WindView™ in IRIX™6.4

Silicon Graphics, Inc.

A description of the changes that have been made to WindView, rtmond, and rtmon-client under IRIX 6.4

Introduction

This paper is provided to document the changes that will be noticed when using WindView™ for IRIX™ with systems executing with IRIX 6.4. It contains information that is not currently documented in the on-line IRIS Insight™ WindView for IRIX Programmer's Guide. It also documents the changes that have occurred with the kernel trace tool *rtmon-client* and the real-time monitoring daemon - *rtmond*.

Packaging Changes

WindView for IRIX

WindView for IRIX 6.4 is packaged and distributed on the REACT/Pro 3.1 CD and labeled as WindView 1.2.

Event Collection Software

The real-time monitoring daemon (*rtmond*) and the client event collection utility (*rtmon-client*) are now packaged as part of the default IRIX execution only environment (*eoe*). They are part of the subsystem *eoe.sw.perf* that is distributed on the IRIX 6.4 CD's. The subsystem *eoe.sw.perf* is installed by default on every IRIX 6.4 system. Therefore, users are no longer required to install software from the REACT/Pro CD in order collect timestamp events from an IRIX 6.4 system.

The *librtmon.a* library (to be linked with programs that call *rtmon_log_user_tstamp*) is now packaged as part of the *dev.sw.lib* subsystem found on the IRIX Development Option (IDO) CD. Therefore, with IRIX 6.4 systems, it is no longer necessary for users to install software from the REACT/Pro CD in order to write programs that call *rtmon_log_user_tstamp*. The man page and sample code for *rtmon_log_user_tstamp* are still provided as part of the react software distribution found on the REACT/Pro 3.1 CD.

Display Changes

Events or Process States Added

The new events identified in Table 1 may be encountered when taking WindView traces under IRIX 6.4:

TABLE 1. **New WindView Events**

Event#	Description	Possible Cause
20018	Kernel Profile Event - 32bit	Kernel profiling is enabled and an event has been logged - 32bit kernel
20019	Kernel Profile Event - 64bit	Kernel profiling is enabled and an event has been logged - 64bit kernel
20203	Scheduler Event	IRIX scheduler has evaluated the run queue

Changes to How Certain Events are Displayed

One of the first things that users of WindView under 6.4 will notice is that certain process states will be displayed with negative Process IDentification (PID) numbers (PIDs are displayed in parenthesis after the process names on the right hand side of the WindView View Graph). A negative PID indicates that the process state is either a service thread (a thread that has been dispatched by the kernel to handle a given task) or an interrupt thread (a thread dispatched to process an interrupt). Many service threads previously (pre IRIX 6.4) were executed as IRIX daemons. Service thread and interrupt thread PIDs cannot be correlated to output from *ps* or *top* and for all practical purposes should be ignored by WindView users. IRIX 6.4 WindView users can also ignore the bracketed numbers that may appear in the display of some process states (e.g., ip0intrd [0,12]) as this provides no useful information to WindView users.

Some of the interrupt and service threads that might be encountered are identified in the Table 2.

TABLE 2. **Some Common Sthreads and Ithreads**

Process State	Description
mediad	media daemon service thread
ip0intrd	Kernel profiling is enabled and an event has been logged - 64bit kernel
rtnetd0	real-time network daemon service thread
timein0	timer callout service thread
bdflush	daemon to flush delayed disk writes
vfs_sync	daemon to flush superblocks, dirty fs-nodes
xfsd	daemon for xfs (file system) tasks
ioc3_ivec	interrupt thread for the ioc3 on base I/O
sockd	socket daemon - handles networking timeouts

New Interrupts

No new interrupts have been added from what is documented in the WindView for IRIX Programmer's Guide. However, it should be noted that on systems that have built-in audio (i.e., O2, Octane and Onyx2), the profiler interrupt (shown as interrupt level INT 8) will occur every 1 millisecond, even when no audio is being played/recorded. This interrupt will occur on only a single CPU for multiprocessor systems (typically CPU 0).

Event Collection Changes

Changes to *rtmond*

The location of the rtmond binary is now */usr/etc* (instead of */usr/react/etc*).

Changes to *rtmon-client*

The location of the *rtmon-client* binary is now */usr/sbin* (instead of */usr/react/bin*).

Under 6.4, users **must** use the *-o* option to *rtmon-client* in order to collect timestamp data that is readable by WindView . This option was mistakenly left off of the *rtmon-client* man page. The following error may be received if a timestamp file (**.wvr*) is attempted to be analyzed by WindView:

Could not open *filename.wvr*

The rtmon-client utility has an additional option (*-m event-mask*). This option can be ignored by users who are collecting timestamps for WindView.

Changes to *rtmon_log_user_tstamp*

The prototype for *rtmon_log_user_tstamp* is now:

void rtmon_log_user_tstamp(evt, qual1, qual2, qual3, qual4)

event_t evt;

unsigned long long qual1, qual2, qual3, qual4;

There is no longer a need to specify the number of event qualifiers between the event number and the qualifiers themselves so the total number of parameters in this library call has changed from six to five. Therefore, any code that previously used this library call will need to be modified for 6.4 compilation by removing the argument that specified the number of qualifiers.

Additional utilities

An additional utility has been added that permits users to convert a timestamp file collected with *rtmon_log_user_tstamp* to an ascii representation. This utility, *rtmon-dump,* is currently undocumented and is provided for debug purposes only.

Special Note

IRIX 6.3 systems do not have the kernel support to provide timestamp events to WindView or *rtmon-client*. While IRIX 6.3 systems can run the WindView GUI for display of events and collection of events on other systems, they cannot run rtmond and provide kernel tracing from their own system.