
Subject: Re: [PATCH 1/2] signal checkpoint: define /proc/pid/sig/
Posted by [Cedric Le Goater](#) on Tue, 12 Jun 2007 16:30:46 GMT
[View Forum Message](#) <> [Reply to Message](#)

[...]

```
>>> +static int print_sigpending_alloc(char **bufp, struct sigpending *pending)
>>> +{
>>> + int allocated=0;
>>> + char *buf, *p;
>>> + struct sigqueue *q;
>>> + struct siginfo *info;
>>> +
>>> + allocated = PAGE_SIZE;
>>> + p = buf = kmalloc(allocated, GFP_KERNEL);
>>> + if (!buf)
>>> + return -ENOMEM;
>>> +
>>> + p += print_sigset(buf, &pending->signal);
>>> + p += sprintf(p, "\n");
>>> +
>>> + list_for_each_entry(q, &pending->list, list) {
>>> + info = &q->info;
>>> + if (p-buf+215 > allocated) {
>>> + int len=p-buf;
>>> + char *buf2;
>>> + allocated += PAGE_SIZE;
>>> + buf2 = kmalloc(allocated, GFP_KERNEL);
>>> + if (!buf2) {
>>> + kfree(buf);
>>> + return -ENOMEM;
>>> + }
>>> + memcpy(buf2, buf, allocated - PAGE_SIZE);
>>> + kfree(buf);
>>> + buf = buf2;
>>> + p = buf+len;
>>> + }
>>> +
>>> + p += sprintf(p, "sig %d: user %d flags %d",
>>> + info->si_signo, (int)q->user->uid, q->flags);
>>> + p += sprintf(p, " errno %d code %d\n",
>>> + info->si_errno, info->si_code);
>>> +
>>> + switch(info->si_signo) {
>>> + case SIGKILL:
>>> + p += sprintf(p, " spid %d suid %d\n",
>>> + info->_sifields._kill._pid,
>>> + info->_sifields._kill._uid);
```

```

>>> + break;
>>> + /* XXX skipping posix1b timers and signals for now */
>>> + case SIGCHLD:
>>> + p += sprintf(p, " pid %d uid %d status %d utime %lu stime %lu\n",
>>> + info->_sifields._sigchld._pid,
>>> + info->_sifields._sigchld._uid,
>>> + info->_sifields._sigchld._status,
>>> + info->_sifields._sigchld._utime,
>>> + info->_sifields._sigchld._stime);
>>> + break;
>>> + case SIGILL:
>>> + case SIGFPE:
>>> + case SIGSEGV:
>>> + case SIGBUS:
>>> + #ifdef __ARCH_SI_TRAPNO
>>> + p += sprintf(p, " addr %lu trapno %d\n",
>>> + (unsigned long)info->_sifields._sigfault._addr,
>>> + info->_sifields._sigfault._trapno);
>>> + #else
>>> + p += sprintf(p, " addr %lu\n",
>>> + (unsigned long)info->_sifields._sigfault._addr);
>>> + #endif
>>> + break;
>>> + case SIGPOLL:
>>> + p += sprintf(p, " band %ld fd %d\n",
>>> + (long)info->_sifields._sigpoll._band,
>>> + info->_sifields._sigpoll._fd);
>>> + break;
>>> + default:
>>> + p += sprintf(p, " Unsupported siginfo for signal %d\n",
>>> + info->si_signo);
>>> + break;
>>> + }
>>> + }
>>> + *bufp = buf;
>>> + return p-buf;
>>> +}
>> I think we are reaching the limit of /proc when we expose the pending siginfos.
>
> Why?

```

well, we are really exposing the internals of signal delivery, which are not that useful for /proc. IMO, it would be better to use an "opaque" to get/set the data we need.

C.

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