System V IPC Parameters

Last Update: 20.07.2001 (*change log*)

Due to the extend of possible System-V IPC related option I chose to put them into a separate document, the one you are currently reading.

You can watch your current System-V IPC configuration with the help of the `sysdef -i` command. But do not be confused by the output, the necessary modules for IPC are loaded dynamically into the kernel on demand. The `sysdef` command does not forceload the necessary modules, but you may want to try a litte program of mine which references each module, thus loading them into the kernel.

After all modules are loaded into the kernel they stay loaded. Now you can obtain reliable output regarding System-V IPC from the output of the `sysdef -i` command.

1. Message queues

Message queues use the prefix `msgsys:msginfo_` for their respecive entries into the `/etc/system` file. I obtained the first set of default values from `sysdef -i`. The second set of default values are obtained from the header file `/usr/include/sys/msg.h`. **Beware** to exceed the header file hardcoded values. Do it only if you know what you are doing. The set of *typical* values are taken from Stevens' APUE.

Cockroft describes the message queue tunable parameters without the `msginfo_` prefix. If you have troubles with your message queue tuning, try without the prefix for these parameters. The value of 2147483647 is of theoretical impact, not to be used in production systems. More information (for programmers) can be found in the 11'97 SunWorld article Demangling message queues by Jim Mauro.

msgsys:msginfo_msgmap

default 100, max 2147483647, `<sys/msg.h>` 100

The number of entries in a message map, an initialized and pre-allocated resource in the kernel.

msgsys:msginfo_msgmax

default 2048, max 2147483647, `<sys/msg.h>` 8192, typical value 2048

The size in byte of the largest message which can be sent down a message queue. The system will not allow message which exceed this value.

msgsys:msginfo_msgmnb

default 4096, max 2147483647, `<sys/msg.h>` 2048, typical value 4096

The maximum size in byte of a message queue, that is, the sum of all messagessizes within the queue. This value is allocated when the message queue is created, see `msgget(2)`. Only process with UID 0 (root) can increase this value from a program.

msgsys:msginfo_msgmni

default 50, max 2147483647, `<sys/msg.h>` 50, typical value 50

The maximum number of message queues, systemwide. Some call them *identifiers*, some call them *handles*.

msgsys:msginfo_msgssz

default 8, max 2147483647, `<sys/msg.h>` 8

Size of the message segment - should be a multiple of the wordsize.

msgsys:msginfo_msgtql

default 40, max 2147483647, `<sys/msg.h>` 50, typical value 40

The maximum number of messages, systemwide, all queues. Programmatically, this value reflects the number of message headers maintained by the kernel. Each message in any message queue requires a message header.

msgsys:msginfo_msgseg>/A>

default 1024, max 32767, `<sys/msg.h>` 1024

Maximum number of message segments.

2. Semaphores

Semaphores are introduced by the prefix `semsys:seminfo_` in the file `/etc/system`. The default set was determined with the help of `sysdef -i`. Typical values were taken from Stevens' documentation in APUE. Please note that System V semaphore cannot exist by themselves. They are always grouped into a semaphore set, even if the set only contains a single semaphore.

There are three sets of semaphores bundled with Solaris. We are talking about the System V interprocess communication semaphores here. The value of 2147483648 is of theoretical impact, not to be used in production systems. More information (for programmers) can be found in the 10'97 SunWorld article

Setting our sights on semaphores by Jim Mauro.

semsys:seminfo_semmap

default 10, min 10, max 2147483647, recommended **semmsl*semmni** (doubtful)

Maximum number of entries in the semaphore map. If you create or use a semphore in a set, the kernel will actually allocate memory chunks for this many semaphores. Thus it avoid unnecessary memory operations with each semaphore call.

semsys:seminfo_semmni

default 10, typical value 10, max 65535, recommended <= **semmap**

The maximum number of semaphore sets, systemwide - Mauro and Cockroft call them *identifiers*, other refer as *handles*.

During initialization, the system allocates kernel memory for **semmni** control structures. Each control structure is 84 bytes, thus you should avoid arbitrarily large values.

semsys:seminfo_semmns

default 60, typical value 60, max 2147483647, recommended **semmni*semmsl**

The maximum number of semaphores in all sets, systemwide. Since **semmni** is the maximum number of possible sets, and **semmsl** the maximum number of semaphores per set, the maximum number of semaphores system-wide can never be greater than the product. It should be calculated automatically, though it is not.

semsys:seminfo_semmnu

default 30, typical value 30, max 2147483647, recommended **semmni**

The maximum number of *undo* structures, systemwide. If you use the value of **semmni**, you would provide an undo structure for each semaphore set. Each undo structure uses 16 bytes. Using **semmni***semmsl** would provide an undo structure for each possible semaphore - probably an overkill.

semsys:seminfo_semmsl

default 25, typical value 25, max 65535

The maximum number of semaphores in a set.

semsys:seminfo_semopm

default 10, typical value 10, max 2147483648, recommended **semmsl**

The maximum number of operations per `semop(2)` call. This goes back to the notion of doing several operations on different semaphore within the same set. The `sempop(2)` checks the array size of itsthird arguments against this value.

semsys:seminfo_semume

default 10, typical value 10, max 2147483648

The maximum number of *undo* operations per process.

The kernel maintains the semaphore adjustments made by a process. In case of a premature exit, the kernel can readjust the semaphore values to their previous value.

semsys:seminfo_semusz

default 96 = sizeof(undo)+**semume***sizeof(undo), max 2147483648, recommended: don't touch

The size in bytes of the undo structure - this should not be a tunable.

semsys:seminfo_semvmx

default 32767, typical value 32767, max 65535

The maximum value a semaphore can have.

semsys:seminfo_semaem

default 16384, typical value 16384, max 32767

The maximum count a semaphore can be adjusted if the respective process dies.

3. Shared Memory

Shared memory parameters are introduced by the prefix `shmsys:shminfo_` in `/etc/system`. The default set was determined with the help of `sysdef -i`. Typical values were taken from Stevens' documentation in APUE.

The maximum values shown are really limited by the maximum main memory size. More information (for programmers) can be found in the 09'97 SunWorld article Shared memory uncovered by Jim Mauro.

shmsys:shminfo_shmmax

default 1048576, typical value 131072, min 1, max 4294967295, recommended 0 < x <= 16777216

The maximum size in byte of a single shared memory segment.

shmsys:shminfo_shmmin

default 1, typical value 1, min 1, max 4294967295

The minimum size in byte of a single shared memory segment.

shmsys:shminfo_shmmni

default 100, typical value 100, min 1, max 2147483648

The kernel maintains a table of a datastructure associated with each shared memory segment. Whenever the `shmget(2)` system call is completed successfully, one such association is built. The current parameter specifies the maximum number of such associations between a shared memory segment (some would call it *identifier* or even *handle*) and the actual shared memory, systemwide.

shmsys:shminfo_shmseg

default 6, typical value 6, min 1, max 32767

The maximum number of shared memory segments (identifiers, handles) per process.

[Back] [Solaris tuning] [TCP transactions] [SYS-V-IPC] [TCP rexmit] [Slow start] [Index] [JSV homepage]

Please send your suggestions, bugfixes, comments, and ideas for new items to solaris@sean.de

In hope of supplying useful information, Jens-S. Vöckler

Last Modified: Friday, 20-Jul-2001 18:54:39 MEST