
Subject: [PATCH 0/15] Make common helpers for seq_files that work with list_head-s (v2)

Posted by [xemul](#) on Fri, 18 May 2007 09:11:28 GMT

[View Forum Message](#) <> [Reply to Message](#)

Many places in kernel use seq_file API to iterate over a regular list_head. The code for such iteration is identical in all the places, so it's worth introducing a common helpers.

This makes code about 300 lines smaller:

```
block/genhd.c          | 40 +++-----
crypto/proc.c          | 17 ---
drivers/char/misc.c    | 18 ----
drivers/input/input.c  | 29 -----
drivers/isdn/capi/kcapi_proc.c | 28 -----
fs/afs/proc.c          | 81 ++-----
fs/namespace.c         | 14 ---
fs/nfs/client.c        | 54 +-----
fs/proc/proc_tty.c     | 15 ---
fs/seq_file.c          | 34 +++++++
include/linux/seq_file.h | 11 ++
kernel/module.c        | 17 ---
mm/slab.c              | 28 +-----
net/atm/br2684.c       | 22 ----
net/core/sock.c        | 39 -----
net/ipv4/netfilter/nf_conntrack_l3proto_ipv4_compat.c | 27 -----
net/netfilter/nf_conntrack_expect.c | 27 -----
net/rxrpc/ar-proc.c    | 48 +-----
18 files changed, 126 insertions(+), 423 deletions(-)
```

The first version of this patch made the helper functions static inline in the seq_file.h header. This patch moves them to the fs/seq_file.c as Andrew proposed. The vmlinux .text section sizes are as follows:

```
2.6.22-rc1-mm1:      0x001794d5
with the previous version: 0x00179505
with this patch:      0x00179135
```

The config file used was make allnoconfig with the "y" inclusion of all the possible options to make the files modified by the patch compile plus drivers I have on the test node.

Subject: [PATCH 1/15] The helper functions themselves

Posted by [xemul](#) on Fri, 18 May 2007 09:17:10 GMT

[View Forum Message](#) <> [Reply to Message](#)

Many places in kernel use seq_file API to iterate over a regular list_head. The code for such iteration is identical in all the places, so it's worth introducing a common helpers.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```
diff --git a/fs/seq_file.c b/fs/seq_file.c
index 0ac22af..49194a4 100644
--- a/fs/seq_file.c
+++ b/fs/seq_file.c
@@ -447,3 +447,37 @@ int seq_puts(struct seq_file *m, const c
    return -1;
}
EXPORT_SYMBOL(seq_puts);
+
+struct list_head *seq_list_start(struct list_head *head, loff_t pos)
+{
+    struct list_head *lh;
+
+    list_for_each(lh, head)
+        if (pos-- == 0)
+            return lh;
+
+    return NULL;
+}
+
+EXPORT_SYMBOL(seq_list_start);
+
+struct list_head *seq_list_start_head(struct list_head *head, loff_t pos)
+{
+    if (!pos)
+        return head;
+
+    return seq_list_start(head, pos - 1);
+}
+
+EXPORT_SYMBOL(seq_list_start_head);
+
+struct list_head *seq_list_next(void *v, struct list_head *head, loff_t *ppos)
+{
+    struct list_head *lh;
+
+    lh = ((struct list_head *)v)->next;
+    ++*ppos;
+    return lh == head ? NULL : lh;
+}
```

```

+
+EXPORT_SYMBOL(seq_list_next);
diff --git a/include/linux/seq_file.h b/include/linux/seq_file.h
index 3e3cccb..83783ab 100644
--- a/include/linux/seq_file.h
+++ b/include/linux/seq_file.h
@@ -50,5 +50,16 @@ int seq_release_private(struct inode *,

#define SEQ_START_TOKEN ((void *)1)

+/*
+ * Helpers for iteration over list_head-s in seq_files
+ */
+
+extern struct list_head *seq_list_start(struct list_head *head,
+ loff_t pos);
+extern struct list_head *seq_list_start_head(struct list_head *head,
+ loff_t pos);
+extern struct list_head *seq_list_next(void *v, struct list_head *head,
+ loff_t *ppos);
+
#endif
#endif

```

Subject: [PATCH 2/15] Make AFS use seq_list_xxx helpers
Posted by [xemul](#) on Fri, 18 May 2007 09:20:19 GMT
[View Forum Message](#) <> [Reply to Message](#)

These proc files show some header before dumping
the list, so the seq_list_start_head() is used.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```

diff --git a/fs/afs/proc.c b/fs/afs/proc.c
index d5601f6..d5300e4 100644
--- a/fs/afs/proc.c
+++ b/fs/afs/proc.c
@@ -200,23 +200,9 @@ static int afs_proc_cells_open(struct in
 */
static void *afs_proc_cells_start(struct seq_file *m, loff_t *_pos)
{
- struct list_head *_p;
- loff_t pos = *_pos;
-
/* lock the list against modification */

```

```

    down_read(&afs_proc_cells_sem);
-
- /* allow for the header line */
- if (!pos)
-     return (void *) 1;
- pos--;
-
- /* find the n'th element in the list */
- list_for_each(_p, &afs_proc_cells)
-     if (!pos--)
-         break;
-
- return _p != &afs_proc_cells ? _p : NULL;
+ return seq_list_start_head(&afs_proc_cells, *_pos);
}

/*
@@ -224,14 +210,7 @@ static void *afs_proc_cells_start(struct
*/
static void *afs_proc_cells_next(struct seq_file *p, void *v, loff_t *pos)
{
- struct list_head *_p;
-
- (*pos)++;
-
- _p = v;
- _p = v == (void *) 1 ? afs_proc_cells.next : _p->next;
-
- return _p != &afs_proc_cells ? _p : NULL;
+ return seq_list_next(v, &afs_proc_cells, pos);
}

/*
@@ -249,7 +228,7 @@ static int afs_proc_cells_show(struct se
{
    struct afs_cell *cell = list_entry(v, struct afs_cell, proc_link);

- if (v == (void *) 1) {
+ if (v == &afs_proc_cells) {
    /* display header on line 1 */
    seq_puts(m, "USE NAME\n");
    return 0;
@@ -502,26 +481,13 @@ static int afs_proc_cell_volumes_release
*/
static void *afs_proc_cell_volumes_start(struct seq_file *m, loff_t *_pos)
{
- struct list_head *_p;
- struct afs_cell *cell = m->private;

```

```

- loff_t pos = *_pos;

_enter("cell=%p pos=%Ld", cell, *_pos);

/* lock the list against modification */
down_read(&cell->vl_sem);
-
- /* allow for the header line */
- if (!pos)
- return (void *) 1;
- pos--;
-
- /* find the n'th element in the list */
- list_for_each(_p, &cell->vl_list)
- if (!pos--)
- break;
-
- return _p != &cell->vl_list ? _p : NULL;
+ return seq_list_start_head(&cell->vl_list, *_pos);
}

/*
@@ -530,17 +496,10 @@ static void *afs_proc_cell_volumes_start
static void *afs_proc_cell_volumes_next(struct seq_file *p, void *v,
    loff_t *_pos)
{
- struct list_head *_p;
- struct afs_cell *cell = p->private;

_enter("cell=%p pos=%Ld", cell, *_pos);
-
- (*_pos)++;
-
- _p = v;
- _p = (v == (void *) 1) ? cell->vl_list.next : _p->next;
-
- return (_p != &cell->vl_list) ? _p : NULL;
+ return seq_list_next(v, &cell->vl_list, _pos);
}

/*
@@ -568,11 +527,12 @@ const char afs_vlocation_states[][4] = {
*/
static int afs_proc_cell_volumes_show(struct seq_file *m, void *v)
{
+ struct afs_cell *cell = m->private;
+ struct afs_vlocation *vlocation =
+ list_entry(v, struct afs_vlocation, link);

```

```

/* display header on line 1 */
- if (v == (void *) 1) {
+ if (v == &cell->vl_list) {
    seq_puts(m, "USE STT VLID[0] VLID[1] VLID[2] NAME\n");
    return 0;
}
@@ -733,26 +693,13 @@ static int afs_proc_cell_servers_release
static void *afs_proc_cell_servers_start(struct seq_file *m, loff_t *_pos)
__acquires(m->private->servers_lock)
{
- struct list_head *_p;
- struct afs_cell *cell = m->private;
- loff_t pos = *_pos;

    _enter("cell=%p pos=%Ld", cell, *_pos);

    /* lock the list against modification */
    read_lock(&cell->servers_lock);
-
- /* allow for the header line */
- if (!pos)
- return (void *) 1;
- pos--;
-
- /* find the n'th element in the list */
- list_for_each(_p, &cell->servers)
- if (!pos--)
- break;
-
- return _p != &cell->servers ? _p : NULL;
+ return seq_list_start_head(&cell->servers, *_pos);
}

/*
@@ -761,17 +708,10 @@ static void *afs_proc_cell_servers_start
static void *afs_proc_cell_servers_next(struct seq_file *p, void *v,
    loff_t *_pos)
{
- struct list_head *_p;
- struct afs_cell *cell = p->private;

    _enter("cell=%p pos=%Ld", cell, *_pos);
-
- (*_pos)++;
-
- _p = v;
- _p = v == (void *) 1 ? cell->servers.next : _p->next;

```

```

-
- return _p != &cell->servers ? _p : NULL;
+ return seq_list_next(v, &cell->servers, _pos);
}

/*
@@ -790,11 +730,12 @@ static void afs_proc_cell_servers_stop(s
*/
static int afs_proc_cell_servers_show(struct seq_file *m, void *v)
{
+ struct afs_cell *cell = m->private;
  struct afs_server *server = list_entry(v, struct afs_server, link);
  char ipaddr[20];

  /* display header on line 1 */
- if (v == (void *) 1) {
+ if (v == &cell->servers) {
    seq_puts(m, "USE ADDR          STATE\n");
    return 0;
  }

```

Subject: [PATCH 3/15] Make ATM driver use seq_list_xxx helpers

Posted by [xemul](#) on Fri, 18 May 2007 09:23:28 GMT

[View Forum Message](#) <> [Reply to Message](#)

The .show callback receives the list_head pointer now, not the struct br2684_dev one.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```

diff --git a/net/atm/br2684.c b/net/atm/br2684.c
index 0e9f00c..3e26438 100644
--- a/net/atm/br2684.c
+++ b/net/atm/br2684.c
@@ -699,28 +699,13 @@ static struct atm_ioctl br2684_ioctl_ops
#ifdef CONFIG_PROC_FS
static void *br2684_seq_start(struct seq_file *seq, loff_t *pos)
{
- loff_t offs = 0;
- struct br2684_dev *brd;
-
  read_lock(&devs_lock);
-
- list_for_each_entry(brd, &br2684_devs, br2684_devs) {
- if (offs == *pos)

```

```

- return brd;
- ++offs;
- }
- return NULL;
+ return seq_list_start(&br2684_devs, *pos);
}

static void *br2684_seq_next(struct seq_file *seq, void *v, loff_t *pos)
{
- struct br2684_dev *brd = v;
-
- ++*pos;
-
- brd = list_entry(brd->br2684_devs.next,
-   struct br2684_dev, br2684_devs);
- return (&brd->br2684_devs != &br2684_devs) ? brd : NULL;
+ return seq_list_next(v, &br2684_devs, pos);
}

static void br2684_seq_stop(struct seq_file *seq, void *v)
@@ -730,7 +715,8 @@ static void br2684_seq_stop(struct seq_f

static int br2684_seq_show(struct seq_file *seq, void *v)
{
- const struct br2684_dev *brdev = v;
+ const struct br2684_dev *brdev = list_entry(v, struct br2684_dev,
+   br2684_devs);
  const struct net_device *net_dev = brdev->net_dev;
  const struct br2684_vcc *brvcc;

```

Subject: [PATCH 4/15] Make block layer /proc files use seq_list_xxx helpers
 Posted by [xemul](#) on Fri, 18 May 2007 09:27:29 GMT
[View Forum Message](#) <> [Reply to Message](#)

The /proc/partitions .show callback checked the *v to be the first element in list to show the header. Now *v is the struct list_head pointer and it is checked for the head of the list.

The comment in /proc/diskstats .show handler is also updated not to forget it in the future.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

diff --git a/block/genhd.c b/block/genhd.c

```

index 863a8c0..8813c14 100644
--- a/block/genhd.c
+++ b/block/genhd.c
@@ -270,22 +270,13 @@ void __init printk_all_partitions(void)
/* iterator */
static void *part_start(struct seq_file *part, loff_t *pos)
{
- struct list_head *p;
- loff_t l = *pos;
-
- mutex_lock(&block_subsys_lock);
- list_for_each(p, &block_subsys.list)
- if (!l--)
- return list_entry(p, struct gendisk, kobj.entry);
- return NULL;
+ return seq_list_start_head(&block_subsys.list, *pos);
}

static void *part_next(struct seq_file *part, void *v, loff_t *pos)
{
- struct list_head *p = ((struct gendisk *)v)->kobj.entry.next;
- ++*pos;
- return p==&block_subsys.list ? NULL :
- list_entry(p, struct gendisk, kobj.entry);
+ return seq_list_next(v, &block_subsys.list, pos);
}

static void part_stop(struct seq_file *part, void *v)
@@ -295,13 +286,16 @@ static void part_stop(struct seq_file *p

static int show_partition(struct seq_file *part, void *v)
{
- struct gendisk *sgp = v;
+ struct gendisk *sgp;
- int n;
- char buf[BDEVNAME_SIZE];

- if (&sgp->kobj.entry == block_subsys.list.next)
+ if (v == &block_subsys.list) {
- seq_puts(part, "major minor #blocks name\n\n");
+ seq_puts(part, "major minor #blocks name\n\n");
+ return 0;
+ }

+ sgp = list_entry(v, struct gendisk, kobj.entry);
/* Don't show non-partitionable removeable devices or empty devices */
if (!get_capacity(sgp) ||
(sgp->minors == 1 && (sgp->flags & GENHD_FL_REMOVABLE)))
@@ -622,22 +616,13 @@ decl_subsys(block, &ktype_block, &block_

```

```

/* iterator */
static void *diskstats_start(struct seq_file *part, loff_t *pos)
{
- loff_t k = *pos;
- struct list_head *p;
-
  mutex_lock(&block_subsys_lock);
- list_for_each(p, &block_subsys.list)
- if (!k--)
- return list_entry(p, struct gendisk, kobj.entry);
- return NULL;
+ return seq_list_start(&block_subsys.list, *pos);
}

static void *diskstats_next(struct seq_file *part, void *v, loff_t *pos)
{
- struct list_head *p = ((struct gendisk *)v)->kobj.entry.next;
- ++*pos;
- return p==&block_subsys.list ? NULL :
- list_entry(p, struct gendisk, kobj.entry);
+ return seq_list_next(v, &block_subsys.list, pos);
}

static void diskstats_stop(struct seq_file *part, void *v)
@@ -647,18 +632,21 @@ static void diskstats_stop(struct seq_file

static int diskstats_show(struct seq_file *s, void *v)
{
- struct gendisk *gp = v;
+ struct gendisk *gp;
  char buf[BDEVNAME_SIZE];
  int n = 0;

  /*
- if (&sgp->kobj.entry == block_subsys.kset.list.next)
+ if (v == &block_subsys.list) {
    seq_puts(s, "major minor name"
              "    rio rmerge rsect ruse wio wmerge "
              "wsect wuse running use aveq"
              "\n\n");
+ return 0;
+ }
  */

+ gp = list_entry(v, struct gendisk, kobj.entry);
  preempt_disable();
  disk_round_stats(gp);
  preempt_enable();

```

Subject: [PATCH 5/15] Make crypto API use seq_list_xxx helpers

Posted by [xemul](#) on Fri, 18 May 2007 09:29:41 GMT

[View Forum Message](#) <> [Reply to Message](#)

Simple and stupid - just use the same code from another place in the kernel.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

diff --git a/crypto/proc.c b/crypto/proc.c

index 102c751..3d73323 100644

--- a/crypto/proc.c

+++ b/crypto/proc.c

@@ -23,24 +23,13 @@

```
static void *c_start(struct seq_file *m, loff_t *pos)
{
- struct list_head *v;
- loff_t n = *pos;
-
  down_read(&crypto_alg_sem);
- list_for_each(v, &crypto_alg_list)
- if (!n--)
- return list_entry(v, struct crypto_alg, cra_list);
- return NULL;
+ return seq_list_start(&crypto_alg_list, *pos);
}
```

```
static void *c_next(struct seq_file *m, void *p, loff_t *pos)
{
- struct list_head *v = p;
-
- (*pos)++;
- v = v->next;
- return (v == &crypto_alg_list) ?
- NULL : list_entry(v, struct crypto_alg, cra_list);
+ return seq_list_next(p, &crypto_alg_list, pos);
}
```

```
static void c_stop(struct seq_file *m, void *p)
```

```
@@ -50,7 +39,7 @@ static void c_stop(struct seq_file *m, v
```

```
static int c_show(struct seq_file *m, void *p)
{
- struct crypto_alg *alg = (struct crypto_alg *)p;
+ struct crypto_alg *alg = list_entry(p, struct crypto_alg, cra_list);
```

```
seq_printf(m, "name      : %s\n", alg->cra_name);
seq_printf(m, "driver    : %s\n", alg->cra_driver_name);
```

Subject: [PATCH 6/15] Make input layer use seq_list_xxx helpers

Posted by [xemul](#) on Fri, 18 May 2007 09:33:28 GMT

[View Forum Message](#) <> [Reply to Message](#)

This is essentially just a renaming of the existing functions as the seq_list_start() and seq_list_next() copies already exist in the input.c file. Now we have them in the generic place.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

diff --git a/drivers/input/input.c b/drivers/input/input.c

index ccd8aba..da83c17 100644

--- a/drivers/input/input.c

+++ b/drivers/input/input.c

@@ -471,37 +471,16 @@ static unsigned int input_proc_devices_p

return 0;

}

-static struct list_head *list_get_nth_element(struct list_head *list, loff_t *pos)

-{

- struct list_head *node;

- loff_t i = 0;

-

- list_for_each(node, list)

- if (i++ == *pos)

- return node;

-

- return NULL;

-}

-

-static struct list_head *list_get_next_element(struct list_head *list, struct list_head *element, loff_t *pos)

-{

- if (element->next == list)

- return NULL;

-

- ++(*pos);

- return element->next;

-}

-

static void *input_devices_seq_start(struct seq_file *seq, loff_t *pos)

```

{
/* acquire lock here ... Yes, we do need locking, I knowi, I know... */

- return list_get_nth_element(&input_dev_list, pos);
+ return seq_list_start(&input_dev_list, *pos);
}

static void *input_devices_seq_next(struct seq_file *seq, void *v, loff_t *pos)
{
- return list_get_next_element(&input_dev_list, v, pos);
+ return seq_list_next(v, &input_dev_list, pos);
}

static void input_devices_seq_stop(struct seq_file *seq, void *v)
@@ -592,13 +571,13 @@ static void *input_handlers_seq_start(st
{
/* acquire lock here ... Yes, we do need locking, I knowi, I know... */
seq->private = (void *) (unsigned long) *pos;
- return list_get_nth_element(&input_handler_list, pos);
+ return seq_list_start(&input_handler_list, *pos);
}

static void *input_handlers_seq_next(struct seq_file *seq, void *v, loff_t *pos)
{
seq->private = (void *) (unsigned long) (*pos + 1);
- return list_get_next_element(&input_handler_list, v, pos);
+ return seq_list_next(v, &input_handler_list, pos);
}

static void input_handlers_seq_stop(struct seq_file *seq, void *v)

```

Subject: [PATCH 7/15] Make ISDN CAPI use seq_list_xxx helpers

Posted by [xemul](#) on Fri, 18 May 2007 09:36:24 GMT

[View Forum Message](#) <> [Reply to Message](#)

The similar code exists here and is called capi_driver_get_idx().
Use generic helpers now and remember to convert list_head to
struct capi_driver in .show callback.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```

diff --git a/drivers/isdn/capi/kcapi_proc.c b/drivers/isdn/capi/kcapi_proc.c
index 31f4fd8..845a797 100644
--- a/drivers/isdn/capi/kcapi_proc.c
+++ b/drivers/isdn/capi/kcapi_proc.c

```

@@ -243,36 +243,15 @@ create_seq_entry(char *name, mode_t mode

```
// -----  
  
-  
-static __inline__ struct capi_driver *capi_driver_get_idx(loff_t pos)  
-  
-  
- {  
- struct capi_driver *drv = NULL;  
- struct list_head *l;  
- loff_t i;  
-  
- i = 0;  
- list_for_each(l, &capi_drivers) {  
- drv = list_entry(l, struct capi_driver, list);  
- if (i++ == pos)  
- return drv;  
- }  
- return NULL;  
-}  
-  
static void *capi_driver_start(struct seq_file *seq, loff_t *pos)  
{  
- struct capi_driver *drv;  
  read_lock(&capi_drivers_list_lock);  
- drv = capi_driver_get_idx(*pos);  
- return drv;  
+ return seq_list_start(&capi_drivers, *pos);  
}  
  
static void *capi_driver_next(struct seq_file *seq, void *v, loff_t *pos)  
{  
- struct capi_driver *drv = (struct capi_driver *)v;  
- ++*pos;  
- if (drv->list.next == &capi_drivers) return NULL;  
- return list_entry(drv->list.next, struct capi_driver, list);  
+ return seq_list_next(v, &capi_drivers, pos);  
}  
  
static void capi_driver_stop(struct seq_file *seq, void *v)  
@@ -282,7 +261,8 @@ static void capi_driver_stop(struct seq_  
  
static int capi_driver_show(struct seq_file *seq, void *v)  
{  
- struct capi_driver *drv = (struct capi_driver *)v;  
+ struct capi_driver *drv = list_entry(v, struct capi_driver, list);  
+  
  seq_printf(seq, "%-32s %s\n", drv->name, drv->revision);  
  return 0;  
}
```

}

Subject: [PATCH 8/15] Make /proc/misc use seq_list_xxx helpers

Posted by [xemul](#) on Fri, 18 May 2007 09:39:02 GMT

[View Forum Message](#) <> [Reply to Message](#)

Simple and stupid - just use the helpers.

Sorry for no maintainer in Cc: - I haven't found any.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

diff --git a/drivers/char/misc.c b/drivers/char/misc.c

index 4e6fb96..71c8cd7 100644

--- a/drivers/char/misc.c

+++ b/drivers/char/misc.c

@@ -67,25 +67,13 @@ extern int pmu_device_init(void);

#ifdef CONFIG_PROC_FS

static void *misc_seq_start(struct seq_file *seq, loff_t *pos)

{

- struct miscdevice *p;

- loff_t off = 0;

-

mutex_lock(&misc_mtx);

- list_for_each_entry(p, &misc_list, list) {

- if (*pos == off++)

- return p;

- }

- return NULL;

+ return seq_list_start(&misc_list, *pos);

}

static void *misc_seq_next(struct seq_file *seq, void *v, loff_t *pos)

{

- struct list_head *n = ((struct miscdevice *)v)->list.next;

-

- ++*pos;

-

- return (n != &misc_list) ? list_entry(n, struct miscdevice, list)

- : NULL;

+ return seq_list_next(v, &misc_list, pos);

}

static void misc_seq_stop(struct seq_file *seq, void *v)

@@ -95,7 +83,7 @@ static void misc_seq_stop(struct seq_fil

```
static int misc_seq_show(struct seq_file *seq, void *v)
{
- const struct miscdevice *p = v;
+ const struct miscdevice *p = list_entry(v, struct miscdevice, list);

seq_printf(seq, "%3i %s\n", p->minor, p->name ? p->name : "");
return 0;

```

Subject: [PATCH 9/15] Make /proc/modules use seq_list_xxx helpers
 Posted by [xemul](#) on Fri, 18 May 2007 09:41:34 GMT

[View Forum Message](#) <> [Reply to Message](#)

Here there is not need even in .show callback altering.
 The original code passes list_head in *v.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```
diff --git a/kernel/module.c b/kernel/module.c
index 015d60c..7a1a4d3 100644
--- a/kernel/module.c
+++ b/kernel/module.c
@@ -2232,26 +2232,13 @@ unsigned long module_kallsyms_lookup_name
/* Called by the /proc file system to return a list of modules. */
static void *m_start(struct seq_file *m, loff_t *pos)
{
- struct list_head *i;
- loff_t n = 0;
-
mutex_lock(&module_mutex);
- list_for_each(i, &modules) {
- if (n++ == *pos)
- break;
- }
- if (i == &modules)
- return NULL;
- return i;
+ return seq_list_start(&modules, *pos);
}

static void *m_next(struct seq_file *m, void *p, loff_t *pos)
{
- struct list_head *i = p;
- (*pos)++;
- if (i->next == &modules)

```

```
- return NULL;
- return i->next;
+ return seq_list_next(p, &modules, pos);
}
```

```
static void m_stop(struct seq_file *m, void *p)
```

Subject: [PATCH 10/15] Make some network-related proc files use seq_list_xxx helpers

Posted by [xemul](#) on Fri, 18 May 2007 09:48:30 GMT

[View Forum Message](#) <> [Reply to Message](#)

This includes /proc/net/protocols, /proc/net/rxrpc_calls and /proc/net/rxrpc_connections files.

All three need seq_list_start_head to show some header.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```
diff --git a/net/core/sock.c b/net/core/sock.c
```

```
index 22183c2..528f65b 100644
```

```
--- a/net/core/sock.c
```

```
+++ b/net/core/sock.c
```

```
@@ -1839,46 +1839,15 @@ void proto_unregister(struct proto *prot
EXPORT_SYMBOL(proto_unregister);
```

```
#ifdef CONFIG_PROC_FS
```

```
-static inline struct proto *__proto_head(void)
```

```
-{
```

```
- return list_entry(proto_list.next, struct proto, node);
```

```
-}
```

```
-
```

```
-static inline struct proto *proto_head(void)
```

```
-{
```

```
- return list_empty(&proto_list) ? NULL : __proto_head();
```

```
-}
```

```
-
```

```
-static inline struct proto *proto_next(struct proto *proto)
```

```
-{
```

```
- return proto->node.next == &proto_list ? NULL :
```

```
- list_entry(proto->node.next, struct proto, node);
```

```
-}
```

```
-
```

```
-static inline struct proto *proto_get_idx(loff_t pos)
```

```
-{
```

```

- struct proto *proto;
- loff_t i = 0;
-
- list_for_each_entry(proto, &proto_list, node)
- if (i++ == pos)
- goto out;
-
- proto = NULL;
-out:
- return proto;
-}
-
static void *proto_seq_start(struct seq_file *seq, loff_t *pos)
{
    read_lock(&proto_list_lock);
- return *pos ? proto_get_idx(*pos - 1) : SEQ_START_TOKEN;
+ return seq_list_start_head(&proto_list, *pos);
}

static void *proto_seq_next(struct seq_file *seq, void *v, loff_t *pos)
{
- ++*pos;
- return v == SEQ_START_TOKEN ? proto_head() : proto_next(v);
+ return seq_list_next(v, &proto_list, pos);
}

static void proto_seq_stop(struct seq_file *seq, void *v)
@@ -1926,7 +1895,7 @@ static void proto_seq_printf(struct seq_

static int proto_seq_show(struct seq_file *seq, void *v)
{
- if (v == SEQ_START_TOKEN)
+ if (v == &proto_list)
    seq_printf(seq, "%-9s %-4s %-8s %-6s %-5s %-7s %-4s %-10s %s",
        "protocol",
        "size",
@@ -1938,7 +1907,7 @@ static int proto_seq_show(struct seq_fil
    "module",
    "cl co di ac io in de sh ss gs se re sp bi br ha uh gp em\n");
else
- proto_seq_printf(seq, v);
+ proto_seq_printf(seq, list_entry(v, struct proto, node));
return 0;
}

diff --git a/net/rxrpc/ar-proc.c b/net/rxrpc/ar-proc.c
index 1c0be0e..56a5317 100644
--- a/net/rxrpc/ar-proc.c

```

```

+++ b/net/rxrpc/ar-proc.c
@@ -30,31 +30,13 @@ static const char *rxrpc_conn_states[] =
 */
static void *rxrpc_call_seq_start(struct seq_file *seq, loff_t *_pos)
{
- struct list_head *_p;
- loff_t pos = *_pos;
-
- read_lock(&rxrpc_call_lock);
- if (!pos)
- return SEQ_START_TOKEN;
- pos--;
-
- list_for_each(_p, &rxrpc_calls)
- if (!pos--)
- break;
-
- return _p != &rxrpc_calls ? _p : NULL;
+ return seq_list_start_head(&rxrpc_calls, *_pos);
}

static void *rxrpc_call_seq_next(struct seq_file *seq, void *v, loff_t *pos)
{
- struct list_head *_p;
-
- (*pos)++;
-
- _p = v;
- _p = (v == SEQ_START_TOKEN) ? rxrpc_calls.next : _p->next;
-
- return _p != &rxrpc_calls ? _p : NULL;
+ return seq_list_next(v, &rxrpc_calls, pos);
}

static void rxrpc_call_seq_stop(struct seq_file *seq, void *v)
@@ -68,7 +50,7 @@ static int rxrpc_call_seq_show(struct se
struct rxrpc_call *call;
char lbuff[4 + 4 + 4 + 4 + 5 + 1], rbuff[4 + 4 + 4 + 4 + 5 + 1];

- if (v == SEQ_START_TOKEN) {
+ if (v == &rxrpc_calls) {
seq_puts(seq,
"Proto Local          Remote          "
" SvID ConnID  CallID  End Use State  Abort  "
@@ -129,32 +111,14 @@ struct file_operations rxrpc_call_seq_fo
*/
static void *rxrpc_connection_seq_start(struct seq_file *seq, loff_t *_pos)
{

```

```

- struct list_head *_p;
- loff_t pos = *_pos;
-
  read_lock(&rxrpc_connection_lock);
- if (!pos)
-   return SEQ_START_TOKEN;
- pos--;
-
- list_for_each(_p, &rxrpc_connections)
-   if (!pos--)
-     break;
-
- return _p != &rxrpc_connections ? _p : NULL;
+ return seq_list_start_head(&rxrpc_connections, *_pos);
}

static void *rxrpc_connection_seq_next(struct seq_file *seq, void *v,
                                       loff_t *pos)
{
- struct list_head *_p;
-
- (*pos)++;
-
- _p = v;
- _p = (v == SEQ_START_TOKEN) ? rxrpc_connections.next : _p->next;
-
- return _p != &rxrpc_connections ? _p : NULL;
+ return seq_list_next(v, &rxrpc_connections, pos);
}

static void rxrpc_connection_seq_stop(struct seq_file *seq, void *v)
@@ -168,7 +132,7 @@ static int rxrpc_connection_seq_show(str
  struct rxrpc_transport *trans;
  char lbuff[4 + 4 + 4 + 4 + 5 + 1], rbuff[4 + 4 + 4 + 4 + 5 + 1];

- if (v == SEQ_START_TOKEN) {
+ if (v == &rxrpc_connections) {
  seq_puts(seq,
    "Proto Local      Remote      "
    " SvID ConnID  Calls  End Use State  Key  "

```

Subject: [PATCH 11/15] Make some netfilter-related proc files use seq_list_xxx helpers

Posted by [xemul](#) on Fri, 18 May 2007 09:55:32 GMT

[View Forum Message](#) <> [Reply to Message](#)

This includes /proc/net/ip_conntrack_expect file.

Although struct nf_conntrack_expect has list_head as the very first element I use list_entry in .show callback to emphasize the fact that *v is the list_head pointer.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```
diff --git a/net/ipv4/netfilter/nf_conntrack_l3proto_ipv4_compat.c
b/net/ipv4/netfilter/nf_conntrack_l3proto_ipv4_compat.c
index 89f933e..bec843a 100644
```

```
--- a/net/ipv4/netfilter/nf_conntrack_l3proto_ipv4_compat.c
+++ b/net/ipv4/netfilter/nf_conntrack_l3proto_ipv4_compat.c
@@ -208,35 +208,15 @@ static const struct file_operations ct_f
/* expects */
```

```
static void *exp_seq_start(struct seq_file *s, loff_t *pos)
{
```

```
- struct list_head *e = &nf_conntrack_expect_list;
- loff_t i;
```

```
-
```

```
/* strange seq_file api calls stop even if we fail,
 * thus we need to grab lock since stop unlocks */
read_lock_bh(&nf_conntrack_lock);
```

```
-
```

```
- if (list_empty(e))
- return NULL;
```

```
-
```

```
- for (i = 0; i <= *pos; i++) {
- e = e->next;
- if (e == &nf_conntrack_expect_list)
- return NULL;
- }
```

```
- return e;
+ return seq_list_start(&nf_conntrack_expect_list, *pos);
}
```

```
static void *exp_seq_next(struct seq_file *s, void *v, loff_t *pos)
{
```

```
- struct list_head *e = v;
```

```
-
```

```
- ++*pos;
- e = e->next;
```

```
-
```

```
- if (e == &nf_conntrack_expect_list)
- return NULL;
```

```
-
```

```
- return e;
```

```

+ return seq_list_next(v, &nf_conntrack_expect_list, pos);
}

static void exp_seq_stop(struct seq_file *s, void *v)
@@ -246,7 +226,8 @@ static void exp_seq_stop(struct seq_file

static int exp_seq_show(struct seq_file *s, void *v)
{
- struct nf_conntrack_expect *exp = v;
+ struct nf_conntrack_expect *exp = list_entry(v,
+ struct nf_conntrack_expect, list);

if (exp->tuple.src.l3num != AF_INET)
return 0;
diff --git a/net/netfilter/nf_conntrack_expect.c b/net/netfilter/nf_conntrack_expect.c
index 117cbfd..7ea80e4 100644
--- a/net/netfilter/nf_conntrack_expect.c
+++ b/net/netfilter/nf_conntrack_expect.c
@@ -366,35 +366,15 @@ EXPORT_SYMBOL_GPL(nf_conntrack_expect_re
#ifdef CONFIG_PROC_FS
static void *exp_seq_start(struct seq_file *s, loff_t *pos)
{
- struct list_head *e = &nf_conntrack_expect_list;
- loff_t i;
-
/* strange seq_file api calls stop even if we fail,
 * thus we need to grab lock since stop unlocks */
read_lock_bh(&nf_conntrack_lock);
-
- if (list_empty(e))
- return NULL;
-
- for (i = 0; i <= *pos; i++) {
- e = e->next;
- if (e == &nf_conntrack_expect_list)
- return NULL;
- }
- return e;
+ return seq_list_start(&nf_conntrack_expect_list, *pos);
}

static void *exp_seq_next(struct seq_file *s, void *v, loff_t *pos)
{
- struct list_head *e = v;
-
- ++*pos;
- e = e->next;
-

```

```

- if (e == &nf_contrack_expect_list)
- return NULL;
-
- return e;
+ return seq_list_next(v, &nf_contrack_expect_list, pos);
}

static void exp_seq_stop(struct seq_file *s, void *v)
@@ -404,7 +384,8 @@ static void exp_seq_stop(struct seq_file

static int exp_seq_show(struct seq_file *s, void *v)
{
- struct nf_contrack_expect *expect = v;
+ struct nf_contrack_expect *expect = list_entry(v,
+ struct nf_contrack_expect, list);

if (expect->timeout.function)
seq_printf(s, "%ld ", timer_pending(&expect->timeout)

```

Subject: Re: [PATCH 1/15] The helper functions themselves
 Posted by [David Howells](#) on Fri, 18 May 2007 09:59:08 GMT
[View Forum Message](#) <> [Reply to Message](#)

Pavel Emelianov <xemul@sw.ru> wrote:

> Many places in kernel use seq_file API to iterate over a regular
 > list_head. The code for such iteration is identical in all the
 > places, so it's worth introducing a common helpers.

And the documentation for this?

David

Subject: [PATCH 12/15] Make NFS client use seq_list_xxx helpers
 Posted by [xemul](#) on Fri, 18 May 2007 10:00:14 GMT
[View Forum Message](#) <> [Reply to Message](#)

This includes /proc/fs/nfsfs/servers and /proc/fs/nfsfs/volumes
 entries.

Both need to show the header and use the list_head.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```

diff --git a/fs/nfs/client.c b/fs/nfs/client.c
index 50c6821..10355ec 100644
--- a/fs/nfs/client.c
+++ b/fs/nfs/client.c
@@ -1232,23 +1232,9 @@ static int nfs_server_list_open(struct i
 */
static void *nfs_server_list_start(struct seq_file *m, loff_t *_pos)
{
- struct list_head *_p;
- loff_t pos = *_pos;
-
- /* lock the list against modification */
- spin_lock(&nfs_client_lock);
-
- /* allow for the header line */
- if (!pos)
- return SEQ_START_TOKEN;
- pos--;
-
- /* find the n'th element in the list */
- list_for_each(_p, &nfs_client_list)
- if (!pos--)
- break;
-
- return _p != &nfs_client_list ? _p : NULL;
+ return seq_list_start_head(&nfs_client_list, *_pos);
}

/*
@@ -1256,14 +1242,7 @@ static void *nfs_server_list_start(struc
 */
static void *nfs_server_list_next(struct seq_file *p, void *v, loff_t *pos)
{
- struct list_head *_p;
-
- (*pos)++;
-
- _p = v;
- _p = (v == SEQ_START_TOKEN) ? nfs_client_list.next : _p->next;
-
- return _p != &nfs_client_list ? _p : NULL;
+ return seq_list_next(v, &nfs_client_list, pos);
}

/*
@@ -1282,7 +1261,7 @@ static int nfs_server_list_show(struct s
struct nfs_client *clp;

```

```

/* display header on line 1 */
- if (v == SEQ_START_TOKEN) {
+ if (v == &nfs_client_list) {
    seq_puts(m, "NV SERVER  PORT USE HOSTNAME\n");
    return 0;
}
@@ -1323,23 +1302,9 @@ static int nfs_volume_list_open(struct i
*/
static void *nfs_volume_list_start(struct seq_file *m, loff_t *_pos)
{
- struct list_head *_p;
- loff_t pos = *_pos;
-
/* lock the list against modification */
spin_lock(&nfs_client_lock);
-
/* allow for the header line */
- if (!pos)
- return SEQ_START_TOKEN;
- pos--;
-
/* find the n'th element in the list */
- list_for_each(_p, &nfs_volume_list)
- if (!pos--)
- break;
-
- return _p != &nfs_volume_list ? _p : NULL;
+ return seq_list_start_head(&nfs_volume_list, *_pos);
}

/*
@@ -1347,14 +1312,7 @@ static void *nfs_volume_list_start(struc
*/
static void *nfs_volume_list_next(struct seq_file *p, void *v, loff_t *pos)
{
- struct list_head *_p;
-
- (*pos)++;
-
- _p = v;
- _p = (v == SEQ_START_TOKEN) ? nfs_volume_list.next : _p->next;
-
- return _p != &nfs_volume_list ? _p : NULL;
+ return seq_list_next(v, &nfs_volume_list, pos);
}

/*

```

```

@@ -1375,7 +1333,7 @@ static int nfs_volume_list_show(struct s
    char dev[8], fsid[17];

    /* display header on line 1 */
- if (v == SEQ_START_TOKEN) {
+ if (v == &nfs_volume_list) {
    seq_puts(m, "NV SERVER  PORT DEV  FSID\n");
    return 0;
}

```

Subject: [PATCH 13/15] Make /proc/tty/drivers use seq_list_xxx helpers
 Posted by [xemul](#) on Fri, 18 May 2007 10:02:45 GMT
[View Forum Message](#) <> [Reply to Message](#)

Simple and stupid like some previous ones. Just use new API.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```

diff --git a/fs/proc/proc_tty.c b/fs/proc/proc_tty.c
index b3a473b..2284622 100644
--- a/fs/proc/proc_tty.c
+++ b/fs/proc/proc_tty.c
@@ -69,7 +69,7 @@ static void show_tty_range(struct seq_file

static int show_tty_driver(struct seq_file *m, void *v)
{
- struct tty_driver *p = v;
+ struct tty_driver *p = list_entry(v, struct tty_driver, tty_drivers);
  dev_t from = MKDEV(p->major, p->minor_start);
  dev_t to = from + p->num;

@@ -106,22 +106,13 @@ static int show_tty_driver(struct seq_file
/* iterator */
static void *t_start(struct seq_file *m, loff_t *pos)
{
- struct list_head *p;
- loff_t l = *pos;
-
  mutex_lock(&tty_mutex);
- list_for_each(p, &tty_drivers)
- if (!l--)
-   return list_entry(p, struct tty_driver, tty_drivers);
- return NULL;
+ return seq_list_start(&tty_drivers, *pos);
}

```

```
static void *t_next(struct seq_file *m, void *v, loff_t *pos)
{
- struct list_head *p = ((struct tty_driver *)v)->tty_drivers.next;
- (*pos)++;
- return p==&tty_drivers ? NULL :
- list_entry(p, struct tty_driver, tty_drivers);
+ return seq_list_next(v, &tty_drivers, pos);
}
```

```
static void t_stop(struct seq_file *m, void *v)
```

Subject: [PATCH 14/15] Make /proc/slabinfo use seq_list_xxx helpers

Posted by [xemul](#) on Fri, 18 May 2007 10:06:51 GMT

[View Forum Message](#) <> [Reply to Message](#)

This entry prints a header in .start callback. This is OK, but the more elegant solution would be to move this into the .show callback and use seq_list_start_head() in .start one.

I have left it as is in order to make the patch just switch to new API and noting more.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```
diff --git a/mm/slab.c b/mm/slab.c
```

```
index 6edfd34..faad0d7 100644
```

```
--- a/mm/slab.c
```

```
+++ b/mm/slab.c
```

```
@@ -4161,26 +4161,17 @@ static void print_slabinfo_header(struct
```

```
static void *s_start(struct seq_file *m, loff_t *pos)
```

```
{
    loff_t n = *pos;
```

```
- struct list_head *p;
```

```
    mutex_lock(&cache_chain_mutex);
```

```
    if (!n)
```

```
        print_slabinfo_header(m);
```

```
- p = cache_chain.next;
```

```
- while (n--) {
```

```
-     p = p->next;
```

```
-     if (p == &cache_chain)
```

```
-         return NULL;
```

```
- }
```

```
- return list_entry(p, struct kmem_cache, next);
```

```

+
+ return seq_list_start(&cache_chain, *pos);
}

static void *s_next(struct seq_file *m, void *p, loff_t *pos)
{
- struct kmem_cache *cachep = p;
- ++*pos;
- return cachep->next.next == &cache_chain ?
- NULL : list_entry(cachep->next.next, struct kmem_cache, next);
+ return seq_list_next(p, &cache_chain, pos);
}

static void s_stop(struct seq_file *m, void *p)
@@ -4190,7 +4181,7 @@ static void s_stop(struct seq_file *m, v

static int s_show(struct seq_file *m, void *p)
{
- struct kmem_cache *cachep = p;
+ struct kmem_cache *cachep = list_entry(p, struct kmem_cache, next);
  struct slab *slabp;
  unsigned long active_objs;
  unsigned long num_objs;
@@ -4359,17 +4350,8 @@ ssize_t slabinfo_write(struct file *file

static void *leaks_start(struct seq_file *m, loff_t *pos)
{
- loff_t n = *pos;
- struct list_head *p;
-
- mutex_lock(&cache_chain_mutex);
- p = cache_chain.next;
- while (n--) {
-   p = p->next;
-   if (p == &cache_chain)
-     return NULL;
- }
- return list_entry(p, struct kmem_cache, next);
+ return seq_list_start(&cache_chain, *pos);
}

static inline int add_caller(unsigned long *n, unsigned long v)

```

Subject: [PATCH 15/15] Make /proc/self/mounts(tats) use seq_list_xxx helpers
 Posted by [xemul](#) on Fri, 18 May 2007 10:10:03 GMT
[View Forum Message](#) <> [Reply to Message](#)

One more simple and stupid switching to the new API.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```
diff --git a/fs/namespace.c b/fs/namespace.c
index 11f2fbf..82d4fd2 100644
--- a/fs/namespace.c
+++ b/fs/namespace.c
@@ -393,22 +393,16 @@ EXPORT_SYMBOL(mnt_unpin);
static void *m_start(struct seq_file *m, loff_t *pos)
{
    struct mnt_namespace *n = m->private;
- struct list_head *p;
- loff_t l = *pos;

    down_read(&namespace_sem);
- list_for_each(p, &n->list)
- if (!l--)
- return list_entry(p, struct vfsmount, mnt_list);
- return NULL;
+ return seq_list_start(&n->list, *pos);
}

static void *m_next(struct seq_file *m, void *v, loff_t *pos)
{
    struct mnt_namespace *n = m->private;
- struct list_head *p = ((struct vfsmount *)v)->mnt_list.next;
- (*pos)++;
- return p == &n->list ? NULL : list_entry(p, struct vfsmount, mnt_list);
+
+ return seq_list_next(v, &n->list, pos);
}

static void m_stop(struct seq_file *m, void *v)
@@ -423,7 +417,7 @@ static inline void mangle(struct seq_fil

static int show_vfsmnt(struct seq_file *m, void *v)
{
- struct vfsmount *mnt = v;
+ struct vfsmount *mnt = list_entry(v, struct vfsmount, mnt_list);
    int err = 0;
    static struct proc_fs_info {
        int flag;
@@ -481,7 +475,7 @@ struct seq_operations mounts_op = {

static int show_vfsstat(struct seq_file *m, void *v)
```

```
{  
- struct vfsmount *mnt = v;  
+ struct vfsmount *mnt = list_entry(v, struct vfsmount, mnt_list);  
  int err = 0;  
  
  /* device */
```

Subject: Re: [PATCH 5/15] Make crypto API use seq_list_xxx helpers
Posted by [Herbert Xu](#) on Fri, 18 May 2007 10:13:28 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Fri, May 18, 2007 at 01:34:23PM +0400, Pavel Emelianov wrote:
> Simple and stupid - just use the same code from another
> place in the kernel.
>
> Signed-off-by: Pavel Emelianov <xemul@openvz.org>

Ack.

--

Visit Openswan at <http://www.openswan.org/>
Email: Herbert Xu ~{PmV>Hl~} <herbert@gondor.apana.org.au>
Home Page: <http://gondor.apana.org.au/~herbert/>
PGP Key: <http://gondor.apana.org.au/~herbert/pubkey.txt>

Subject: Re: [PATCH 10/15] Make some network-related proc files use seq_list_xxx helpers
Posted by [davem](#) on Fri, 18 May 2007 10:24:20 GMT
[View Forum Message](#) <> [Reply to Message](#)

From: Pavel Emelianov <xemul@sw.ru>
Date: Fri, 18 May 2007 13:53:13 +0400

> This includes /proc/net/protocols, /proc/net/rxrpc_calls
> and /proc/net/rxrpc_connections files.
>
> All three need seq_list_start_head to show some header.
>
> Signed-off-by: Pavel Emelianov <xemul@openvz.org>

Acked-by: David S. Miller <davem@davemloft.net>

Subject: Re: [PATCH 7/15] Make ISDN CAPI use seq_list_xxx helpers
Posted by [Karsten Keil](#) on Fri, 18 May 2007 11:10:07 GMT

The similar code exists here and is called capi_driver_get_idx().
Use generic helpers now and remember to convert list_head to struct capi_driver in .show callback.

Acked-by: Karsten Keil <kkeil@suse.de>

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

diff --git a/drivers/isdn/capi/kcapi_proc.c b/drivers/isdn/capi/kcapi_proc.c
index 31f4fd8..845a797 100644

--- a/drivers/isdn/capi/kcapi_proc.c

+++ b/drivers/isdn/capi/kcapi_proc.c

@@ -243,36 +243,15 @@ create_seq_entry(char *name, mode_t mode

```
// -----  
  
-  
-static __inline__ struct capi_driver *capi_driver_get_idx(loff_t pos)  
-  
-  
- struct capi_driver *drv = NULL;  
- struct list_head *l;  
- loff_t i;  
-  
- i = 0;  
- list_for_each(l, &capi_drivers) {  
-   drv = list_entry(l, struct capi_driver, list);  
-   if (i++ == pos)  
-     return drv;  
- }  
- return NULL;  
-}  
-  
static void *capi_driver_start(struct seq_file *seq, loff_t *pos)  
{  
- struct capi_driver *drv;  
  read_lock(&capi_drivers_list_lock);  
- drv = capi_driver_get_idx(*pos);  
- return drv;  
+ return seq_list_start(&capi_drivers, *pos);  
}  
  
static void *capi_driver_next(struct seq_file *seq, void *v, loff_t *pos)  
{  
- struct capi_driver *drv = (struct capi_driver *)v;  
- ++*pos;  
- if (drv->list.next == &capi_drivers) return NULL;
```

```

- return list_entry(drv->list.next, struct capi_driver, list);
+ return seq_list_next(v, &capi_drivers, pos);
}

static void capi_driver_stop(struct seq_file *seq, void *v)
@@ -282,7 +261,8 @@ static void capi_driver_stop(struct seq_

static int capi_driver_show(struct seq_file *seq, void *v)
{
- struct capi_driver *drv = (struct capi_driver *)v;
+ struct capi_driver *drv = list_entry(v, struct capi_driver, list);
+
seq_printf(seq, "%-32s %s\n", drv->name, drv->revision);
return 0;
}

```

--
Karsten Keil
SuSE Labs
ISDN and VOIP development
SUSE LINUX Products GmbH, Maxfeldstr.5 90409 Nuernberg, GF: Markus Rex, HRB 16746 (AG
Nuernberg)

Subject: Re: [PATCH 12/15] Make NFS client use seq_list_xxx helpers
Posted by [Trond Myklebust](#) on Fri, 18 May 2007 17:42:02 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Fri, 2007-05-18 at 14:04 +0400, Pavel Emelianov wrote:
> This includes /proc/fs/nfsfs/servers and /proc/fs/nfsfs/volumes
> entries.
>
> Both need to show the header and use the list_head.
>
> Signed-off-by: Pavel Emelianov <xemul@openvz.org>

Acked-by: Trond Myklebust <Trond.Myklebust@netapp.com>

```

>
> ---
>
> diff --git a/fs/nfs/client.c b/fs/nfs/client.c
> index 50c6821..10355ec 100644
> --- a/fs/nfs/client.c
> +++ b/fs/nfs/client.c
> @@ -1232,23 +1232,9 @@ static int nfs_server_list_open(struct i
> */

```

```

> static void *nfs_server_list_start(struct seq_file *m, loff_t *_pos)
> {
> - struct list_head *_p;
> - loff_t pos = *_pos;
> -
> - /* lock the list against modification */
> - spin_lock(&nfs_client_lock);
> -
> - /* allow for the header line */
> - if (!pos)
> - return SEQ_START_TOKEN;
> - pos--;
> -
> - /* find the n'th element in the list */
> - list_for_each(_p, &nfs_client_list)
> - if (!pos--)
> - break;
> -
> - return _p != &nfs_client_list ? _p : NULL;
> + return seq_list_start_head(&nfs_client_list, *_pos);
> }
>
> /*
> @@ -1256,14 +1242,7 @@ static void *nfs_server_list_start(struct
> */
> static void *nfs_server_list_next(struct seq_file *p, void *v, loff_t *pos)
> {
> - struct list_head *_p;
> -
> - (*pos)++;
> -
> - _p = v;
> - _p = (v == SEQ_START_TOKEN) ? nfs_client_list.next : _p->next;
> -
> - return _p != &nfs_client_list ? _p : NULL;
> + return seq_list_next(v, &nfs_client_list, pos);
> }
>
> /*
> @@ -1282,7 +1261,7 @@ static int nfs_server_list_show(struct s
> struct nfs_client *clp;
>
> - /* display header on line 1 */
> - if (v == SEQ_START_TOKEN) {
> + if (v == &nfs_client_list) {
> seq_puts(m, "NV SERVER PORT USE HOSTNAME\n");
> return 0;
> }

```

```

> @@ -1323,23 +1302,9 @@ static int nfs_volume_list_open(struct i
> */
> static void *nfs_volume_list_start(struct seq_file *m, loff_t *_pos)
> {
> - struct list_head *_p;
> - loff_t pos = *_pos;
> -
> /* lock the list against modification */
> spin_lock(&nfs_client_lock);
> -
> - /* allow for the header line */
> - if (!pos)
> - return SEQ_START_TOKEN;
> - pos--;
> -
> - /* find the n'th element in the list */
> - list_for_each(_p, &nfs_volume_list)
> - if (!pos--)
> - break;
> -
> - return _p != &nfs_volume_list ? _p : NULL;
> + return seq_list_start_head(&nfs_volume_list, *_pos);
> }
>
> /*
> @@ -1347,14 +1312,7 @@ static void *nfs_volume_list_start(struc
> */
> static void *nfs_volume_list_next(struct seq_file *p, void *v, loff_t *pos)
> {
> - struct list_head *_p;
> -
> - (*pos)++;
> -
> - _p = v;
> - _p = (v == SEQ_START_TOKEN) ? nfs_volume_list.next : _p->next;
> -
> - return _p != &nfs_volume_list ? _p : NULL;
> + return seq_list_next(v, &nfs_volume_list, pos);
> }
>
> /*
> @@ -1375,7 +1333,7 @@ static int nfs_volume_list_show(struct s
> char dev[8], fsid[17];
>
> /* display header on line 1 */
> - if (v == SEQ_START_TOKEN) {
> + if (v == &nfs_volume_list) {
> seq_puts(m, "NV SERVER  PORT DEV  FSID\n");

```

```
> return 0;  
> }  
>
```
