

```

*****
103076 Thu Jan 17 14:57:24 2019
new/usr/src/lib/fm/topo/modules/common/ses/ses.c
10115 ses topo module needs smatch fixes
*****
_____unchanged_portion_omitted_____

459 /*ARGSUSED*/
460 static void *
461 ses_contract_thread(void *arg)
462 {
463     int efd, ctlfd, statfd;
464     ct_evthdl_t ev;
465     ctevid_t evid;
466     uint_t event;
467     char path[PATH_MAX];
468     char buf[80];
469     ses_enum_target_t *stp;
470     ct_stathdl_t stathdl;
471     ctid_t ctid;
472     struct pollfd fds;
473     int pollret;
474     sigset_t sigset;

476     ses_ct_print("start contract event thread");
477     efd = open64(CTFS_ROOT "/device/pbundle", O_RDONLY);
478     fds.fd = efd;
479     fds.events = POLLIN;
480     fds.revents = 0;
481     (void) sigaddset(&sigset, sesthread.thr_sig);
482     (void) pthread_sigmask(SIG_UNBLOCK, &sigset, NULL);
481     sigaddset(&sigset, sesthread.thr_sig);
482     pthread_sigmask(SIG_UNBLOCK, &sigset, NULL);
483     for (;;) {
484         /* check if we've been asked to exit */
485         (void) pthread_mutex_lock(&sesthread.mt);
486         if (sesthread.doexit) {
487             (void) pthread_mutex_unlock(&sesthread.mt);
488             break;
489         }
490         (void) pthread_mutex_unlock(&sesthread.mt);

492         /* poll until an event arrives */
493         if ((pollret = poll(&fds, 1, 10000)) <= 0) {
494             if (pollret == 0)
495                 ses_recheck_dir();
496             continue;
497         }

499         /* read the event */
500         (void) pthread_mutex_lock(&ses_sslmt);
501         ses_ct_print("read contract event");
502         if (ct_event_read(efd, &ev) != 0) {
503             (void) pthread_mutex_unlock(&ses_sslmt);
504             continue;
505         }

507         /* see if it is an event we are expecting */
508         ctid = ct_event_get_ctid(ev);
509         (void) snprintf(buf, sizeof (buf),
510             "got contract event ctid=%d", ctid);
511         ses_ct_print(buf);
512         event = ct_event_get_type(ev);
513         if (event != CT_DEV_EV_OFFLINE && event != CT_EV_NEGEND) {
514             (void) snprintf(buf, sizeof (buf),
515                 "bad contract event %x", event);

```

```

516         ses_ct_print(buf);
517         ct_event_free(ev);
518         (void) pthread_mutex_unlock(&ses_sslmt);
519         continue;
520     }

522     /* find target pointer saved in cookie */
523     evid = ct_event_get_evid(ev);
524     (void) snprintf(path, PATH_MAX, CTFS_ROOT "/device/%ld/status",
525         ctid);
526     statfd = open64(path, O_RDONLY);
527     (void) ct_status_read(statfd, CTD_COMMON, &stathdl);
528     stp = (ses_enum_target_t *) (uintptr_t)
529         ct_status_get_cookie(stathdl);
530     ct_status_free(stathdl);
531     (void) close(statfd);

533     /* check if target pointer is still valid */
534     if (ses_ssl_valid(stp) == 0) {
535         (void) snprintf(buf, sizeof (buf),
536             "contract already abandoned %x", event);
537         ses_ct_print(buf);
538         (void) snprintf(path, PATH_MAX,
539             CTFS_ROOT "/device/%ld/ctl", ctid);
540         ctlfd = open64(path, O_WRONLY);
541         if (event != CT_EV_NEGEND)
542             (void) ct_ctl_ack(ctlfd, evid);
543         else
544             (void) ct_ctl_abandon(ctlfd);
545         (void) close(ctlfd);
546         ct_event_free(ev);
547         (void) pthread_mutex_unlock(&ses_sslmt);
548         continue;
549     }

551     /* find control device for ack/abandon */
552     (void) pthread_mutex_lock(&stp->set_lock);
553     (void) snprintf(path, PATH_MAX, CTFS_ROOT "/device/%ld/ctl",
554         ctid);
555     ctlfd = open64(path, O_WRONLY);
556     if (event != CT_EV_NEGEND) {
557         /* if this is an offline event, do the offline */
558         ses_ct_print("got contract offline event");
559         if (stp->set_target) {
560             ses_ct_print("contract thread rele");
561             ses_snap_rele(stp->set_snap);
562             ses_close(stp->set_target);
563             stp->set_target = NULL;
564         }
565         (void) ct_ctl_ack(ctlfd, evid);
566     } else {
567         /* if this is the negend, then abandon the contract */
568         ses_ct_print("got contract negend");
569         if (stp->set_ctid) {
570             (void) snprintf(buf, sizeof (buf),
571                 "abandon old contract %d", stp->set_ctid);
572             ses_ct_print(buf);
573             stp->set_ctid = NULL;
574         }
575         (void) ct_ctl_abandon(ctlfd);
576     }
577     (void) close(ctlfd);
578     (void) pthread_mutex_unlock(&stp->set_lock);
579     ct_event_free(ev);
580     (void) pthread_mutex_unlock(&ses_sslmt);
581 }

```

new/usr/src/lib/fm/topo/modules/common/ses/ses.c

3

```
582     (void) close(efd);  
583     return (NULL);  
584 }  
_____unchanged_portion_omitted_____
```