Kernel Introspection: A Proposal -- Design Discussion#

This is a partial collection of design notes. There is more in the individual meeting notes liked from the Kernel Introspection page.

1. Ideas for specific stats
2. Pathname Evolution

Ideas for specific stats#

The kernel stats notifier, bulk transfer, and pathname space all have to identify data items of interest in the kernel. There may be some natural groupings.

Notification Domains#

- System
- Sub-system (ill-defined, but think memory/time partitions)
- Process
- Thread

Potential Notification Items#

A few stats that may be of interest, with the notification domain identified.

- CPU time [RLIMIT_CPU] (thread, process, sub-system?)
- File size [RLIMIT_FILE] (external managers...)
- Data segment size [RLIMIT_DATA] (process)
- Stack size [RLIMIT_STACK (only primary stack)] (thread)
- Core file size [RLIMIT_CORE] (process)
- Num open files [RLIMIT_NOFILE] (process)
- Virtual addr space [RLIMIT_AS] (process)
- mlock'd memory [RLIMIT_MEMLOCK] (process)
- Num child processes [RLIMIT_NPROC] (process, sub-system, system)
- Num threads [RLIMIT_NTHR] (process, ?system?)

* Send queue length (process)#

- Num system calls (thread)
- Num sync objects (system)
- Num timers (process)
- Num pending signals (process)
- Physical memory usage (sub-system, system)
- channel count (process)
- servier id count (process)
- stack size. both RLIMIT_STACK which is main thread only, and any thread in process
- tbd: data_pages_used

An idea for CPU time usage#

POSIX has API's (clock_getcpuclockid(), pthread_getcpuclockid()) that return clock_id's for the cpu time used by a process or thread within a process. Right now we only allow those clock_id's to be used to get the current

value, but POSIX allows them to be used when setting timers. If we put in support for that (not a heck of a lot of code), we could handle the CPU time usage notification item in a POSIX compliant manner.

Pathname evolution#

Objective#

Define an extension of the current pathname space to support

- the generic notifier, part of Kernel Introsepction's kernel stats notifier
- bulk data transfer, part of Kernel Introspection
- notification needs of memory partitioning
- notification needs of adaptive partitioning scheduler

While being conceptually consistant with the current /proc name space.

Discussion#

General applicability#

- These notions may be useful for all resmgrs. Whereupon we should offer a lib to parse paths with URI modifiers.

Bulk Transfer#

The observation is that opening /proc and reading is fundementall what we are trying to accomplish with bulk transfer: get data about every entity that subtends /proc, that is all pids and tids. The only difference between the current behavior of reading /proc and what we want for bulk transfer (a stream of tagged structs of all pids/tids with possibly several specialised structs for each entity plus some options for filtering) is:

1. the format of the returned data
2. the detail returned

For example reading /proc returns a list of pids. For bulk transfer we also want to return something about all pids/tids, but in a different format.

So the observation is that bulk transfer should use the same name space, /proc, rather than something like /proc/info or /proc/allpids/info as previously proposed, since bulk transfer is only returning a different format, not introducing a new entity worthy of a new name in the names space.

So rather than introduce a new name into the name space for bulk transfer, the idea is to introduce a qualifer to specify the format and detail of the data to be returned.

The general idea is to add URI modifier on the end of the path.

Examples

- /proc?bulkinfo means return a stream of taged structs (bulk transfer format) for all pids/tid, and also for all structs availablbe for pids and tids.
    - this accompished the primary goal of bulk transfer to support Cisco's Wdsysmon requirements

- /proc/4721?bulkinfo means return a stream of tag structs for all structs about pid 4721. These would include
  - tagged version of debug_process_t
  - a tagged version of debug_thread_t for each thread of process 4721
  - new structs for memory usage, time since last state change, and all the other fields to support customers HA controllers
  - tagged version of the the partition info normally returned by /proc/4721/partition if memory partitioning is present

- /proc?bulkinfo=partition+process filters the struct stream to return only tagged versions of debug_process_t and partition information.

- /proc/bulkinfo?filter=<list of tag names> is the general format for bulk transfer path that specifies an explicit of tagged struct names.

- /proc?bulkinfo=all an alternative to /proc?bulkinfo if we want to be explicit. This would introduce the idea of tagnames which mean a set tags.

The allowable qualifiers in a URI for bulk transfer are tag names of structs. The tag names, their numerical values, and the structs will be defined in public interfaces.

The design permits some tags/structs to be optional, kernel modules, if need be.

URI References

- Wikipedia
- RFC1630

What? Is that all?

Err... We we haven't written up cohesive picture of all our design discussions yet. The raw notes are in the meeting minutes.