NAME

ktr - kernel tracing facility

SYNOPSIS

options KTR
options ALQ
options KTR_ALQ
options KTR_COMPILE=(KTR_LOCK|KTR_INTR|KTR_PROC)
options KTR_CPUMASK=0x3
options KTR_ENTRIES=8192
options KTR_MASK=(KTR_INTR|KTR_PROC)
options KTR_VERBOSE

DESCRIPTION

The **ktr** facility allows kernel events to be logged while the kernel executes so that they can be examined later when debugging. The only mandatory option to enable **ktr** is "options KTR".

The KTR_ENTRIES option sets the size of the buffer of events. The size of the buffer in the currently running kernel can be found via the sysctl *debug.ktr.entries*. By default the buffer contains 1024 entries.

Event Masking

Event levels can be enabled or disabled to trim excessive and overly verbose logging. First, a mask of events is specified at compile time via the KTR_COMPILE option to limit which events are actually compiled into the kernel. The default value for this option is for all events to be enabled.

Secondly, the actual events logged while the kernel runs can be further masked via the run time event mask. The KTR_MASK option sets the default value of the run time event mask. The runtime event mask can also be set by the loader(8) via the *debug.ktr.mask* environment variable. It can also be examined and set after booting via the *debug.ktr.mask* sysctl. By default the run time mask is set to block any tracing. The definitions of the event mask bits can be found in <*sys/ktr_class.h*>.

Furthermore, there is a CPU event mask whose default value can be changed via the KTR_CPUMASK option. When two or more parameters to KTR_CPUMASK, are used, it is important they are not separated by whitespace. A CPU must have the bit corresponding to its logical id set in this bitmask for events that occur on it to be logged. This mask can be set by the loader(8) via the *debug.ktr.cpumask* environment variable. It can also be examined and set after booting via the *debug.ktr.cpumask* sysctl. By default, only CPUs specified in KTR_CPUMASK will log events. See *sys/conf/NOTES* for more information.

Verbose Mode

By default, events are only logged to the internal buffer for examination later, but if the verbose flag is set then they are dumped to the kernel console as well. This flag can also be set from the loader via the *debug.ktr.verbose* environment variable, or it can be examined and set after booting via the *debug.ktr.verbose* sysctl. If the flag is set to zero, which is the default, then verbose output is disabled. If the flag is set to one, then the contents of the log message and the CPU number are printed to the kernel console. If the flag is greater than one, then the filename and line number of the event are output to the console in addition to the log message and the CPU number. The KTR_VERBOSE option sets the flag to one.

Examining the Events

The KTR buffer can be examined from within ddb(4) via the **show ktr** [**/vV**] command. This command displays the contents of the trace buffer one page at a time. At the "--more--" prompt, the Enter key displays one more entry and prompts again. The spacebar displays another page of entries. Any other key quits. By default the timestamp, filename, and line number are not displayed with each log entry. If the **/v** modifier is specified, then they are displayed in addition to the normal output. If the **/V** modifier is specified, then just the timestamp is displayed in addition to the normal output. Note that the events are displayed in reverse chronological order. That is, the most recent events are displayed first.

Logging ktr to Disk

The KTR_ALQ option can be used to log **ktr** entries to disk for post analysis using the ktrdump(8) utility. This option depends on the ALQ option. Due to the potentially high volume of trace messages the trace mask should be selected carefully. This feature is configured through a group of sysctls.

debug.ktr.alq_file displays or sets the file that **ktr** will log to. By default its value is */tmp/ktr.out*. If the file name is changed while **ktr** is enabled it will not take effect until the next invocation.

debug.ktr.alq_enable enables logging of **ktr** entries to disk if it is set to one. Setting this to 0 will terminate logging to disk and revert to logging to the normal ktr ring buffer. Data is not sent to the ring buffer while logging to disk.

debug.ktr.alq_max is the maximum number of entries that will be recorded to disk, or 0 for infinite. This is helpful for limiting the number of particularly high frequency entries that are recorded.

debug.ktr.alq_depth determines the number of entries in the write buffer. This is the buffer that holds entries before they are written to disk and defaults to the value of the KTR_ENTRIES option.

debug.ktr.alq_failed records the number of times we failed to write an entry due to overflowing the

write buffer. This may happen if the frequency of the logged **ktr** messages outpaces the depth of the queue.

debug.ktr.alq_cnt records the number of entries that have currently been written to disk.

SEE ALSO

ktrdump(8), alq(9), ktr(9)

HISTORY

The KTR kernel tracing facility first appeared in BSD/OS 3.0 and was imported into FreeBSD 5.0.