Subject: Re: [PATCH 1/2] signals: kill(-1) should only signal processes in the same namespace
Posted by Sukadev Bhattiprolu on Thu, 17 Jul 2008 18:13:13 GMT
View Forum Message <> Reply to Message

Daniel Hokka Zakrisson [daniel@hozac.com] wrote:
| While moving Linux-VServer to using pid namespaces, I noticed that
| kill(-1) from inside a pid namespace is currently signalling every
| process in the entire system, including processes that are otherwise
| unreachable from the current process.
|
| This patch fixes it by making sure that only processes which are in
| the same pid namespace as current get signalled.
|
| Signed-off-by: Daniel Hokka Zakrisson <daniel@hozac.com>
|
| diff --git a/include/linux/pid_namespace.h b/include/linux/pid_namespace.h
| index caff528..4cf41bd 100644
| --- a/include/linux/pid_namespace.h
| +++ b/include/linux/pid_namespace.h
| @@ -40,6 +40,8 @@ static inline struct pid_namespace *get_pid_ns(struct
| pid_namespace *ns)
| extern struct pid_namespace *copy_pid_ns(unsigned long flags, struct
| pid_namespace *ns);
| extern void free_pid_ns(struct kref *kref);
| extern void zap_pid_ns_processes(struct pid_namespace *pid_ns);
| +extern int task_in_pid_ns(struct task_struct *tsk,
| +   struct pid_namespace *pid_ns);
|
| static inline void put_pid_ns(struct pid_namespace *ns)
| {
| @@ -72,6 +74,12 @@ static inline void zap_pid_ns_processes(struct
| pid_namespace *ns)
| {
| BUG();
| }
| +
| +static inline int task_in_pid_ns(struct task_struct *tsk,
| +   struct pid_namespace *ns)
| +{
| + return 1;
| +}
| #endif /* CONFIG_PID_NS */
|
| static inline struct pid_namespace *task_active_pid_ns(struct
| task_struct *tsk)
| diff --git a/kernel/pid_namespace.c b/kernel/pid_namespace.c
| index 98702b4..3e71011 100644

```
| --- a/kernel/pid_namespace.c
| +++ b/kernel/pid_namespace.c
| @@ -188,6 +188,26 @@ void zap_pid_ns_processes(struct pid_namespace *pid_ns)
|    return;
|   }
|
| +/*
| + * Checks whether tsk has a pid in the pid namespace ns.
| + * Must be called with tasklist_lock read-locked or under rcu_read_lock()
| + */
| +int task_in_pid_ns(struct task_struct *tsk, struct pid_namespace *ns)
| +{
| + struct pid *pid = task_pid(tsk);
| +
| + if (!pid)
| +  return 0;
| +
| + if (pid->level < ns->level)
| +  return 0;
```

ns can be NULL if tsk is exiting.

Like Pavel said, we had couple of attempts to fix the larger problem of
signal semantics in containers but did not have a consensus on handling
blocked/unhandled signals to container-init.

It would still be good to fix this "kill -1" problem.

Eric had a slightly optimized interface, 'pid_in_pid_ns()' in following
patchset. Maybe we could use that ?

https://lists.linux-foundation.org/pipermail/containers/2007-December/009174.html

```
| +
| + if (pid->numbers[ns->level].ns != ns)
| +  return 0;
| +
| + return 1;
| +}
| +
|   static __init int pid_namespaces_init(void)
|   {
|    pid_ns_cachep = KMEM_CACHE(pid_namespace, SLAB_PANIC);
| diff --git a/kernel/signal.c b/kernel/signal.c
| index 6c0958e..93713a5 100644
| --- a/kernel/signal.c
| +++ b/kernel/signal.c
```

```
| @@ -1145,7 +1145,8 @@ static int kill_something_info(int sig, struct
| siginfo *info, int pid)
|     struct task_struct * p;
|
|     for_each_process(p) {
| -   if (p->pid > 1 && !same_thread_group(p, current)) {
| +    if (p->pid > 1 && !same_thread_group(p, current) &&
| +        task_in_pid_ns(p, current->nsproxy->pid_ns)) {
|       int err = group_send_sig_info(sig, info, p);
|       ++count;
|       if (err != -EPERM)
| --
| 1.5.5.1
| _____
| Containers mailing list
| Containers@lists.linux-foundation.org
| https://lists.linux-foundation.org/mailman/listinfo/containers
```

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers