# Kernel Introspection:Design Meeting 2007-04-24#

## Who#

bstecher, adanko, dbailey, mkisel, shiv

# Summary#

### What we did#

- discussed deadlock-timer api
- considered every thread state to determine which are necessary for deadlock-warning notification.
- we considered the rlimits, and fields in the Cisco specific debug_thread and debug_process structs, as candidates for thresholds in the Generic Notifier.

### Stuff to do#

Deadlock related

- follow up with filesystem guys to check that customer's HA controllers could register for notification on changes to /proc as a way to know a process has been created.

Notifier related

- decide if we will have global_number_of_processes threshold in Generic Notifier
- design Generic Notifier to have threshold events, and "any change" events.
- design the hooks for incrementing and testing individual thresholds
- design the optionality for indivdual thresholds and the whole threshold notification system.
- decide what does it mean to the sender when the act of sending blows one of the receiver's limits
    - we should list the set of signals that return an **errno** to the sender when the receiver blows a limit
        - as a first approximation, this may be all signals
- need to clarify what we mean by sub-system level thresholds (per-partition?)

Pathname space related

- we need to describe the pathnames to distinguish thread-level, process-level, sub-system and system-level versions of analogous thresholds.
- we need to decide if setting a threshold on a partial path does:
    1. apply its limit, recursively to each subtending node in the path
    2. apply its limit to the sum of the usage of all the subtending nodes
- for system limits, with per-process analogs, what does the pathname look like?

At this point, the major missing piece of the design is the path name space.

# More on Deadlock#

---

### Default Timer interface#

- We should use the existing timer_timeout interface to set the timeout value for default timers set for deadlock warning notification. The timeout value would be set per process.
- the function takes a bit set of blocking states
- advise customers to have their HA controllers register for process creation (and death) notifications.
  - on notification, the customer's HA controller should set the default timeout for that process
  - this requires us to create a new notification for process creation. There are two alternatives:
    1. new threshold in the Generic Noftifier: global_number_of_proceses. In addition, we neeed a threshold option of "notify any change". A threshold crossing is not enough in this case.
    2. advise customers to use an api in the file system which notifies of directory changes in /proc. (Need to check with FS guys.)

### Canonical list of thread states#

We examined all the thread states to pick the minimum list, that customer's HA controlller's would have to set default timers on, to be assured of being notified whenever a deadlock has occurred. The criteria were:

- state state must not be transient
- it must not be legal to stay in the state forever. eg. not receive
- the associate kernel object must be owned (so it's possible to write a loop-detection algorithm to trace it)
- not a network state (because we choose to not do inter-node loop detection)

That leaves these states:

- state_send
- state_mutex
- state_join

We also re-confirmed our belief that it is sufficient for the kernel to send one notification per timer (i.e. oneshot timer) since we only want to notifiy on the blocking event that triggers the deadlock.

---

## Additional Limits#

---

We reviewed the current list of rlimits, plus the some new fields requested by customers for the debug_thread_t and debug_process_t structures, to find limits we will support in the Generic Notifier.

For a customers HA controller's main purpose of defending their system from resource hogs, we specifically looked for resources which are globally exhaustible by the misbehavior of one process. We have also included limits on individual processes because we believe they would be generally useful. (For example, a process may want to be notified when one of its own threads enters a recursion loop.)

### Prospective Limits#

Limits which we think we should add to the generic notifier:

- channel count (perprocess)
- server id count (perprocess)
- stack size. Note we wish to support both the POSIX rlimit behavior in addition to a general behavior:
  1. POSIX implies that RLIMIT_STACK should only apply to the main thread
  2. we also want to be able to set a threshold on any thread in the process
- data_pages_used (maybe)

## Deferred#

RLIMITS or fields which meet the theoretical criteria for thresholds, but we think are low priority or of limited usefulness.

- RLIMIT_FILESIZE
- RLIMIT_MEMLOCK