



### Exercise 20.2 Examining System V IPC Activity

**System V IPC** is a rather old method of Inter Process Communication that dates back to the early days of **UNIX**. It involves three mechanisms:

1. **Shared Memory Segments**
2. **Semaphores**
3. **Message Queues**

More modern programs tend to use **POSIX IPC** methods for all three of these mechanisms, but there are still plenty of **System V IPC** applications found in the wild.

To get an overall summary of **System V IPC** activity on your system, do:

```
$ ipcs
```

```
----- Message Queues -----
key      msqid      owner      perms      used-bytes  messages

----- Shared Memory Segments -----
key      shmid      owner      perms      bytes       nattch     status

0x01114703 0          root       600        1000        6          dest
0x00000000 98305      coop       600        4194304     2          dest
0x00000000 196610     coop       600        4194304     2          dest
0x00000000 23068675   coop       700        1138176     2          dest
0x00000000 23101444   coop       600        393216      2          dest
0x00000000 23134213   coop       600        524288      2          dest
0x00000000 24051718   coop       600        393216      2          dest
0x00000000 23756807   coop       600        524288      2          dest
0x00000000 24018952   coop       600        67108864    2          dest
0x00000000 23363593   coop       700        95408       2          dest
0x00000000 1441811    coop       600        2097152     2          dest

----- Semaphore Arrays -----
key      semid      owner      perms      nsems

0x00000000 98304      apache     600        1
0x00000000 131073     apache     600        1
0x00000000 163842     apache     600        1
0x00000000 196611     apache     600        1
0x00000000 229380     apache     600        1
```

Note almost all of the currently running shared memory segments have a key of 0 (also known as `IPC_PRIVATE`) which means they are only shared between processes in a parent/child relationship. Furthermore, all but one are marked for destruction when there are no further attachments.

One can gain further information about the processes that have created the segments and last attached to them with:

```
$ ipcs -p
```

```
----- Message Queues PIDs -----
msqid      owner      lspid      lrpids

----- Shared Memory Creator/Last-op PIDs -----
shmid      owner      cpid      lpids
0          root       1023      1023
98305      coop       2265      18780
196610     coop       2138      18775
23068675   coop       989       1663
```

```

23101444  coop      989      1663
23134213  coop      989      1663
24051718  coop     20573     1663
23756807  coop     10735     1663
24018952  coop     17875     1663
23363593  coop      989      1663
1441811   coop     2048     20573

```

Thus, by doing:

```
$ ps aux |grep -e 20573 -e 2048
```

```

coop      2048  5.3  3.7 1922996 305660 ?      Rl   Oct27  77:07 /usr/bin/gnome-shell
coop      20573 1.9  1.7 807944 141688 ?      Sl   09:56   0:01 /usr/lib64/thunderbird/thunderbird
coop      20710 0.0  0.0 112652  2312 pts/0    S+   09:57   0:00 grep --color=auto -e 20573 -e 2048

```

we see **thunderbird** is using a shared memory segment created by **gnome-shell**.

Perform these steps on your system and identify the various resources being used and by who. Are there any potential **leaks** (shared resources no longer being used by any active processes) on the system? For example, doing:

```
$ ipcs
```

```

....
----- Shared Memory Segments -----
key          shmid      owner      perms      bytes      nattch     status
....
0x00000000  622601     coop      600        2097152    2          dest
0x0000001a  13303818  coop      666        8196       0
....

```

shows a shared memory segment with no attachments and not marked for destruction. Thus it might persist forever, leaking memory if no subsequent process attaches to it.