Subject: [PATCH 1/2] cryo: re-enable checkpointing of thread area
Posted by Benjamin Thery on Wed, 11 Jun 2008 14:14:00 GMT
View Forum Message <> Reply to Message

This patch re-enable the code that checkpoints (and restore) and thread
area (ldt) using ptrace_get_thread_area(). This is seem to improve the
situation a lot on systems with NPTL: it solved one of the general
protection fault I had when restarting a program.

Signed-off-by: Benjamin Thery <benjamin.thery@bull.net>
---
 cr.c | 17 ++++++++---------
 1 file changed, 8 insertions(+), 9 deletions(-)

Index: cryodev/cr.c
===
--- cryodev.orig/cr.c
+++ cryodev/cr.c
@@ -24,7 +24,7 @@
 #include <signal.h>
 #include <errno.h>

-#include <asm/ldt.h> /* for redhat 9.0, NPTL */
+#include <asm/ldt.h> /* for NPTL */

 #include "utils.h"
 #include "sci.h"
@@ -513,14 +513,12 @@ static int save_process_data(pid_t pid,
   return 0;
  }

- /* This is required in redhat9 */
-#if 0
+ /* This is required for NPTL */
  {
- modify_ldt_t ldt;
+ struct user_desc ldt;
   if (ptrace_get_thread_area(pid, &ldt) == 0)
    write_item(fd, "ldt", (void *)&ldt, sizeof(ldt));
  }
-#endif

  snprintf(fname, sizeof(fname), "/proc/%u/exe", pid);
  memset(exe, 0, sizeof(exe));
@@ -1237,7 +1235,7 @@ static int process_restart(int fd, int m
  char *exe = NULL, *cwd = NULL, *sargv = NULL, *senv = NULL;
  struct user_regs_struct *regs = NULL;
  struct user_fpregs_struct *fpregs = NULL;

```
- //modify_ldt_t *ldt = NULL;
+ struct user_desc *ldt = NULL;
  int *exitsig = NULL;
  sigset_t *sigmask = NULL, *sigpend = NULL;
  struct sigaction *sigact = NULL;
@@ -1262,7 +1260,7 @@ static int process_restart(int fd, int m
    Free(senv);
    Free(regs);
    Free(fpregs);
-   //Free(ldt);
+   Free(ldt);
    Free(sigact);
    Free(sigmask);
    Free(sigpend);
@@ -1276,7 +1274,7 @@ static int process_restart(int fd, int m
   else ITEM_SET(regs, struct user_regs_struct);
   else ITEM_SET(fpregs, struct user_fpregs_struct);
-  //else ITEM_SET(ldt, modify_ldt_t);
+  else ITEM_SET(ldt, struct user_desc);
   else ITEM_SET(sigact, struct sigaction);
   else ITEM_SET(sigmask, sigset_t);
   else ITEM_SET(sigpend, sigset_t);
@@ -1304,7 +1302,8 @@ static int process_restart(int fd, int m
     ERROR("lh_hash_add(%p, %u, %p)\n", &hpid, (unsigned)*pid, (void *)npid);
     return -1;
    }
-   //if (ldt) ptrace_set_thread_area(npid, ldt);
+   if (ldt)
+    ptrace_set_thread_area(npid, ldt);
    if (cwd) PT_CHDIR(npid, cwd);
    restore_fd(fd, npid);
   } else if (ITEM_IS("SOCK")) {

--
```

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

Subject: Re: [PATCH 1/2] cryo: re-enable checkpointing of thread area
Posted by serue on Wed, 11 Jun 2008 15:28:06 GMT
View Forum Message <> Reply to Message

Quoting Benjamin Thery (benjamin.thery@bull.net):
> This patch re-enable the code that checkpoints (and restore) and thread
> area (ldt) using ptrace_get_thread_area(). This is seem to improve the

> situation a lot on systems with NPTL: it solved one of the general
> protection fault I had when restarting a program.
>
> Signed-off-by: Benjamin Thery <benjamin.thery@bull.net>

Benjamin, you rock. This fixes my kvm image as well.

Nadia, Suka, could you confirm that this does *not* break cryo on your
systems?

Patched added to git tree.

thanks,
-serge

> ---
> cr.c | 17 ++++++++---------
> 1 file changed, 8 insertions(+), 9 deletions(-)
>
> Index: cryodev/cr.c
> ===
> --- cryodev.orig/cr.c
> +++ cryodev/cr.c
> @@ -24,7 +24,7 @@
> #include <signal.h>
> #include <errno.h>
>
> -#include <asm/ldt.h> /* for redhat 9.0, NPTL */
> +#include <asm/ldt.h> /* for NPTL */
>
> #include "utils.h"
> #include "sci.h"
> @@ -513,14 +513,12 @@ static int save_process_data(pid_t pid,
>   return 0;
>   }
>
> - /* This is required in redhat9 */
> -#if 0
> + /* This is required for NPTL */
>   {
> - modify_ldt_t ldt;
> + struct user_desc ldt;
>   if (ptrace_get_thread_area(pid, &ldt) == 0)
>   write_item(fd, "ldt", (void *)&ldt, sizeof(ldt));
>   }
> -#endif
>
>   snprintf(fname, sizeof(fname), "/proc/%u/exe", pid);

```
>   memset(exe, 0, sizeof(exe));
> @@ -1237,7 +1235,7 @@ static int process_restart(int fd, int m
>   char *exe = NULL, *cwd = NULL, *sargv = NULL, *senv = NULL;
>   struct user_regs_struct *regs = NULL;
>   struct user_fpregs_struct *fpregs = NULL;
> - //modify_ldt_t *ldt = NULL;
> + struct user_desc *ldt = NULL;
>   int *exitsig = NULL;
>   sigset_t *sigmask = NULL, *sigpend = NULL;
>   struct sigaction *sigact = NULL;
> @@ -1262,7 +1260,7 @@ static int process_restart(int fd, int m
>     Free(senv);
>     Free(regs);
>     Free(fpregs);
> -   //Free(ldt);
> +   Free(ldt);
>     Free(sigact);
>     Free(sigmask);
>     Free(sigpend);
> @@ -1276,7 +1274,7 @@ static int process_restart(int fd, int m
>    else ITEM_SET(cwd, char);
>    else ITEM_SET(regs, struct user_regs_struct);
>    else ITEM_SET(fpregs, struct user_fpregs_struct);
> -  //else ITEM_SET(ldt, modify_ldt_t);
> +  else ITEM_SET(ldt, struct user_desc);
>    else ITEM_SET(sigact, struct sigaction);
>    else ITEM_SET(sigmask, sigset_t);
>    else ITEM_SET(sigpend, sigset_t);
> @@ -1304,7 +1302,8 @@ static int process_restart(int fd, int m
>      ERROR("lh_hash_add(%p, %u, %p)\n", &hpid, (unsigned)*pid, (void *)npid);
>      return -1;
>     }
> -   //if (ldt) ptrace_set_thread_area(npid, ldt);
> +   if (ldt)
> +    ptrace_set_thread_area(npid, ldt);
>     if (cwd) PT_CHDIR(npid, cwd);
>     restore_fd(fd, npid);
>    } else if (ITEM_IS("SOCK")) {
>
> --
```

_____

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