联动北方
LANDING

企业级 IT 运维宝典系列

联动北方 出品

目　录

Beijing Landing Technologies

第 1 章 IPC 相关故障

1.1 Tuxedo 使用的 IPC

Tuxedo 系统大量使用进程间通信（IPC）的资源，以提高通信的有效性以及管理访问共享资源，如公告板（BB）。

IPC 是 UNIX 操作系统核心的一部分，因此所有的 UNIX 都提供了 IPC 的功能。对于 Windows，Tuxedo 开发了 IPC 模块，来提供所需的功能。

由于 Tuxedo 广泛使用 IPC，因而通常需要调整操作系统的 IPC 值，从而支持 Tuxedo 的高效运行。

Tuxedo 常用的 IPC 资源有三类，它们是信号量，消息队列和共享内存。在对 IPC 进行调整之前，我们先分别了解下每一类 IPC 资源的功能和特性。

注：只有少数操作系统（如 AIX）的 IPC 资源是自适应的，不需要手工调整，其他大多数的 UNIX 系统（如 Solaris，HP-UX 等）以及 Linux 的 IPC 资源需要手工调整，否则会导致 Tuxedo 系统无法运行或效率低下。

1.1.1 信号量（Semaphore）

信号量是在操作系统级别设置的，提供锁机制（类似于文件锁）的一系列标志。当一个资源被信号量锁定后，可以防止操作系统中的其他程序访问该资源。可以通过它们来控制文件，共享内存，或其它任何需要限制访问的资源。

一个共享资源通常通过一个信号量来进行资源的保护。该信号量的初始值为“1”，表明该资源可用。一个进程访问这个资源之前，它先检查信号量的值。如果值是 1，则立即将它置为 “0”（说明该资源被占用），接着便可以对此资源执行操作，没有其他任何进程可以同时访问。如果信号量为 0，其它进程查询或修改资源之前，必须等待。当完成资源的访问时，进程负责把信号量置为 1，说明此进程访问资源结束，从而允许其他进程继续访问。

一个信号量的基本功能是，它可以被设置，被检查（是否已经设置），也可以等待，直到信号被清除并重新设置。

信号量一个复杂的地方在于当进行信号量编程的时候，你申请的不是单一的信号量，而是一个信号量集。你可以申请只包括一个信号量的信号量集，而 Tuxedo 不是这么做，它会在建立一个信号量集的同时申请大量的信号量。Tuxedo 用信号量来确保对共享资源访问的一致性。信号量配置太低会导致 Tuxedo 系统应用程序无法启动。

1.1.2 消息队列（Message Queue）

消息队列提供了系统中不同的进程之间传递信息的途径。消息队列可以在进程之间共享，一个进程放在队列中的消息可以被其他进程读取。

Tuxedo 系统使用 UNIX 系统消息队列进行客户端与服务端的通信。这类典型的消息包括服务请求、服务应答、会话消息、通知消息、管理消息、事务控制消息等。

每一个 MSSQ 集（多服务单队列）和每个单独的服务器都有一个接收请求消息队列。每个客户端都有自己的应答队列。服务器若指定 REPLYQ 参数，则也拥有自己的应答队列。

Beijing Landing Technologies

对于正确调优一个应用程序来说，调整内核的消息队列参数是至关重要的，不正确的参数值可以导致无法启动，或严重的性能下降。

1.1.3 共享内存（Shared Memory）

共享内存是操作系统中一个或者多个进程之间共享的一段内存。多个进程可以直接读写共享内存，所以是最快的一种进程间通信机制。为了实现安全通信，它往往与其它通信机制，如信号量结合使用，来达到进程间的同步及互斥。

在 Tuxedo 的环境中，共享内存被用于公告板（BB）和工作站的监听器进程（WSL）对照表。应用程序同样也可以使用共享内存达到自己的目的。

1.1.4 Tuxedo 使用的 IPC 资源

Tuxedo 广泛地使用操作系统的 IPC 资源。

每个把信息张贴到公告板（BB）进程或线程都需要一个 UNIX 信号，这包括本地客户端，服务进程，WSH，BBL，WSL，TMS，BRIDGE，域网关和任何其他系统进程，比如 Event Broker 或/Q 进程。

UNIX 信号量是 Tuxedo 系统启动时分配的，因此以下情况一般比较容易识别，就是如果操作系统中信号量太少系统没有正常启动就会抛出一个错误，表示没有足够的信号量。

消息队列是 Tuxedo 动态分配的，每个本地客户端和 WSH 都需要有一个应答的消息队列。每个 SERVER 都需要一个请求队列，但 SERVER 可以使用 RQADDR 参数（MSSQ）共享请求队列。SERVER 同样可以设置 REPLYQ = Y，以拥有单独的应答队列。

公告板（BB）使用共享内存，它的大小由 UBBCONFIG 中不同的参数控制，比如 MAXACCESSERS，MAXGTT，MAXSERVERS，MAXSERVICES 等。

1.1.5 定义 IPC 限制

为加快处理速度，大多数 Bulletin Board 的空间都是静态分配的，因此正确地进行调整是十分重要的。如果它设置太大，内存和 IPC 资源就会过度消耗，如果它设置太小，达到限制时，应用就会失败。

以下是在 UBBCONFIG 的*RESOUECES 段下影响共享内存大小的参数：

1、 MAXACCESSERS：一个节点允许连接到 Tuxedo 系统的进程的最大数。这个不是所有进程的总和，而是同时访问这个节点的最大进程数。默认为 50。（你可以在每台机器的 MACHINES 段设置 MAXACCESSERS 覆盖默认值）；

2、 MAXSERVERS：应用中的服务器进程的最大数量，它包括所有管理服务器（比如：BBL ,TMS），它是应用中所有进程的总和，默认为 50；

3、 MAXSERVICES：应用发布的服务的最大数量。它是系统中所有服务的总和，默认是 100；

4、 MAXGTT：应用支持的全局事务的最大数量，默认是 100

5、 MAXINTERFACES：在 Tuxedo 应用中指定 CORBA 接口的最大数量。

6、 MAXCONV：指定一台机器上同时进行的会话的最大数量。

Beijing Landing Technologies

1.2 IPC 设置

通常在 Tuxedo 中 IPC 故障的分析相对较简单，它会作为 Tuxedo 错误日志存储到 ULOG 中。不过，Tuxedo 系统产生错误不仅可能产生在启动时，还可能出现在运行状态。

因此解决 IPC 的问题需要从两个方面入手：

首先，应配置操作系统的预期处理 IPC 的负载；

其次，如果 IPC 的环境配置出现问题，需重新配置操作系统的环境，以提高 IPC 处理能力。

注意在操作系统中，Tuxedo 不是唯一消耗 IPC 资源的应用程序，这是十分重要的。其他应用程序（如 ORACLE 数据库），或 Tuxedo 其他应用程序的实例，也需要使用 IPC 资源，所以要设置的 IPC 的资源，以容纳操作系统上所有的应用程序。

要确定一个 Tuxedo 应用需要占用的 IPC 资源大小，可以运行以下命令：

```
 tmloadcf  -c [ubbconfig file]
```

示例 17-1

运行此命令将出现下面的结果：

```
Ipc sizing (minimum /T values only) ...
Fixed Minimums Per Processor
SHMMIN: 1
SHMALL: 1
SEMMAP: SEMMNI
Variable Minimums Per Processor
                    SEMUME,                A                          SHMMAX
                    SEMMNU,                *                            *
Node                SEMMNS  SEMMSL  SEMMSL  SEMMNI  MSGMNI  MSGMAP  SHMSEG
------              ------  ------  ------  ------  ------  ------  ------
PRODUCTIONMACH         105      13     100   A + 1      31      62   173K
where 1 <= A <= 8.

The number of expected application clients per processor should be added to
each MSGMNI value.
```

示例 17-2

其中，A 表示一个变量，它的值介于 1 和 8 之间，首先要确定‘A’的值，用“A*SEMMSL”除以 SEMMSL 值，得到的就是 A 的值。一旦知道 A 的值，你就能通过 A+1 确定 SEMMNI 的值。

列出的值都是在操作系统中，运行这个应用必须的最小值。设置操作系统参数时，还需要考虑其它的产品和应用。

下表是操作系统 IPC 内核参数的含义，以及 Tuxedo 应用系统所需的配置：

Beijing Landing Technologies

类型	名字	描述	取值
共享内存	SHMMAX	单个共享内存段的最大尺寸	需大于 BB 的大小，参考 tmloadcf -c 的输出
	SHMSEG	一个进程可用的共享内存段的最大数目	每个进程可使用的最大共享内存量 =SHMMAX ×SHMSEG
	SHMMNI	系统范围内允许的共享内存段的最大数目	至少大于 SHMSEG
	SHMMIN	单个共享内存段的最小尺寸	一般设置为 1
信号量	SEMMNS	系统范围的最大信号量数量	至少大于 MAXACCESSERS-MAXWSCLIENTS+13
	SEMMNI	可被创建的信号集的数目	通常等于 SEMMNS
	SEMMSL	每套信号集允许的最大信号量数量	通常等于 SEMMNS，因为 Tuxedo 不会大量创建信号集，但它对每个信号集分配尽可能多的信号量
	SEMMAP	用于管理信号量的映射表大小	SEMMNI+2
	SEMMNU	Undo 结构的数目	通常等于 SEMMNS
	SEMUME	每个进程最大 Undo 数量	通常等于 SEMMNS
消息队列	MSGTQL	操作系统内等待处理的最大消息数	需根据应用系统容量设置
	MSGMNB	消息队列的最大尺寸	至少等于 MSGMAX，Tuxedo 为避免消息队列阻塞，当消息长度大于 MSGMNB ×75% 时，消息将被存到文件中去处理，这将大大降低总体性能。
	MSGMAX	每条消息的最大尺寸	至少大于 MSGMNB ×75%
	MSGSEG	系统拥有消息段数目	整个系统消息队列占用的最大空间为 MSGSEG×MSGSSZ，因此 MSGSEG×MSGSSZ 至少要大于 MSGMAX
	MSGSSZ	每个消息段的尺寸	一般设置为 8,16, 32 或 64 字节

Beijing Landing Technologies

	MSGMNI	最多可被创建的消息队列数目	至少大于 MAXACCESSERS+(有响应队列的服务器数-MSSQ 中的服务器数) +MSSQ 的个数+7
	MSGMAP	用于管理消息的映射表大小	MSGTQL + 2

表 17-1

1.3 IPC 命令

下面是一个 IPC 常用命令列表，可进行 IPC 查看和操作。

1.3.1 ipcs

ipcs 打印 IPC 状态。通过选项来控制输出的信息。

如果没有选项，将打印系统中消息队列，共享内存及信号量的相关信息。

常用参数如下：

选项	描述
-b	打印允许的最大消息字节数、共享内存大小和信号集里的信号量个数
-o	打印消息队列中的消息数和连接到共享内存的进程数
-q	打印活动消息队列信息
-s	打印活动信号的信息
-m	打印活动的共享内存段的信息

表 17-2

下面是“ipcs -boq”命令行：

“-b”显示消息队列上允许的最大消息的字节数；

“-o”显示队列上消息数量；

“-q”指定只显示队列相关信息（即不显示共享内存与信号量的信息）。

```
IPCS status from landingbj as of Wed Jun 22 10:41:54 2005
T     ID     KEY        MODE       OWNER    GROUP CBYTES  QNUM QBYTES
Message Queues:
q    513 0x0000bea2 -Rrw-rw-rw-        0        0      0     0  65536
q    770 0x00000000 -Rrw-rw-rw-        0        0      0     0  65536
q    515 0x00000000 -Rrw-rw-rw-        0        0      0     0  65536
q      4 0x00000000 --rw-rw-rw-        0        0  11396    37  65536
```

示例 17-3

Beijing Landing Technologies

从输出结果可以看到，存在一个 ID 为 4 的队列，有 37 个消息等待处理，共 11396 个字节。

MODE 列显示的是 UNIX 的队列权限，OWNER 和 GROUP 列表示队列组和拥有者。

1.3.2 ipcrm

ipcrm 删除一个或多个消息队列，信号量或共享内存。正常情况下， Tuxedo 应用关闭时自动释放 IPC 资源。这个命令是用来清理 Tuxedo 进程失败后遗留的 IPC 资源。某些情况下，Tuxedo 的服务器没有获得释放 IPC 的资源机会，我们必须手工操作。

ipcrm 使用的 ID 为 ipcs 命令输出的 ID，我们可以针对特定的 IPC 资源进行清理。

常用参数如下：

选项	描述
-m <ID>	根据 id 删除共享内存段
-s <ID>	根据 id 删除信号量
-q <ID>	根据 id 删除消息队列

表 17-3

ipcrm 删除上述所有队列信息：

```
ipcrm  -q 513  -q 770  -q 515  -q 4
```

示例 17-4

1.3.3 tmipcrm

tmipcrm 清除 Tuxedo ATMI 应用程序中所分配 IPC 资源，如共享内存，消息队列，信号量。这个命令运行在 Tuxedo 服务器异常中止情况下。在正常情况下， Tuxedo 应用关闭时自动释放 IPC 资源。清除的 IPC 资源包括 Tuxedo ATMI 服务与客户端的 IPC 资源。

tmipcrm 只能清除本机 IPC 资源，不能用于清除在配置文件中远程机器的 IPC 资源。

使用 tmipcrm 必须指定 TUXCONFIG 文件，TUXCONFIG 必须存在，而且可读。

只有管理员或具有相应权限的人可以运行这个命令。

常用参数如下：

选项	描述
-y	确定执行
-n	不删除 IPC 资源，只输出 IPC 资源列表

表 17-4

tmipcrm 输出实例：

```
$ tmipcrm /home/user/landingbj/tuxconfig
Looking for IPC resources in TUXCONFIG file /home/user/landingbj/tuxconfig
```

Beijing Landing Technologies

```
The following IPC resources were found:
Message Queues:
0x2345
0x3456

Semaphores:
0x34567
0x45678

Shared Memory:
0x45678
0x56789
Remove these IPC resources (y/n)?: y
Removing IPC resources  done!
```

示例 17-5

1.3.4 IPC 清除脚本

另一种是使用 ipcrm 开发自定义脚本删除所有的目前正在由这个 UNIX 用户使用的 IPC 资源。这种风格的脚本，它可以按定制删除用户下所有 IPC 资源，而不是针对特定的 Tuxedo 域中的资源

为避免误删其它产品或应用使用的 IPC 资源，建议不同的应用程序在不同用户帐户下运行

非 Linux 下脚本运行环境如下：

```
ipcrm `ipcs |grep <username>|awk '{print "-" $1 " " $2}'`
```

示例 17-6

Linux 下脚本运行环境如下：

```
ipcrm `ipcs -m|grep <username>|awk '{print "shm" " " $2}'`
ipcrm `ipcs -s|grep <username>|awk '{print "sem" " " $2}'`
ipcrm `ipcs -q|grep <username>|awk '{print "msg" " " $2}'`
```

示例 17-7

1.3.5 bbsread

tmadmin 的 bbsread 子命令可以查看当前 Tuxedo 使用的 IPC 资源状态。

bbsread 输出如下：

```
SITE1> bbsread
IPC resources for the bulletin board on machine LANDINGBJ:
SHARED MEMORY:          Key: 0x1013c38
SEGMENT 0:
                         ID: 15730
                       Size: 36924
          Attached processes: 12
       Last attach/detach by: 4181
```

Beijing Landing Technologies

```
This semaphore is the system semaphore
SEMAPHORE:              Key: 0x1013c38
                        Id: 15666
        | semaphore | current |   last   | # waiting |
        |  number   | status  | accesser | processes |
        |-------------------------------------------|
        |     0     |  free   |   4181   |     0     |
        |-------------------------------------------|
This semaphore set is part of the user-level semaphore
SEMAPHORE:              Key: IPC_PRIVATE
                        Id: 11572
        | semaphore | current |   last   | # waiting |
        |  number   | status  | accesser | processes |
        |-------------------------------------------|
        |     0     | locked  |   4181   |     0     |
        |     1     | locked  |   4181   |     0     |
        |     2     | locked  |   4181   |     0     |
        |     3     | locked  |   4181   |     0     |
        |     4     | locked  |   4181   |     0     |
        |     5     | locked  |   4181   |     0     |
        |     6     | locked  |   4181   |     0     |
        |     7     | locked  |   4181   |     0     |
        |     8     | locked  |   4181   |     0     |
        |     9     | locked  |   4181   |     0     |
        |    10     | locked  |   4181   |     0     |
        |    11     | locked  |   4181   |     0     |
        |    12     | locked  |   4181   |     0     |
        |    13     | locked  |   4181   |     0     |
        |-----------|---------|----------|-----------|
```

示例 17-8

bbsread 输出分为两部分：共享内存与信号量。

其中，共享内存显示的信息：

常用参数如下：

字段	描述
ID	IPC 资源唯一标识
Size	共享内存大小
Attached processes	显示共享内存段中活动进程数。这些进程要处理如 BBL,WSH 进程，Tuxedo 服务器等
Last attach/detach by	最后一个附加到或断开共享内存段的进程的 PID

表 17-5

Beijing Landing Technologies

bbsread 显示所有的信号量是否被锁定，访问它的进程 ID，以及等待的进程数。

注意 ipcs 输出的是信号集的信息，bbsread 输出的是信号集中所有信号量的信息。

1.4 IPC 常见疑难问题

当应用服务器关闭失败时，如何清理 IPC 资源？

当 Tuxedo 的应用程序用 tmshutdown 命令关闭。所有的 IPC 资源会被 Tuxedo 从系统中自动清除。但在某些情况下，应用程序可能无法正常关闭，这时 IPC 资源继续保留在系统中无法释放。如果发生这种情况，可能无法重新启动应用程序。

一种解决办法是利用脚本文件来调用系统 IPC 命令删除用户下所有的 IPC 资源。然而，使用这个方法很难区分不同的 IPC 资源集合；一部分可能属于 Tuxedo 系统资源，一部分可能属于 Tuxedo 的应用程序；还有一些属于其它应用程序。如果不小心将 IPC 资源错误地删除，对应用程序会造成严重的危害。

使用 Tuxedo 的 IPC 工具（即 tmipcrm 命令）可以清除某一应用范围内活动 IPC 资源，这样可以避免误删其它 IPC 资源。

Beijing Landing Technologies