

freebsd-arch: REVIEW: handling kldload/kldunload of GEOM classes properly.

REVIEW: handling kldload/kldunload of GEOM classes properly.

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2004-10/0191.html>

From: Poul-Henning Kamp (*phk_at_phk.freebsd.dk*)

Date: 10/22/04

To: arch@freebsd.org

Date: Fri, 22 Oct 2004 23:42:48 +0200

There are a number of system calls which starts events running in GEOM and which for consistency should not return to userland until those events have completed and GEOM has settled down.

The typical example would be a shell-script doing:

```
kldload md
mdconfig bla bla
```

Examples of such syscalls are:

open(2) & close(2): can cause spoils and retastes

mount(2)/umount(2): acts as open/close.

(any other syscall doing a VOP_OPEN/VOP_CLOSE on a diskdevice.)

ioctl(2): directed configuration can do almost anything.

kldload(2)/kldunload(2): can load/unload GEOM classes.

The last one is the most tricky one: The crude solution is to always wait for geom to settle before returning to userland from kldload(2), but there is no point in waiting for GEOM if you loaded a USB serial driver and doing so would increase the risk of cascade failures.

A further complication is that we should not wait for geom to settle until after we have dropped Giant again because the geom events might need to grab giant to do their job.

The cleanest way to solve this once and for all is to add a thread private flag bit, set it on curthread whenever a GEOM event is posted, and check that geom has cleaned all events before that thread is allowed to return to userland. (I properly can be persuaded

REVIEW: handling kldload/kldunload of GEOM classes properly.

freebsd-arch: REVIEW: handling kldload/kldunload of GEOM classes properly.

that the sleep should be interruptible).

That strategy implements the semantics which POLA would suggest "Return when you have done all you know you need to do".

It does not address the issue where a scsi driver needs "camcontrol rescan" to actually look for its disks or where the disk arrives later when it is plugged in, but it does not return to userland while we know we are only half finished.

Doing it with a thread private flag costs us one bit check in the syscall return-path, and normally that would have ruled it right out for me.

After looking at all the gunk we have there, and spotting at least two opportunities for microoptimizations which will offset this extra test, I decided that the cost was less than epsilon.

Patch attached, comments requested.

A number of g_waitidle() calls can subsequently be removed, but didn't want to clutter the patch with that.

Poul-Henning

Index: sys/proc.h

```
=====
RCS file: /home/ncvs/src/sys/sys/proc.h,v
retrieving revision 1.410
diff -u -r1.410 proc.h
--- sys/proc.h 16 Oct 2004 06:38:21 -0000 1.410
+++ sys/proc.h 22 Oct 2004 21:19:26 -0000
@@ -375,6 +375,7 @@
#define TDP_SCHED2 0x00002000 /* Reserved for scheduler private use */
#define TDP_SCHED3 0x00004000 /* Reserved for scheduler private use */
#define TDP_SCHED4 0x00008000 /* Reserved for scheduler private use */
+#define TDP_GEOM 0x00010000 /* Settle GEOM before finishing syscall */

/*
 * Reasons that the current thread can not be run yet.
Index: sys/system.h
```

```
=====
RCS file: /home/ncvs/src/sys/sys/system.h,v
retrieving revision 1.215
diff -u -r1.215 system.h
--- sys/system.h 30 Sep 2004 07:04:03 -0000 1.215
+++ sys/system.h 22 Oct 2004 21:22:12 -0000
@@ -127,6 +127,7 @@
void hashdestroy(void *, struct malloc_type *, u_long);
void *hashinit(int count, struct malloc_type *type, u_long *hashmask);
void *phashinit(int count, struct malloc_type *type, u_long *nentries);
```

REVIEW: handling kldload/kldunload of GEOM classes properly.

freebsd-arch: REVIEW: handling kldload/kldunload of GEOM classes properly.

```
+void g_waitidle(void);
```

```
#ifdef RESTARTABLE_PANICS
```

```
void panic(const char *, ...) __printflike(1, 2);
```

```
Index: geom/geom.h
```

```
RCS file: /home/ncvs/src/sys/geom/geom.h,v
```

```
retrieving revision 1.85
```

```
diff -u -r1.85 geom.h
```

```
--- geom/geom.h 27 Aug 2004 14:43:11 -0000 1.85
```

```
+++ geom/geom.h 22 Oct 2004 21:22:42 -0000
```

```
@@ -213,7 +213,6 @@
```

```
int g_waitfor_event(g_event_t *func, void *arg, int flag, ...);
```

```
void g_cancel_event(void *ref);
```

```
void g_orphan_provider(struct g_provider *pp, int error);
```

```
-void g_waitidle(void);
```

```
/* geom_subr.c */
```

```
int g_access(struct g_consumer *cp, int nread, int nwrite, int nexcl);
```

```
Index: geom/geom_event.c
```

```
RCS file: /home/ncvs/src/sys/geom/geom_event.c,v
```

```
retrieving revision 1.50
```

```
diff -u -r1.50 geom_event.c
```

```
--- geom/geom_event.c 8 Jul 2004 16:17:14 -0000 1.50
```

```
+++ geom/geom_event.c 22 Oct 2004 21:19:45 -0000
```

```
@@ -47,12 +47,14 @@
```

```
#include <sys/kernel.h>
```

```
#include <sys/lock.h>
```

```
#include <sys/mutex.h>
```

```
‑#include <machine/stdarg.h>
```

```
+#include <sys/proc.h>
```

```
#include <sys/errno.h>
```

```
#include <sys/time.h>
```

```
#include <geom/geom.h>
```

```
#include <geom/geom_int.h>
```

```
+#include <machine/stdarg.h>
```

```
+
```

```
TAILQ_HEAD(event_tailq_head, g_event);
```

```
static struct event_tailq_head g_events = TAILQ_HEAD_INITIALIZER(g_events);
```

```
@@ -279,6 +281,7 @@
```

```
    wakeup(&g_wait_event);
```

```
    if (epp != NULL)
```

```
        *epp = ep;
```

```
+ curthread->td_pflags |= TDP_GEOM;
```

```
    return (0);
```

```
}
```

```
Index: kern/subr_trap.c
```

REVIEW: handling kldload/kldunload of GEOM classes properly.

freebsd-arch: REVIEW: handling kldload/kldunload of GEOM classes properly.

RCS file: /home/ncvs/src/sys/kern/subr_trap.c,v

retrieving revision 1.275

diff -u -r1.275 subr_trap.c

--- kern/subr_trap.c 5 Oct 2004 18:51:11 -0000 1.275

+++ kern/subr_trap.c 22 Oct 2004 21:39:40 -0000

@@ -95,6 +95,15 @@

#endif

/*

+ * If this thread tickled GEOM, we need to wait for the giggling to

+ * stop before we return to userland

+ */

+ if (td->td_pflags & TDP_GEOM) {

+ td->td_pflags &= ~TDP_GEOM;

+ g_waitidle();

+ }

+

+ /*

 * Let the scheduler adjust our priority etc.

 */

 sched_userret(td);

freebsd-arch@freebsd.org mailing list

<http://lists.freebsd.org/mailman/listinfo/freebsd-arch>

To unsubscribe, send any mail to "freebsd-arch-unsubscribe@freebsd.org"