRM versions >= C.01.08.2):

PRM product patches (choose one, depending on your PRM version):
PHSS_24864 - 11.X PRM C.01.08.2 Cumulative Patch
PHSS_26845 - 11.X PRM C.02.00 Cumulative Patch
PHSS_26846 - 11.X PRM C.02.00.02 Cumulative Patch
PHSS_30985 - 11.X PRM C.02.02 Cumulative Patch

OS patches:
PHCO_24478 - 11.11 mpsched(1) patch
PHKL_29704 - 11.11 Psets Enablement, SCHED_NOAGE, FSS
PHKL_30034 - 11.11 Psets Enablement Patch; FSS cap
PHKL_30035 - 11.11 Psets Enablement; FSS iCOD; callback; FSS
PHKL_30197 - 11.11 Psets, vPar, SLVM, Reboot Hangs, Serial Num

PHKL_30398 - 11.11 FSS ID and KI_rfscall
PHKL_30977 - 11.11 vPars panic;Syscall cumulative;FSS;msem_lock
PHKL_31993 - 11.11 Core PM, vPar, Psets, slpq1, FSS, rtprio
PHKL_31995 - 11.11 RTE PSETs Enablement, FSS, PSETs
PHKL_32061 - 11.11 detach;NOSTOP,Abrt,Psets;slpq1;FSS;getlwp
PHKL_32619 - 11.11 Cumulative VM, Psets, Preemption, PRM, MRG

for UX 11.23 = UX 11i v2 (PRM versions >= C.02.02):

PRM product patches:
PHSS_30986 - 11.23 PRM C.02.02 Cumulative Patch

OS patches:
PHKL_33052 - 11.23 Single CPU FSS capping algorithm fix
PHKL_31500 - 11.23 Sept04 base patch
(need BUNDLE 11i Sep04 for that patch, refer to special installation intructions)

If you like to use the Java GUI then do also install the Java related patches on
http://hp.com/go/java (follow the "patches" link).

Post Installation Steps

- **PRM driver:**
  Check if prm driver is in the kernel. It should be after successful installation of PRM:

```
# grep -i prm /stand/system
prm
```

- **Link GlancePlus Library:**
  There is a private PRM library shipped with the GlancePlus product. Glance needs the library in order to show performance data based on PRM groups (“P” key). Unfortunately the library has some problems. Processes may not be linked to their appropriate PRM groups. To fix this just use the version of the library that is shipped with the PRM product.

  Check if Glance is installed on your system:

```
# swlist -l product Glance
  Glance                C.03.50.00     HP GlancePlus/UX
```

  NOTE: It is recommended to update Glance/Measureware since new features like PSETS are not implemented in older versions.

  Link the library as follows in case it is installed do the following:

  1. Stop all running GlancePlus instances. Stop Measureware daemons using `/sbin/init.d/mwa stop`.

  2. Update GlancePlus’s PRM library:

```
$ mv /opt/perf/lib/libprmext.2 /opt/perf/lib/libprmext.2.org
$ ln -s /opt/prm/lib/libprmext.2 /opt/perf/lib/libprmext.2
$ ll /opt/perf/lib/libprmext.2
lrwxrwxr-x   1 root       sys             24 May  3 09:41
/opt/perf/lib/libprmext.2 -> /opt/prm/lib/libprmext.2
```

NOTE: for UX 11.2X the libraries are at different locations:

```
$ mv /opt/perf/lib/hpux32/libprmext.so.2 \
/opt/perf/lib/hpux32/libprmext.so.2.org

$ ln –s /opt/prm/lib/hpux32/libprmext.so.2 \
/opt/perf/lib/hpux32/libprmext.so.2

#ll /opt/perf/lib/hpux32/libprmext.so.2

lrwxr-xr-x   1 root       sys             34 Dec 13 15:45
/opt/perf/lib/hpux32/libprmext.so.2 ->
/opt/prm/lib/hpux32/libprmext.so.2
```

3. Start Measureware daemons again using `/sbin/init.d/mwa start`.

Preparation

The following sections helps you to answer the following questions:

- Which applications should run under control of PRM?
- Which processes belong to those applications?
- What should be the names of the PRM groups?
- Which of the system resources (CPU, MEMORY, DISK) should be controlled by PRM?
- Which entitlements should each PRM group have?

Dividing Processes into Groups

Therefore you need the output of the process table during normal system load:

```
$ ps -ef

     UID     PID  PPID  C    STIME TTY       TIME COMMAND
    root       0     0  0  Jan 10  ?         1:09 swapper
    root       1     0  0  Jan 10  ?         1:34 init
    root       2     0  0  Jan 10  ?         8:24 vhand
    root       3     0  0  Jan 10  ?        79:28 statdaemon
    root       4     0  0  Jan 10  ?         1:03 unhashdaemon
    root       8     0  0  Jan 10  ?         0:00 supsched
    root      12     0  0  Jan 10  ?        33:18 ttisr
    root      19     0  0  Jan 10  ?         2:23 lvmkd
     ...
     ...
  kdoi22    5403     1  0  Jan 10  ?         0:42 ora_smon_bfl
  kdoi22    5242     1  0  Jan 10  ?         1:18
/oracle/app/oracle/product/8.1.5/bin/tnslsnr LISTENER -inherit
  kdoi22    5420     1  0  Jan 10  ?         0:12 ora_s000_bfl
  kdoi22    5422     1  0  Jan 10  ?         0:13 ora_s001_bfl
  kdoi22    5424     1  0  Jan 10  ?         0:12 ora_s002_bfl
```

```
kdoi22    5426     1  0  Jan 10  ?         0:11 ora_s003_bfl
kdoi22    5428     1  0  Jan 10  ?         0:12 ora_s004_bfl
   ...
   ...
```

There are several possibilites to associate the application processes with PRM groups:

1. **based on the User-ID (UID)**

   All processes belonging to the users `oesdba`, `oesinit` and `oesguest` should e.g. be assigned to a PRM group named `EXPRESS`.

   All processes belonging to the user `oraias` should e.g. be assigned to a PRM group named `OAS`.

2. **based on the executables (inode number)**

   In some cases the classification based on the user id is not sufficient, i.e. you want to put processes from the same user into different groups or processes from different (maybe newly added) users into the same group.
   In such a case you have the possibility to specify the processes based on their executable (PRM uses the inode of the executable file).

   Example:

   All processes belonging to Perl should be placed in the PRM group `PRM_SYS`. You need to know the name of the executable (`/usr/contrib/bin/perl`).

   The PRM application manager checks that applications are running in the correct PRM groups every *interval* seconds. The default interval is 30 seconds.

3. **using alternate names**

   There are situations where even the classification above is not sufficient. Imagine processes created by other processes. These processes don't have full qualified executables and therefore have no disk inode.
   Anyway - PRM application manager allows you to place these processes into separate PRM groups by using alternate names

   Oracle example:
   If you intend to put Oracle database instances into different PRM groups you need to consider that when you open an instance, the Oracle executable (`$ORACLE_HOME/bin/oracle`) renames itself regarding the environment variable `$ORACLE_SID`. The Oracle processes are named:

   ```
   ora_<process-name>_$ORACLE_SID
   ```

   `$ORACLE_HOME` and `$ORACLE_SID` are standard environment variable names for Oracle's home directory (usually `/oracle/app/oracle/product/<oracle-version>/`) and the name of the Oracle database instance, respectively.

   The Oracle Server Manager (`$ORACLE_HOME/bin/svrmgrl`) starts following instance processes:

```
/oracle/app/oracle/product/8.0.6          $ORACLE_SID = "ORA_ORM"
/oracle/app/oracle/product/8.1.5          $ORACLE_SID = "ORA_BFL"
```

The instances are listed in the file `/etc/oratab`:

```
bfl:/oracle/app/oracle/product/8.1.5:Y
ormt:/oracle/app/oracle/product/8.0.6:Y
ormq:/oracle/app/oracle/product/8.0.6:Y
...
```

These instances create the following sub-processes by doing fork & rename:

Instance ORA_ORM	**Instance ORA_BFL**

`$ ps -ef | grep orm`
 	
`ora_arch_orm7t` (server process)
`ora_ckpt_orm7t` (server process)
`ora_dbw0_orm7t` (server process)
`ora_lgwr_orm7t` (server process)
`ora_pmon_orm7t` (server process)
`ora_reco_orm7t` (server process)
`ora_smon_orm7t` (server process)
`oracleorm7t` (shadow process)
`oracleorm7t` (shadow process)
`...`
 	
`ora_arch_orm7q` (server process)
`ora_ckpt_orm7q` (server process)
`ora_dbw0_orm7q` (server process)
`ora_lgwr_orm7q` (server process)
`ora_pmon_orm7q` (server process)
`ora_reco_orm7q` (server process)
`ora_smon_orm7q` (server process)
`oracleorm7q` (shadow process)
`oracleorm7q` (shadow process)
`...`
	`$ ps -ef | grep bfl`
	
	`ora_arc0_bfl` (server process)
	`ora_ckpt_bfl` (server process)
	`ora_d000_bfl` (server process)
	`ora_d001_bfl` (server process)
	`ora_d002_bfl` (server process)
	`ora_dbw0_bfl` (server process)
	`ora_lgwr_bfl` (server process)
	`ora_pmon_bfl` (server process)
	`ora_reco_bfl` (server process)
	`ora_smon_bfl` (server process)
	`ora_snp0_bfl` (server process)
	`oraclebfl` (server process)
	`oraclebfl` (server process)
	`...`

NOTE: A shadow processes is created for each user connecting to the database instance.

To put these two instances into different PRM groups you need to specify alternate names using wildcards:

`ora*orm*`	includes all processes (server & shadow) of the instance ORA_ORM
`ora*bflt`	includes all processes (server & shadow) of the instance ORA_BFL

NOTE: Pattern matching follows the rules outlined in the regexp(5) manual page. Because pattern matching is intended to match a collection of alternate names to a fully qualified executable, PRM was engineered to make sure that other applications and processes matching the pattern were not inadvertently put in the wrong PRM group. PRM does this by comparing filesystem inode numbers. For a process matching the alternate name pattern to be placed in the configured PRM group, it must share the filesystem inode number of the parent process. This is why you should always use the actual executable names, not symbolic links, in application records.

The Oracle database administrator does not have to worry about inode numbers because all processes launched from the same executable, such as Oracle database instances, share the same filesystem inode number.

NOTE: managing **listener** processes

Although listener processes can be associated with a particular database instance, they are actually started as part of the networking processes that run with Oracle. In ps output, they appear as fully qualified executables:

```
oracle  1769     1  0 09:35:02 ?      0:00
/oracle/app/oracle/product/9.0.0/bin/tnslsnr LISTENER_Sales
oracle  1779     1  0 09:35:23 ?      0:00
/oracle/app/oracle/product/9.0.0/bin/tnslsnr LISTENER_Mktg
oracle  1774     1  0 09:35:12 ?      0:00
/oracle/app/oracle/product/9.0.0/bin/tnslsnr LISTENER_Support
```

The three listeners have exactly the same executable name, but with different parameters. The listener process associated with the Sales database cannot be placed in the Sales PRM group by using an application record in the PRM configuration file. To put listener processes in the same PRM group as the database instance, use the following prmmove command with the listener's process ID:

```
# prmmove Sales -p 1769
```

You may want to automate this procedure with a script that runs after all databases are online:

```
#!/usr/bin/sh

# Script to move Oracle listener processes to PRM groups

# Find and move listener process for Sales database
  pid1=$(ps -ef | grep tnslsnr | grep LISTENER_Sales | awk ‘{print $2}’)
  prmmove Sales -p $pid1

# Find listener process for Support database
  pid2=$(ps -ef | grep tnslsnr | grep LISTENER_Support | awk ‘{print $2}’)
  prmmove Support -p $pid2

# Find listener process for Mktg database
  pid3=$(ps -ef | grep tnslsnr | grep LISTENER_Mktg | awk ‘{print $2}’)
  prmmove Mktg -p $pid3
```

The above script does not handle the case in which a database instance has multiple listener processes.

NOTE: If you like to set up an application record for a **shell script**, do it like this:

```
/sbin/sh::::GroupA,foo.sh
```

You must also ensure that the full path for the shell or interpreter for your script is listed in the file `/etc/shells`.

Because the full pathname is not required for the script, a rogue user can get access to PRM groups he may not otherwise have access to by using the name of the script for his own scripts or wrappers.

So we have specified four PRM groups: EXPRESS, OAS, ORA_ORM and ORA_BFL.

Additionally the following two groups exist but do not have to be specified:

PRM_SYS
this group includes all system processes, i.e. all that are started by user root. There are no resource restrictions for this group.

OTHERS
this group includes all processes that are not assigned to any of the groups above.

Resource Entitlement per PRM Group

the command `prmavail(1)` displays a quick information about the available system resources:

```
# prmavail
16 CPUs
8388608 real memory pages or 32768 MB available (PRM estimate)
5 volume groups
        /dev/vg00
        /dev/vgora
        /dev/vgdbt
```

Now you need to think about how to divide the resources (CPU, MEMORY, DISK) among the PRM groups. Basically there are two different approaches:

1) analytic approach

The command `prmanalyze(1)` reads standard HP-UX accounting information and allows an estimate of the resource consumption per process. Therefore process accounting needs to be running for some time to deliver statistically valuable information.

```
$ /usr/sbin/acct/accton /var/adm/pacct
```

switches accounting on

Check if the file `/var/adm/pacct` grows over time.

```
$ /usr/sbin/acct/accton
```

switches accounting off again

```
$ prmanalyze -1 -p
summary CPU report by command name : 10736 records processed
unique id   processes    ave secs   peak secs    total secs  % total
agdbserv            2       98.38      196.01        196.75     1.94
alarmgen            2      378.17      754.16        756.34     7.45
cmcld               2     1453.49     2896.64       2906.98    28.64
dced                2      107.14      213.76        214.27     2.11
mib2agt             2      149.62      298.00        299.24     2.95
midaemon            2      779.75     1554.00       1559.49    15.36
netfmt              2       65.00      129.63        130.01     1.28
opcmona             2      107.27      213.86        214.54     2.11
rep_serv            2      135.79      270.56        271.59     2.68
scopeux             2      243.03      484.40        486.07     4.79
scrdaemo            2       83.74      167.18        167.48     1.65
```

```
snmpdm              2      136.94      272.71        273.87     2.70
```

The `-1` option filters out all processes, that have used less than 1% CPU time.
The `-p` option includes all processes that are currently running.

```
$ prmanalyze -1 -p -r mem
summary memory report by command name : 10301 records processed
unique id   processes      ave KB     peak KB    KB minutes  % total
automoun            2      156.79      158.76  17712288741.64     1.59
biod               32        7.90        9.26  14272290774.21     1.28
cmclconf            9       73.61      115.67  21106092748.33     1.89
cmcld               2      625.02     2119.00  89601638011.65     8.04
cmlvmd              2      242.69      357.00  34790978934.67     3.12
cmsnmpd             2      103.80      117.46  14886363987.98     1.34
dced                2      366.15      382.66  52508918875.98     4.71
disp+wor            8     3603.40     5910.00  85128425600.00     7.64
dmisp               2      155.43      407.00  22289890807.85     2.00
hpuxci              2      266.43      275.70  38207763568.95     3.43
mib2agt             2      102.05      103.89  14634840165.09     1.31
mibmond             2      150.38      169.37  21558326270.47     1.93
midaemon            2      614.04      778.00  88056096650.69     7.90
nfsd               21      122.47      214.00  60223461496.00     5.40
opcctla             2      405.29      506.00  58101720266.34     5.21
opcmsgi             2      110.04      138.67  15775275584.00     1.41
opcuisrv            2      210.32      247.86  30151185915.25     2.70
prm3d               6      857.77     1024.00  73680510360.90     6.61
rep_serv            2      267.61      338.00  38375920535.21     3.44
rpcbind             2      157.79      187.04  22636186149.09     2.03
scopeux             2      389.66      557.00  55879680622.10     5.01
snmpdm              2      205.46      226.95  29465439020.68     2.64
```

```
$ prmanalyze -1 -p -r disk
summary disk report by command name : 1961 records processed
unique id   processes     ave Bps    peak Bps      total KB  % total
diagmond            1      265.43      265.43    8685980.47     3.01
disp+wor            8     8435.80    66413.13  195128029.79    67.67
dmisp               2       21.05       89.43    2948265.72     1.02
instl_bo            1      870.52      870.52   28489607.62     9.88
mib2agt             1      133.84      133.84    4380250.00     1.52
oracle             14      376.39     1338.61   15243801.86     5.29
pwgrd               2       31.71      100.54    4441540.62     1.54
scopeux             2       36.87       56.04    5163305.76     1.79
snmpdm              2       36.31      155.38    5085586.52     1.76
```

You may sort the output by User-ID:

```
$ prmanalyze -p -s uid
summary CPU report by user id : 10736 records processed
unique id   processes    ave secs   peak secs    total secs  % total
root            10610        0.96     2896.64      10146.12    99.95
daemon              1        0.02        0.02          0.02     0.00
n07adm             12        0.19        0.91          2.33     0.02
oran07              4        0.08        0.26          0.32     0.00
awernig            93        0.02        0.13          1.69     0.02
adernier           14        0.03        0.17          0.45     0.00
hosterka            2        0.34        0.64          0.68     0.01
```

Details can be found in the prmanalyze man page or in the PRM Users Guide.

2) empiric approach (recommended)

It is usually a good idea to start with an initial configuration based on estimated entitlements. After monitoring the resource consumption with PRM active for some days this initial configuration may be modified to meet the actual demands.

Because the different resources cannot be isolated from each other, it is recommended not to limit all resources at once but to start with CPU entitlement.

NOTE:
A process that has to wait for memory cannot use up its CPU entitlement.

In this case we start with a configuration as follows:

PRM Group	PRM-ID	CPU	MEM	DISK
OTHERS	1	10%	10%	/
EXPRESS	2	15%	20%	/
ORA_ORM	3	30%	10%	/
ORA_BFL	4	25%	15%	/
OAS	5	20%	15%	/

(PRM-ID 1 is reserved for group OTHERS)

Configuration

Creating the PRM Configuration File /etc/prmconf

The PRM configuration is contained in the ASCII file `/etc/prmconf`.

Use the prmloadconf command to create a template. Specify a file name using the `–f` option. Default file name is `/etc/prmconf`:

```
$ prmloadconf [-f file]
```

This template needs to be modified. In this case the configuration looks as follows (detailed syntax in the table below):

```
$ cat /etc/prmconf

########## PRM group record ##################################
#
# GROUP:{PRMID | HIER}:SHARES::            (for FSS groups)
# GROUP:PSET:::[CPUS]:[CPU_LIST]           (for PSET groups)
#
OTHERS:1:10::
EXPRESS:2:15::
ORA_ORM:3:30::
ORA_BFL:4:25::
OAS:5:20::
#
# in this section the group names, group ids and cpu entitlements are defined


########## PRM memory record ######################################
#
# #!PRM_MEM:{PRMID|GROUP}:SHARES:[CAP]:[SUPPRESS]::[[IMPORT]:[EXPORT]:[LOCKABLE]]
#
#!PRM_MEM:OTHERS:10::::
#!PRM_MEM:EXPRESS:20::::
#!PRM_MEM:ORA_ORM:10::::
#!PRM_MEM:ORA_BFL:15::::
#!PRM_MEM:OAS:15::::
#
# in this section the memory entitlement are defined


########## PRM application record ##################################
#
# APPLICATION::::GROUP[,ALTERNATE_NAME[, ...]]
#
/usr/contrib/bin/perl::::PRM_SYS
/oracle/app/oracle/product/8.1.5/bin/oracle::::ORA_BFL,ora*bfl
/oracle/app/oracle/product/8.0.6/bin/oracle::::ORA_ORM,ora*orm*
#
# in this section the executables and their sub-processes are
# assigned to PRM groups


########## PRM user record ########################################
#
# USER::::INITIAL_GROUP[,ALTERNATE_GROUP[, ...]]
#
dsuser::::PRM_SYS
```

```
oesdba::::EXPRESS
oesinit::::EXPRESS
oesguest::::EXPRESS
oraias::::OAS
#
# in this section the users are assigned to the PRM groups
# All other users from /etc/passwd, that are not explicitely specified
# here will be assigned to group OTHERS automatically. The root user
# always belongs to PRM_SYS group


########## PRM disk record ########################################
#
# VOLUME:{PRMID | GROUP}:SHARES::
#
# in this section the disk bandwidth is specified per volume group
# (not in this example)
```

Syntax for prmconf. See also prmconf man page

Group record:

GROUP:{PRMID|HIER}:SHARES:	(FSS group record)
GROUP	FSS PRM group name. For proper display by the ps command, use no more than seven characters in the PRM group name.
PRMID|HIER	Parent nodes in a hierarchy use the HIER keyword, while leaf nodes use PRM group ID. PRMIDs must be unique and can range from 1 to 63. PRMID 1 (the default group for users not explicitly assigned to a group) is required.
SHARES	CPU shares and, if CPU capping is enabled, the capping value. Share amounts can range from 1 to MAXINT. Shares determine the resource allocation proportions for all children of the same parent.
GROUP:PSET:::[CPUS]:[CPU_LIST]	(PSET group record)
PSET	The PSET keyword must be specified for PSET PRM group records.
CPUS	The number of CPUs allocated to this PSET PRM group. This value may range from 0 to one less than the maximum number of CPUs on a system. If blank, the value is determined from the number of CPUs specified in CPU_LIST.
CPU_LIST	Comma separated list of specific CPU IDs to include in the PSET PRM group. If this list is empty then the PRM API will automatically choose the CPUs to include in the PSET PRM group. If CPUS is specified, then CPU_LIST must contain zero CPU entries or the number of entries specified by CPUS.

Memory record:

#!PRM_MEM:{PRMID|GROUP}:SHARES:[CAP]:[SUPPRESS]::[[IMPORT]:[EXPORT]:[LOCKABLE]]	
GROUP	PRM group ID or group name that corresponds to an existing PRM group. When specifying parents in a PRM group hierarchy or a PRM PSET group, use their names. If Memory records are used, there must be a one-to-one match between PRMIDs in Group and Memory records.
SHARES	Specifies the PRM group's guaranteed proportion of available memory. Shares are integer values ranging from one to MAXINT.
CAP	(optional, default is no cap limit) Specifies a cap (upper bound) for memory

	consumption. The cap value is expressed as a percent, which is an integer value, ranging from the percentage determined by the group's number of memory shares to 100. There is no requirement that the cap values total 100%. You can not specify values for both the CAP and the IMPORT/EXPORT/LOCKABLE fields.
SUPPRESS	(optional) (Default is ALL) Indicates how to select processes for suppression when the group is overusing the memory resource. Valid keywords are: ALL Suppress all processes in the group. LARGEST To suppress the largest processes in the group first and then progress to smaller processes, as needed. Note: This field is ignored when using prm2d memory manager (the default on HP-UX 11i).
IMPORT	(optional) (Default is no import limit) Used for memory isolation of a group. Leave this field blank to disable the memory isolation feature. Assign the value 0 to both the IMPORT field and the EXPORT field if you would like to isolate a memory critical group to ensure it does not share memory with other groups. The IMPORT field is supported on HP-UX version 11i and above.
EXPORT	(optional) (Default is no export limit) Used for memory isolation of a group. Leave this field blank to disable the memory isolation feature. Assign the value 0 to both the IMPORT field and the EXPORT field if you would like to isolate a memory critical group to ensure it does not share memory with other groups. The EXPORT field is supported on HP-UX version 11i and above.
LOCKABLE	(optional) (Default is no lockable limit) Portion in shares of available memory that may be locked. Valid range is from 0 to the number of shares defined in the memory record. You can not use the CAP field if the LOCKABLE or SUPPRESS field is used. The LOCKABLE field is supported on HP-UX version 11i and above.

NOTE: When specifying memory records for parents in a PRM group hierarchy, you can not specify values for the following fields (CAP, IMPORT, EXPORT, LOCKABLE).

User record

USER:::INITIAL_GROUP[,ALTERNATE_GROUP[, ...]]	
USER	This is either an individual user's login name, or a + character followed by netgroup name. Login names should be found in /etc/passwd. Netgroup names should be associated with a list of login names in /etc/netgroup. At configuration time, any member of this netgroup without an explicit user record will assume the INITIAL and ALTERNATE groups from this record.
INITIAL_GROUP	Name of user’s initial PRM group upon login.
ALTERNATE_GROUP	(optional) Name of another group in which this user has permission to run. Group may me changed from the INITIAL_GROUP to ALTERNATE_GROUP by means of prmrun or prmmove commands. Note: Only the root user can have PRM_SYS as an initial or alternate group

NOTE: PRM User records other than root cannot contain the PRMID 0 group PRM_SYS.

Application record

APPLICATION::::GROUP[,ALTERNATE_NAME[, ...]]	
APPLICATION	Full path of the application, starting with a slash (/). The directory name must be fully qualified, but the file name can include the shell file name wildcard characters *, ?, [, and]. At configuration time, all valid executables matching the expression without explicit application records of their own will assume the GROUP of this record.
GROUP	Name of the application's assigned PRM group. Is the name of the PRM group in which the application will run. Applications cannot be assigned to parent groups.
ALTERNATE_NAME	(optional) Is an alternate name that the application assigns to itself at runtime. Processes running the same executable can be given different argv[0] names when they are executed. The argv[0] name is the process name that appears in ps(1) output. Application records with altnames match processes that have an argv[0] that matches ALT_NAME and are running APPLICATION. This is most common for database programs that launch multiple processes from the same executable and rename them. Using alternate names, you can place the various processes that are running the same executable in different PRM groups. Alternate names can also be used to distinguish between processes executed via different symbolic links to the same binary. Pattern matching notation can be used to designate a group of similarly named processes.

Disk record

VOLUME:{PRMID | GROUP}:SHARES::	
VOLUME	Full pathname of the desired logical volume beginning with /dev/v
PRMID | GROUP	PRM group ID or PRM group name. If Disk records are used for a volume, there must be a one-to-one match between CPU and Disk records.
SHARES	Disk shares. Share amounts can range from 0 to MAXINT. Shares determine the resource allocation proportions for all children of the same parent.

NOTE: By default, PRM gives PRM_SYS 100 CPU shares. If you assign 100 shares to the PRM groups you create, PRM_SYS gets 50% (100/200) of the CPU. The PRM_SYS group must get at least 20% of the CPU. Thus, if you assign more than 400 shares to your groups, the total shares assigned is greater than 500, and the PRM_SYS group’s 100 shares do not represent at least 20%. In this case, PRM scales the shares for your groups proportionately so they are less than or equal to 400 shares.

You can explicitly add the PRM_SYS (PRMID 0) group to a configuration file. However, if you explicitly add the PRM_SYS group to a configuration file, it gets the CPU shares you assign it, which must equate to at least 20%.

prmmonitor -s includes the PRM_SYS group in output.

Testing the Configuration

Use the `-s` option of `prmconfig(1)` in order to detect syntax errors in `/etc/prmconf`.

```
$ prmconfig –s
Configuration file check complete. No errors found.
```

Configuring the Startup Script

The default startup configuration is as follows:

```
$ cat /etc/rc.config.d/prm

#!/sbin/sh
#
# PRM configuration.  See prmconfig(1)
#
# To configure (and enable) PRM automatically at boot time, PRM must have
# been previously configured.  The most recent configuration will be used,
# that is the configuration in place prior to system shutdown or before the
# most recent prmconfig -r (reset).
#
# PRM_CONFIG:   Set to 1 to configure PRM
# PRM_ENABLE:   Set to 1 to enable PRM
# PRM_CAPPING:  Set to 1 to start PRM in CPU capping mode when enabled
# PRM_SNMPAGT:  Set to 1, always spawn prmagt at boot time
# PRM_RMTCONF:  Set to 1, always spawn remote config daemon at boot time
#
# NOTE:  in order to use PRM_ENABLE=1, you must set PRM_CONFIG=1
# NOTE:  in order to use PRM_CAPPING=1, you must set PRM_ENABLE=1
#
PRM_CONFIG=0
PRM_ENABLE=0
PRM_CAPPING=0
PRM_SNMPAGT=0
PRM_RMTCONF=1
```

If you want that PRM is enabled after a reboot you need to set `PRM_CONFIG`, `PRM_ENABLE` to 1. If you want PRM to run in capping mode, set `PRM_CAPPING` to1 additionally.

Activation

Starting and Stopping PRM

The command `prmconfig(1)` activates PRM.

```
$ prmconfig -i -e APPL
```

`-i` this option assigns the currently running processes into their groups based on the configuration in `/etc/prmconf`. The PRM resource managers CPU, MEM and DISK are not yet in effect, i.e. no process will be surpressed by PRM.

`-e` this option enables the application manager APPL that will continue to assign new processes into their groups. If not enabled then new processes won't be moved into their correct groups. There is no 1:1 correspondance between APPL manager and application records. You need APPL even if you did not specify application records.

The new `-P` option of the `ps(1)` command displays the PRM group and allows you to check if the configuration has been applied properly:

```
$ ps -efP | grep ORA_ORM

  kdoi22 ORA_ORM  14649     1  0 19:26:06 ?         0:01 oracleormq (LOCAL=NO)
  kdoi22 ORA_ORM   5872     1  0  Jan 10  ?         0:25 ora_smon_ormt
  kdoi22 ORA_ORM   5870     1  0  Jan 10  ?         5:33 ora_ckpt_ormt
  kdoi22 ORA_ORM   5868     1  4  Jan 10  ?         4:24 ora_lgwr_ormt
  kdoi22 ORA_ORM   5851     1  0  Jan 10  ?         2:10 ora_pmon_ormt
  kdoi22 ORA_ORM   5864     1  0  Jan 10  ?         0:00 ora_arch_ormt
  kdoi22 ORA_ORM   5874     1  0  Jan 10  ?         0:02 ora_reco_ormt
  kdoi22 ORA_ORM   5861     1  2  Jan 10  ?         2:55 ora_dbw0_ormt
  kdoi22 ORA_ORM   6059     1  0  Jan 10  ?         2:16 ora_dbw0_ormq
  kdoi22 ORA_ORM   6057     1  0  Jan 10  ?         2:13 ora_pmon_ormq
  kdoi22 ORA_ORM   6069     1  0  Jan 10  ?         5:41 ora_ckpt_ormq
  kdoi22 ORA_ORM   6062     1  0  Jan 10  ?         0:00 ora_arch_ormq
  kdoi22 ORA_ORM   6071     1  0  Jan 10  ?         0:20 ora_smon_ormq
  kdoi22 ORA_ORM   6066     1  0  Jan 10  ?         2:48 ora_lgwr_ormq
  kdoi22 ORA_ORM  14679     1  0 19:26:07 ?         0:18 oracleormq (LOCAL=NO)
  kdoi22 ORA_ORM   6073     1  0  Jan 10  ?         0:01 ora_reco_ormq
  kdoi22 ORA_ORM   2817     1  0 14:00:57 ?         0:00 oracleormt (LOCAL=NO)
  kdoi22 ORA_ORM  14543     1  0 19:26:02 ?         0:05 oracleormq (LOCAL=NO)
  ...
  ...
```

The PRM group is shown in the second column.

The command

```
$ prmconfig –e
```

finally enables **all** configured PRM resource managers (CPU, APPL, MEM, DISK).

You may do initialization and enablement in one single step:

```
$ prmconfig –ie
```

ATTENTION:

> If you like to activate the CPU capping feature (i.e. unused CPU time will not be given to PRM groups that like to go beyond their entitlement), you need to specify the `-M` option with prmconfig:
>
> ```
> $ prmconfig –ie –M CPUCAPON
> ```
>
> Details regarding the capping feature can be found in the PRM Users Guide.

The commands `prmconfig(1)` and/or `prmlist(1)` let you display the PRM groups:

```
# prmconfig

PRM configured from file:  /etc/prmconf
File last modified:        Thu Mar  1 17:22:57 2001

PRM CPU scheduler state:  Enabled

PRM Group                      PRMID    CPU Entitlement
-------------------------------------------------------
OTHERS                             1             55.56%
SETI                               3              2.78%
WWW                                2             41.67%

PRM memory manager state:  Not Running

PRM User                       Initial Group                  Alternate Group(s)
-------------------------------------------------------------------------------
ohaensel                       SETI
root                           PRM_SYS
www                            WWW

PRM application manager state:  Enabled  (polling interval: 30 seconds)

Disk manager state:  Disabled


# prmlist

PRM configured from file:  /etc/prmconf
File last modified:        Thu Mar  1 17:22:57 2001

PRM Group                      PRMID    CPU Entitlement
-------------------------------------------------------
OTHERS                             1             55.56%
SETI                               3              2.78%
WWW                                2             41.67%

PRM User                       Initial Group                  Alternate Group(s)
-------------------------------------------------------------------------------
ohaensel                       SETI
root                           PRM_SYS
www                            WWW

PRM Application                Assigned Group                 Alternate Name(s)
-------------------------------------------------------------------------------
/home/ohaensel/seti            SETI
/home/ohaensel/seti            SETI
/home/ohaensel/seti            SETI
```

You may activate logging for the PRM resource managers APPL and/or MEM. Status messages will be written to `/var/adm/syslog/syslog.log`, e.g.:

```
# prmconfig -L APPL
```

```
# tail /var/adm/syslog/syslog.log
...
...
Jul 11 19:56:27 grcdg455 HP-PRM: [7639]: prm1d: Application manager has been
enabled
Jul 11 19:56:32 grcdg455 HP-PRM: [7639]: prm1d: Application manager polling
interval is 30 seconds.
Jul 11 20:31:57 grcdg455 HP-PRM: [7639]: prm1d: Application manager logging is ON
Jul 11 20:33:12 grcdg455 HP-PRM: [7639]: prm1d: moved pgrp 7654 (top) to PRMID 20
```

To switch logging off do:

```
# prmconfig -L APPL STOP
```

Monitoring

Monitoring the Resource Consumption

You can monitor and verify your PRM configuration with a number of commands:

- `prmmonitor`
- `prmconfig`
- `prmlist`
- `id -P`
- `ps [ -P ][ -R group_list ]`
- `acctcom [ -P ][ -R group ]`
- `prmanalyze`
- Glance
- Measureware
- Servicecontrol Manager tools

The commands `prmonitor(1)` and/or `prmlist(1)` report the current resource consumption of the PRM groups:

```
# prmmonitor

PRM configured from file:  /etc/prmconf
File last modified:        Thu Mar  1 17:22:57 2002

HP-UX grcdg071 B.11.11 U 9000/800    03/05/01

Mon Mar  5 09:24:27 2002    Sample:  1 second
CPU scheduler state:  Enabled
                                                CPU      CPU
PRM Group                       PRMID   Entitlement     Used
____________________________________________________________
OTHERS                              1        55.56%    0.00%
WWW                                 2        41.67%    0.00%
SETI                                3         2.78%    0.00%

PRM application manager state:  Enabled  (polling interval: 30 seconds)
```

```
# prmanalyze -s prmid
summary CPU report by PRM id : 56436 records processed
unique id   processes    ave secs   peak secs    total secs  % total
  0             54220        0.43     7888.64      23465.12    98.01
  1              2209        0.22      161.42        475.84     1.99
  2                 7        0.05        0.06          0.35     0.00
```

This should help you to locate potential performance problems on the one hand or to finetune the configuration on the other hand.

Glance and Measureware are also aware of PRM groups. In Glance you can access the PRM screen by pressing key ‘P’.

NOTE: Glance does not correctly track the PRM ID at the process level for HP-UX 11i in versions C.02.65.00 through C.03.25.00. For correct metrics reporting for FSS PRM groups, use Glance version C.03.35.00 or later.

Modifying the Configuration

If it seems that the chosen entitlements need to be tuned, then the PRM configuration can be changed online, i.e without stopping any processes. After modifying the configuration file `/etc/prmconf` the changes can be applied immediately.

`# vi /etc/prmconf`	modify the current configuration
`# prmconfig –r`	reset/disable PRM
`# prmconfig –ie`	initialize and enable PRM again

It is also possible to change a configuration using cron jobs.

Appendix

PRM Manual Pages

The following manual pages/commands are available with PRM:

`prm(1)`	PRM introduction
`xprm(1)`	PRM graphical user interface. xprm allows modifying PRM configurations without manually editing configuration files. Also, xprm provides a graphical means to perform many of the tasks available with the prmconfig command.
`prmanalyze(1)`	Utility that uses the HP-UX system accounting log to generate PRM group usage statistics. This is useful for configuration planning, monitoring, and billing.
`prmavail(1)`	Displays resource availability to help plan PRM configurations.
`prmconfig(1)`	Configures, enables, disables and resets PRM. Also when used with no options, validates PRM configuration files and controls message logging. You can also perform these tasks with xprm.
`prmlist(1)`	Displays the current PRM group, memory, user, application, and disk configuration information.
`prmloadconf(1)`	Creates a PRM configuration file or updates an existing configuration file according to user records defined in /etc/passwd.
`prmmonitor(1)`	Displays configuration and resource usage by PRM group.
`prmmove(1)`	Move processes or groups of processes into a PRM group.
`prmrecover(1)`	Reactivates processes suppressed by PRM's memory daemon prm0d on UX 10.20 and UX 11.00. As of HP-UX 11.11 it restores the memory manager to its default state by removing the MRG groups created by prm2d.
`prmrun(1)`	Runs an application in its assigned group or a specified group.
`prmconf(4)`	Explains the syntax of the PRM configuration files /etc/prmconf.

PRM File Locations

All PRM files are under `/opt/prm/`:

`/opt/prm/bin/`	PRM utilities and Java Runtime Environment for xprm
`/opt/prm/conf/`	Directory for storing PRM configuration files
`/opt/prm/doc/`	PDF and PostScript versions of the PRM User's Guide

/opt/prm/exempt	File for specifying applications you do not want the PRM memory manager to suppress (available on HP-UX 10.20 and HP-UX 11.0)
/opt/prm/help/C/	English version of xprm online help files. The xprm online help files are available only in english.
/opt/prm/lib/	PRM API libraries
/opt/prm/lib/C/	xprm related files
/opt/prm/man/	Manual pages
/opt/prm/newconfig/RelNotes/	Release notes in PostScript and text

PRM Quick Setup

This guide enables you to quickly set up PRM:

# prmavail	(information about # of CPUs, RAM, VGs)
# prmanalyze	(turn accounting on before: /usr/sbin/acct/accton /var/adm/pacct)
# prmloadconf [-f /etc/prmconf]	(create PRM configuration file)
# vi /etc/prmconf	(customize it)
# prmconfig –s	(check syntax)
# prmconfig –ie APPL	(initialize, i.e. assign running processes/applications to group)
# prmconfig -e [CPU]	(enable)
# ps -efP	(verify)
# prmconfig -d	(disable)
# vi /etc/prmconf	(modify the configuration)
# prmconfig -ie APPL	(takes effect)
# prmconfig -M CPUCAPON	(turn CPU capping mode on)
# vi /etc/rc.config.d/prm	(prmconfig -i /-e / CPUCAPON at startup or not)

CPU Hogger Source Code

the following C program is suited to test the functionality of PRM’s CPU resource manager. It simply tries to consume as much CPU time as it can get.

```
# cat cpuhogger.c

main()
{
   for (;;);
}
```

NOTE: A process can only execute on one CPU at a time! Hence a single cpuhogger process can never use e.g. 50% CPU time on an 8-way system. the maximum would be 12,5%. In order to exploit the CPU time of a PRM group you may need to start more than one cpuhogger process.

New Features in PRM Version C.02.xx

PRM C.02.00 is for systems running UX 11.00 or greater and has the following new features:

In-kernel memory management (HP-UX 11i only)

On HP-UX 11i, memory is controlled in the kernel, through the prm2d daemon, rather than in user space, through the prm0d daemon. Additional fields for IMPORT, EXPORT and LOCKABLE values have been added.
The syntax for a memory record now looks like the following:

```
#!PRM_MEM:{ PRMID |
GROUP}:SHARES:[CAP]:[SUPPRESS]::[IMPORT]:[EXPORT]:[LOCKABLE]
```

Running HP Process Resource Manager on systems with HP-UX 11i and the prm2d memory manager (the default for HP-UX 11i), is strongly recommended.

Processor sets and PSET PRM groups (HP-UX 11i only)

Processor sets (PSET) allow processors to be grouped together in a set by the system administrator. The default PSET initially contains all the processors in the system and is set up at system initialization time. In PRM Version C.02.00, the traditional PRM group uses the Fair Share Scheduler (FSS) in the HP-UX kernel within the system's default processor set. This type of PRM group is now referred to as an FSS PRM group. PRM Version C.02.00 introduces a new type of PRM group that is based on processor sets, called a PSET PRM group. A PSET PRM group is a PRM group whose CPU entitlement is specified by assigning it a subset of the system's processors. Processes in a PSET have equal access to CPU cycles on their assigned CPUs through the HP-UX standard scheduler. PSETs can be installed independently from PRM. The product can be downloaded free of purchase from http://software.hp.com. There is also a CLI if you like to use PSETS without PRM: Refer to the `psrset(1m)` man page for details.

The PSET bundle (`PROCSETS`) contains of a ProcessorSets product and a set of enablement patches. The ProcessorSets product contains the pset library and the psrset command:

```
# ProcessorSets            A.01.00.00.06  HP-UX Processor Sets Product
  # ProcessorSets.PSET-KRN   A.01.00.00.06  Processor Sets Kernel Files
    ProcessorSets.PSET-KRN: /usr/conf/lib/libpset.a
    ProcessorSets.PSET-KRN: /usr/conf/master.d/pset
  # ProcessorSets.PSET-RUN   A.01.00.00.06  Processor Sets User Space Commands
    ProcessorSets.PSET-RUN: /usr/lib/nls/msg/C/psrset.cat
    ProcessorSets.PSET-RUN: /usr/sbin/psrset
```

The installation of the bundle requires a reboot.

Here's an example prmconf group record that uses PSET groups. The system has 6 processors (0,1,..,5)

```
# Group/CPU records
#
OTHERS:1:20::
SALES:2:50::
DEVELOPMENT:3:30::
WEB:PSET:::2:4,5
```

Syntax is:

`GROUP:{PRMID | HIER}:SHARES::`	for FSS groups
`GROUP:PSET:::[CPUS]:[CPU_LIST]`	for PSET groups

For details see /etc/prmconf above.

The group WEB is a PSET group that uses 2 processors exclusivley: processors 4 and 5. The other groups are standard FSS groups. They are running on the remaining processors (0,1,2 and 3) only. The entitlement shares refer to 4 processors not to 6.

Processor 0 cannot be reassigned to another PSET. All other processors can be reassigned in and out of the default PSET.
PSET groups are treated as part of PRM_SYS (PRMID 0) for disk-bandwidth purposes.
Processes in PSET PRM groups have equal access to CPU cycles through the standard HP-UX timeshare scheduler.
This method of isolating applications and users effectively creates a partition on your system.

Memory isolation for PRM groups (HP-UX 11i only)

In PRM Version C.02.00, in addition to specifying memory shares, the prm2d memory manager allows you to optionally specify a group's memory resources to be restricted from use from other groups and processes on the system. This type of restriction is called memory isolation. When a group's memory shares are isolated, those memory shares cannot be loaned out to or borrowed from other groups. This is helpful if a group required dedicated, on-demand memory resources.

Proportional overachievement (HP-UX 11i only)

In PRM Version C.02.00, the concept of proportional overachievement is introduced. If a group is exceeding its memory shares on system that is under stress, prm2d uses proportional overachievement logic to determine which groups need their import shares reduced. Overachievement for a group is the ratio of memory used to memory entitlement. This value is then compared to the average overachievement of all groups. If a PRM group is overachieving compared to the average, then the import shares for that group are lowered. This allows other groups to start importing the newly available memory.

New functionality for `prmanalyze`

The prmanalyze command has been improved in PRM Version C.02.00 to include three new pieces of functionality:

`-s auto`

> Specifies how to sort the accounting data. The use of auto will automatically make recommendations for both user an application records based on the history of your system. Examination of results by PRMID can help to evaluate the effectiveness of a particular strategy once it has been implemented.

`-f config_file`

> This tells the analysis not to take the PRMID from the accounting file, but to compute it using the rules in the specified configuration file. This can be used to show how usage and conflict patterns might have changed under a new or proposed configuration.

`-m minimum_duration`

Specifies a minimum job duration in seconds for inclusion in reports. The majority of interactive HP-UX jobs such as ps and ls will complete in under one second. This feature may be used to filter out transient noise from interactive users and concentrate on longer running applications important enough to have their own application records.

ServiceControl manager tools for PRM

SCM provides a single point of administration for multiple HP-UX systems. The following lists the PRM tools available to SCM on the nodes in the SCM cluster that have PRM installed:

- Display resource usage
- List resource availability
- Start PRM GUI
- Subtool PRM GUI

For a detailed description of new features refer to the PRM release notes.

Workload Manager (WLM)

HP-UX Workload Manager (WLM) supplements the functionality of HP Process Resource Manager (PRM) by offering automatic resource allocation and dynamic application performance management through prioritized service-level objectives (SLOs) – based on goal-based, time-based, condition-based or usage-based criteria.

What’s the difference?

- PRM is entitlement-based in its management of CPU, memory and I/O resources
- WLM utilizes pre-defined policies and goals to facilitate the dynamic allocation of resources based on conditions such as time of day, application responsiveness, business priorities, etc.
- WLM can work across virtual partitions on the same machine, and can dynamically re-allocate resources between these partitions

Latest release of WLM is

```
# swlist -l bundle -s swdepot:/depot/darts/65.Jun04/11.11 | grep -i workl
  B8843CA              A.02.02        HP-UX Workload Manager
  T1302AA              A.01.05        HP-UX Workload Manager Toolkits
  WLMUtilities         A.02.02        HP-UX Workload Manager Utilities
```

Refer to http://hp.com/go/wlm for further information about WLM.

Additional Information

The starting point for PRM documentation is
http://hp.com/go/prm

This website links to all information about PRM such as:

- PRM User's Guide (explains how PRM manages resources)
- PRM Release Notes
- PRM product overview
- Integrating PRM under IT/Operations
- Using HP PRM with Oracle Databases
- PRM Oracle/TPC-C Evaluation
- Oracle Database Benchmark with HP PRM Memory Management

Expert Center website:
http://wtec.cup.hp.com/~hpuxha/products/prm.html (HP internal)

Lab website:
http://pasl.rsn.hp.com/prm/ (HP internal)