
Subject: Re: [RFC][PATCH 0/15] Pid namespaces
Posted by [Pavel Emelianov](#) on Fri, 27 Jul 2007 06:47:18 GMT
[View Forum Message](#) <> [Reply to Message](#)

sukadev@us.ibm.com wrote:

> sukadev@us.ibm.com [sukadev@us.ibm.com] wrote:

> | Pavel,

> |

> | We seem to have a memory leak. Its new in this patchset (i.e the

> | following test ran fine on the 2.6.22-rc6-mm1 patchset).

> |

> | To repro: run "pidns_exec ./mypid" in a tight-loop - where mypid.c

> | is:

> |

> | #include <stdio.h>

> | #include <unistd.h>

> |

> | main()

> | {

> | printf("Pid %d, Ppid %d, Pgid %d, Sid %d\n",

> | getpid(), getppid(), getpgid(0), getsid(0));

> | }

> |

> | I ran into OOM in about 30 mins. I am still investigating.

> |

> | BTW, can we include a simple test program like the pidns_exec in this

> | patch-0 for whoever want to play with pidns ?

> |

> | Suka

> |

>

> I think the problem is in create_pid_namespace(). kref_init() sets the

> refcount to 1 and then we do a get_pid_ns() which sets it to 2.

>

> free_nsproxy() frees one of this references, but the other is never

> freed.

That's it! I've switched from my old fast (and bad) reference counting scheme to new slow (and correct) one, but forgot to remove this "get". This is the real problem. Thanks :)

> This patch seems to fix the leak. The patch also creates a slab cache

> for the pid_namespace

OK, I'll include this patch. Thanks.

> ---

>

```

> Create a slab-cache for 'struct pid_namespace' and fix a memory leak
> due to an extra reference in create_pid_namespace().
>
> ---
> kernel/pid.c | 10 ++++++----
> 1 file changed, 6 insertions(+), 4 deletions(-)
>
> Index: lx26-23-rc1-mm1/kernel/pid.c
> =====
> --- lx26-23-rc1-mm1.orig/kernel/pid.c 2007-07-26 20:08:16.000000000 -0700
> +++ lx26-23-rc1-mm1/kernel/pid.c 2007-07-26 22:31:42.000000000 -0700
> @@ -35,6 +35,7 @@
> static struct hlist_head *pid_hash;
> static int pidhash_shift;
> struct pid init_struct_pid = INIT_STRUCT_PID;
> +static struct kmem_cache *pid_ns_cachep;
>
> int pid_max = PID_MAX_DEFAULT;
>
> @@ -525,7 +526,7 @@ static struct pid_namespace *create_pid_
> struct pid_namespace *ns;
> int i;
>
> - ns = kmalloc(sizeof(struct pid_namespace), GFP_KERNEL);
> + ns = kmem_cache_alloc(pid_ns_cachep, GFP_KERNEL);
> if (ns == NULL)
> goto out;
>
> @@ -544,7 +545,6 @@ static struct pid_namespace *create_pid_
>
> set_bit(0, ns->pidmap[0].page);
> atomic_set(&ns->pidmap[0].nr_free, BITS_PER_PAGE - 1);
> - get_pid_ns(ns);
>
> for (i = 1; i < PIDMAP_ENTRIES; i++) {
> ns->pidmap[i].page = 0;
> @@ -556,7 +556,7 @@ static struct pid_namespace *create_pid_
> out_free_map:
> kfree(ns->pidmap[0].page);
> out_free:
> - kfree(ns);
> + kmem_cache_free(pid_ns_cachep, ns);
> out:
> return ERR_PTR(-ENOMEM);
> }
> @@ -567,7 +567,7 @@ static void destroy_pid_namespace(struct
>
> for (i = 0; i < PIDMAP_ENTRIES; i++)

```

```
> kfree(ns->pidmap[i].page);
> - kfree(ns);
> + kmem_cache_free(pid_ns_cachep, ns);
> }
>
> struct pid_namespace *copy_pid_ns(unsigned long flags, struct pid_namespace *old_ns)
> @@ -687,4 +687,6 @@ void __init pidmap_init(void)
>   init_pid_ns.pid_cachep = create_pid_cachep(1);
>   if (init_pid_ns.pid_cachep == NULL)
>       panic("Can't create pid_1 cachep\n");
> +
> + pid_ns_cachep = KMEM_CACHE(pid_namespace, SLAB_PANIC);
> }
>
```

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