Subject: [RFC][PATCH] Use struct pid reference in autofs rather than pid_t
Posted by Sukadev Bhattiprolu on Tue, 23 Jan 2007 23:25:26 GMT
View Forum Message <> Reply to Message

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Subject: Use struct pid reference in autofs rather than pid_t

Make autofs container-friendly by caching struct pid reference rather
than pid_t.

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Cc: Cedric Le Goater <clg@fr.ibm.com>
Cc: Dave Hansen <haveblue@us.ibm.com>
Cc: Serge Hallyn <serue@us.ibm.com>
Cc: Eric Biederman <ebiederm@xmission.com>
Cc: containers@lists.osdl.org
---
 fs/autofs/autofs_i.h | 4 ++--
 fs/autofs/inode.c | 16 ++++++++++------
 fs/autofs/root.c | 6 ++++--
 fs/autofs4/autofs_i.h | 10 +++++-----
 fs/autofs4/inode.c | 12 ++++++------
 fs/autofs4/root.c | 3 ++-
 fs/autofs4/waitq.c | 8 ++++----
 7 files changed, 33 insertions(+), 26 deletions(-)

Index: lx26-20-rc4-mm1/fs/autofs/autofs_i.h
===
--- lx26-20-rc4-mm1.orig/fs/autofs/autofs_i.h 2006-11-29 13:57:37.000000000 -0800
+++ lx26-20-rc4-mm1/fs/autofs/autofs_i.h 2007-01-23 12:59:00.245379800 -0800
@@ -101,7 +101,7 @@ struct autofs_symlink {
 struct autofs_sb_info {
  u32 magic;
  struct file *pipe;
- pid_t oz_pgrp;
+ struct pid *oz_pgrp;
  int catatonic;
  struct super_block *sb;
  unsigned long exp_timeout;
@@ -122,7 +122,7 @@ static inline struct autofs_sb_info *aut
    filesystem without "magic".) */

 static inline int autofs_oz_mode(struct autofs_sb_info *sbi) {
- return sbi->catatonic || process_group(current) == sbi->oz_pgrp;
+ return sbi->catatonic || task_pgrp(current) == sbi->oz_pgrp;
 }

 /* Hash operations */

```
Index: lx26-20-rc4-mm1/fs/autofs4/autofs_i.h
===================================================================
--- lx26-20-rc4-mm1.orig/fs/autofs4/autofs_i.h 2007-01-19 17:29:25.000000000 -0800
+++ lx26-20-rc4-mm1/fs/autofs4/autofs_i.h 2007-01-23 13:48:11.085783872 -0800
@@ -35,7 +35,7 @@
 /* #define DEBUG */

 #ifdef DEBUG
-#define DPRINTK(fmt,args...) do { printk(KERN_DEBUG "pid %d: %s: " fmt "\n" , current->pid ,
__FUNCTION__ , ##args); } while(0)
+#define DPRINTK(fmt,args...) do { printk(KERN_DEBUG "pid %d: %s: " fmt "\n" ,
pid_nr(task_pid(current)), __FUNCTION__ , ##args); } while(0)
 #else
 #define DPRINTK(fmt,args...) do {} while(0)
 #endif
@@ -79,8 +79,8 @@ struct autofs_wait_queue {
  u64 ino;
  uid_t uid;
  gid_t gid;
- pid_t pid;
- pid_t tgid;
+ struct pid *pid;
+ struct pid *tgid;
  /* This is for status reporting upon return */
  int status;
  atomic_t wait_ctr;
@@ -96,7 +96,7 @@ struct autofs_sb_info {
  u32 magic;
  int pipefd;
  struct file *pipe;
- pid_t oz_pgrp;
+ struct pid *oz_pgrp;
  int catatonic;
  int version;
  int sub_version;
@@ -127,7 +127,7 @@ static inline struct autofs_info *autofs
    filesystem without "magic".) */

 static inline int autofs4_oz_mode(struct autofs_sb_info *sbi) {
- return sbi->catatonic || process_group(current) == sbi->oz_pgrp;
+ return sbi->catatonic || task_pgrp(current) == sbi->oz_pgrp;
 }

 /* Does a dentry have some pending activity? */
Index: lx26-20-rc4-mm1/fs/autofs/inode.c
===================================================================
--- lx26-20-rc4-mm1.orig/fs/autofs/inode.c 2007-01-19 17:29:25.000000000 -0800
+++ lx26-20-rc4-mm1/fs/autofs/inode.c 2007-01-23 13:17:35.016908768 -0800
```

```
@@ -69,7 +69,8 @@ static match_table_t autofs_tokens = {
  {Opt_err, NULL}
 };

-static int parse_options(char *options, int *pipefd, uid_t *uid, gid_t *gid, pid_t *pgrp, int *minproto,
int *maxproto)
+static int parse_options(char *options, int *pipefd, uid_t *uid, gid_t *gid,
+  struct pid **pgrp, int *minproto, int *maxproto)
 {
  char *p;
  substring_t args[MAX_OPT_ARGS];
@@ -77,7 +78,7 @@ static int parse_options(char *options,

  *uid = current->uid;
  *gid = current->gid;
- *pgrp = process_group(current);
+ *pgrp = task_pgrp(current);

  *minproto = *maxproto = AUTOFS_PROTO_VERSION;

@@ -111,7 +112,7 @@ static int parse_options(char *options,
   case Opt_pgrp:
    if (match_int(&args[0], &option))
     return 1;
-   *pgrp = option;
+   *pgrp = find_pid(option);
    break;
   case Opt_minproto:
    if (match_int(&args[0], &option))
@@ -149,7 +150,7 @@ int autofs_fill_super(struct super_block
  sbi->pipe = NULL;
  sbi->catatonic = 1;
  sbi->exp_timeout = 0;
- sbi->oz_pgrp = process_group(current);
+ sbi->oz_pgrp = task_pgrp(current);
  autofs_initialize_hash(&sbi->dirhash);
  sbi->queues = NULL;
  memset(sbi->symlink_bitmap, 0, sizeof(long)*AUTOFS_SYMLINK_BITMAP_LEN);
@@ -169,7 +170,9 @@ int autofs_fill_super(struct super_block
   goto fail_iput;

  /* Can this call block?  - WTF cares? s is locked. */
- if (
parse_options(data,&pipefd,&root_inode->i_uid,&root_inode->i_gid,&sbi->oz_pgrp,&minproto,&m
axproto) ) {
+ if ( parse_options(data, &pipefd, &root_inode->i_uid,
+    &root_inode->i_gid, &sbi->oz_pgrp, &minproto,
+    &maxproto) ) {
```

```
   printk("autofs: called with bogus options\n");
   goto fail_dput;
  }
@@ -181,7 +184,8 @@ int autofs_fill_super(struct super_block
   goto fail_dput;
  }

- DPRINTK(("autofs: pipe fd = %d, pgrp = %u\n", pipefd, sbi->oz_pgrp));
+ DPRINTK(("autofs: pipe fd = %d, pgrp = %u\n", pipefd,
+     pid_nr(sbi->oz_pgrp)));
  pipe = fget(pipefd);

  if ( !pipe ) {
Index: lx26-20-rc4-mm1/fs/autofs4/inode.c
===================================================================
--- lx26-20-rc4-mm1.orig/fs/autofs4/inode.c 2007-01-19 17:29:25.000000000 -0800
+++ lx26-20-rc4-mm1/fs/autofs4/inode.c 2007-01-23 13:17:52.957181432 -0800
@@ -181,7 +181,7 @@ static int autofs4_show_options(struct s
   return 0;

  seq_printf(m, ",fd=%d", sbi->pipefd);
- seq_printf(m, ",pgrp=%d", sbi->oz_pgrp);
+ seq_printf(m, ",pgrp=%d", pid_nr(sbi->oz_pgrp));
  seq_printf(m, ",timeout=%lu", sbi->exp_timeout/HZ);
  seq_printf(m, ",minproto=%d", sbi->min_proto);
  seq_printf(m, ",maxproto=%d", sbi->max_proto);
@@ -218,7 +218,7 @@ static match_table_t tokens = {
 };

 static int parse_options(char *options, int *pipefd, uid_t *uid, gid_t *gid,
-     pid_t *pgrp, unsigned int *type,
+     struct pid **pgrp, unsigned int *type,
      int *minproto, int *maxproto)
 {
  char *p;
@@ -227,7 +227,7 @@ static int parse_options(char *options,

  *uid = current->uid;
  *gid = current->gid;
- *pgrp = process_group(current);
+ *pgrp = task_pgrp(current);

  *minproto = AUTOFS_MIN_PROTO_VERSION;
  *maxproto = AUTOFS_MAX_PROTO_VERSION;
@@ -261,7 +261,7 @@ static int parse_options(char *options,
   case Opt_pgrp:
    if (match_int(args, &option))
     return 1;
```

```
-   *pgrp = option;
+   *pgrp = find_pid(option);
    break;
   case Opt_minproto:
    if (match_int(args, &option))
@@ -326,7 +326,7 @@ int autofs4_fill_super(struct super_bloc
  sbi->pipe = NULL;
  sbi->catatonic = 1;
  sbi->exp_timeout = 0;
- sbi->oz_pgrp = process_group(current);
+ sbi->oz_pgrp = task_pgrp(current);
  sbi->sb = s;
  sbi->version = 0;
  sbi->sub_version = 0;
@@ -391,7 +391,7 @@ int autofs4_fill_super(struct super_bloc
   sbi->version = sbi->max_proto;
  sbi->sub_version = AUTOFS_PROTO_SUBVERSION;

- DPRINTK("pipe fd = %d, pgrp = %u", pipefd, sbi->oz_pgrp);
+ DPRINTK("pipe fd = %d, pgrp = %u", pipefd, pid_nr(sbi->oz_pgrp));
  pipe = fget(pipefd);

  if ( !pipe ) {
Index: lx26-20-rc4-mm1/fs/autofs/root.c
===================================================================
--- lx26-20-rc4-mm1.orig/fs/autofs/root.c 2007-01-19 17:29:25.000000000 -0800
+++ lx26-20-rc4-mm1/fs/autofs/root.c 2007-01-23 13:50:51.124454296 -0800
@@ -213,8 +213,10 @@ static struct dentry *autofs_root_lookup
  sbi = autofs_sbi(dir->i_sb);

  oz_mode = autofs_oz_mode(sbi);
- DPRINTK(("autofs_lookup: pid = %u, pgrp = %u, catatonic = %d, oz_mode = %d\n",
-   current->pid, process_group(current), sbi->catatonic, oz_mode));
+ DPRINTK(("autofs_lookup: pid = %u, pgrp = %u, catatonic = %d, "
+    "oz_mode = %d\n", pid_nr(task_pid(current)),
+    process_group(current), sbi->catatonic,
+    oz_mode));

  /*
   * Mark the dentry incomplete, but add it. This is needed so
Index: lx26-20-rc4-mm1/fs/autofs4/root.c
===================================================================
--- lx26-20-rc4-mm1.orig/fs/autofs4/root.c 2007-01-19 17:29:25.000000000 -0800
+++ lx26-20-rc4-mm1/fs/autofs4/root.c 2007-01-23 13:49:01.816071696 -0800
@@ -495,7 +495,8 @@ static struct dentry *autofs4_lookup(str
  oz_mode = autofs4_oz_mode(sbi);

  DPRINTK("pid = %u, pgrp = %u, catatonic = %d, oz_mode = %d",
```

```
-   current->pid, process_group(current), sbi->catatonic, oz_mode);
+   pid_nr(task_pid(current)), process_group(current),
+   sbi->catatonic, oz_mode);

  /*
   * Mark the dentry incomplete, but add it. This is needed so
Index: lx26-20-rc4-mm1/fs/autofs4/waitq.c
===================================================================
--- lx26-20-rc4-mm1.orig/fs/autofs4/waitq.c 2006-11-29 13:57:37.000000000 -0800
+++ lx26-20-rc4-mm1/fs/autofs4/waitq.c 2007-01-23 13:46:41.830352760 -0800
@@ -141,8 +141,8 @@ static void autofs4_notify_daemon(struct
   packet->ino = wq->ino;
   packet->uid = wq->uid;
   packet->gid = wq->gid;
-  packet->pid = wq->pid;
-  packet->tgid = wq->tgid;
+  packet->pid = pid_nr(wq->pid);
+  packet->tgid = pid_nr(wq->tgid);
   break;
  }
  default:
@@ -292,8 +292,8 @@ int autofs4_wait(struct autofs_sb_info *
   wq->ino = autofs4_get_ino(sbi);
   wq->uid = current->uid;
   wq->gid = current->gid;
-  wq->pid = current->pid;
-  wq->tgid = current->tgid;
+  wq->pid = task_pid(current);
+  wq->tgid = task_tgid(current);
   wq->status = -EINTR; /* Status return if interrupted */
   atomic_set(&wq->wait_ctr, 2);
   mutex_unlock(&sbi->wq_mutex);
```

Subject: Re: [RFC][PATCH] Use struct pid reference in autofs rather than pid_t
Posted by ebiederm on Wed, 24 Jan 2007 01:36:08 GMT
View Forum Message <> Reply to Message

Sukadev Bhattiprolu <sukadev@us.ibm.com> writes:

> From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
> Subject: Use struct pid reference in autofs rather than pid_t
>
> Make autofs container-friendly by caching struct pid reference rather

> than pid_t.

Are autofs and autofs4 so tightly tied they must be converted together?
If not 2 patches are more appropriate.

Otherwise I'd say this looks fine at first glance.

Just note that using struct pid protects from pid wraparound issues when
dealing with user space so this should also be a small increase in
correctness as well.

At second glance this patch is very incorrect. It is missing get_pid
and put_pid calls.

The big difference between struct pid and pid_t values is that
struct pid is reference counted.

Your find_pid's should be find_get_pid's and you need the put_pids
and unmount or whenever those values stop being valid.

You also didn't take the required locks when using find_pid.

Eric

_____
Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers

Subject: Re: [RFC][PATCH] Use struct pid reference in autofs rather than pid_t
Posted by Sukadev Bhattiprolu on Wed, 24 Jan 2007 06:55:52 GMT
View Forum Message <> Reply to Message

Yuck. My second glance did not catch any problems :-(.

Will fix the ref count and split into two patches.

Thanks,

Suka

Eric W. Biederman [ebiederm@xmission.com] wrote:
| Sukadev Bhattiprolu <sukadev@us.ibm.com> writes:
|
| > From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
| > Subject: Use struct pid reference in autofs rather than pid_t
| >
| > Make autofs container-friendly by caching struct pid reference rather

| > than pid_t.
|
| Are autofs and autofs4 so tightly tied they must be converted together?
| If not 2 patches are more appropriate.
|
| Otherwise I'd say this looks fine at first glance.
|
| Just note that using struct pid protects from pid wraparound issues when
| dealing with user space so this should also be a small increase in
| correctness as well.
|
| At second glance this patch is very incorrect. It is missing get_pid
| and put_pid calls.
|
| The big difference between struct pid and pid_t values is that
| struct pid is reference counted.
|
| Your find_pid's should be find_get_pid's and you need the put_pids
| and unmount or whenever those values stop being valid.
|
| You also didn't take the required locks when using find_pid.
|
| Eric

_____