```
**********************************************************
    7984 Thu Jun  5 21:16:33 2014
new/usr/src/lib/libcryptoutil/common/cryptoutil.h
1667 pkcs11 may deadlock when multi-threaded consumers fork
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 *
 * Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
 */
/*
 * Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
 * Copyright 2014, OmniTI Computer Consulting, Inc. All rights reserved.
 */

#ifndef _CRYPTOUTIL_H
#define _CRYPTOUTIL_H

#ifdef __cplusplus
extern "C" {
#endif

#include <sys/types.h>
#include <syslog.h>
#include <security/cryptoki.h>
#include <sys/param.h>

#define LOG_STDERR      -1
#define SUCCESS         0
#define FAILURE         1
#define MECH_ID_HEX_LEN 11      /* length of mechanism id in hex form */

#define _PATH_PKCS11_CONF       "/etc/crypto/pkcs11.conf"
#define _PATH_KCF_CONF          "/etc/crypto/kcf.conf"
#define _PATH_KCFD_LOCK         "/var/run/kcfd.lock"

/* $ISA substitution for parsing pkcs11.conf data */
#define PKCS11_ISA      "/$ISA/"
#if defined(_LP64)
#define PKCS11_ISA_DIR  "/64/"
#else   /* !_LP64 */
#define PKCS11_ISA_DIR  "/"
#endif

/* keywords and delimiters for parsing configuration files */
#define SEP_COLON       ":"
#define SEP_SEMICOLON   ";"
#define SEP_EQUAL       "="
#define SEP_COMMA       ","
```



```
#define METASLOT_KEYWORD        "metaslot"
#define FIPS_KEYWORD    "fips-140"
#define EF_DISABLED     "disabledlist="
#define EF_ENABLED      "enabledlist="
#define EF_NORANDOM     "NO_RANDOM"
#define METASLOT_TOKEN  "metaslot_token="
#define METASLOT_SLOT   "metaslot_slot="
#define METASLOT_STATUS "metaslot_status="
#define EF_FIPS_STATUS  "fips_status="
#define METASLOT_AUTO_KEY_MIGRATE       "metaslot_auto_key_migrate="
#define ENABLED_KEYWORD         "enabled"
#define DISABLED_KEYWORD        "disabled"
#define SLOT_DESCRIPTION_SIZE   64
#define TOKEN_LABEL_SIZE        32
#define TOKEN_MANUFACTURER_SIZE 32
#define TOKEN_SERIAL_SIZE       16
#define CRYPTO_FIPS_MODE_DISABLED       0
#define CRYPTO_FIPS_MODE_ENABLED        1

/*
 * Define the following softtoken values that are used by softtoken
 * library, cryptoadm and pktool command.
 */
#define SOFT_SLOT_DESCRIPTION   \
                        "Sun Crypto Softtoken            " \
                        "                                "
#define SOFT_TOKEN_LABEL        "Sun Software PKCS#11 softtoken  "
#define SOFT_TOKEN_SERIAL       "                "
#define SOFT_MANUFACTURER_ID    "Sun Microsystems, Inc.          "
#define SOFT_DEFAULT_PIN        "changeme"

typedef char libname_t[MAXPATHLEN];
typedef char midstr_t[MECH_ID_HEX_LEN];

typedef struct umechlist {
        midstr_t                name;   /* mechanism name in hex form */
        struct umechlist        *next;
} umechlist_t;
```

unchanged_portion_omitted_

```
extern void cryptodebug(const char *fmt, ...);
extern void cryptoerror(int priority, const char *fmt, ...);
extern void cryptodebug_init(const char *prefix);
extern void cryptoerror_off();
extern void cryptoerror_on();

extern const char *pkcs11_mech2str(CK_MECHANISM_TYPE mech);
extern CK_RV pkcs11_str2mech(char *mech_str, CK_MECHANISM_TYPE_PTR mech);

extern int get_pkcs11conf_info(uentrylist_t **);
extern umechlist_t *create_umech(char *);
extern void free_umechlist(umechlist_t *);
extern void free_uentrylist(uentrylist_t *);
extern void free_uentry(uentry_t *);
extern uentry_t *getent_uef(char *);

extern void tohexstr(uchar_t *bytes, size_t blen, char *hexstr, size_t hexlen);
extern int hexstr_to_bytes(char *hexstr, size_t hexlen, uchar_t **bytes,
    size_t *blen);
extern CK_RV pkcs11_mech2keytype(CK_MECHANISM_TYPE mech_type,
    CK_KEY_TYPE *ktype);
extern CK_RV pkcs11_mech2keygen(CK_MECHANISM_TYPE mech_type,
    CK_MECHANISM_TYPE *gen_mech);
extern char *pkcs11_strerror(CK_RV rv);

extern int
```

```
 206 get_metaslot_info(boolean_t  *status_enabled, boolean_t *migrate_enabled,
 207     char **objectstore_slot_info, char **objectstore_token_info);

 209 extern char *get_fullpath(char *dir, char *filepath);
 210 extern int str2lifetime(char *ltimestr, uint32_t *ltime);

 212 extern char *pkcs11_default_token(void);
 213 extern int pkcs11_get_pass(char *token_name, char **pdata, size_t *psize,
 214     size_t min_psize, boolean_t with_confirmation);

 216 extern int pkcs11_seed_urandom(void *sbuf, size_t slen);
 217 extern int pkcs11_get_random(void *dbuf, size_t dlen);
 218 extern int pkcs11_get_urandom(void *dbuf, size_t dlen);
 219 extern int pkcs11_get_nzero_urandom(void *dbuf, size_t dlen);
 219 extern void pkcs11_close_random(void);
 220 extern void pkcs11_close_urandom(void);
 221 extern void pkcs11_close_urandom_seed(void);
 220 extern int pkcs11_read_data(char *filename, void **dbuf, size_t *dlen);

 222 extern int open_nointr(const char *path, int oflag, ...);
 223 extern ssize_t readn_nointr(int fd, void *dbuf, size_t dlen);
 224 extern ssize_t writen_nointr(int fd, void *dbuf, size_t dlen);
 225 extern int update_conf(char *conf_file, char *entry);

 227 extern int pkcs11_parse_uri(const char *str, pkcs11_uri_t *uri);
 228 extern void pkcs11_free_uri(pkcs11_uri_t *uri);

 230 #ifdef __cplusplus
 231 }
```

_____***unchanged_portion_omitted_***

```
**********************************************************
    1990 Thu Jun  5 21:16:34 2014
new/usr/src/lib/libcryptoutil/common/mapfile-vers
1667 pkcs11 may deadlock when multi-threaded consumers fork
**********************************************************
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
# Copyright (c) 2006, 2010, Oracle and/or its affiliates. All rights reserved.
# Copyright 2014, OmniTI Computer Consulting Inc. All rights reserved.
#

#
# MAPFILE HEADER START
#
# WARNING:  STOP NOW.  DO NOT MODIFY THIS FILE.
# Object versioning must comply with the rules detailed in
#
#       usr/src/lib/README.mapfiles
#
# You should not be making modifications here until you've read the most current
# copy of that file. If you need help, contact a gatekeeper for guidance.
#
# MAPFILE HEADER END
#

$mapfile_version 2

SYMBOL_VERSION SUNWprivate {
    global:
        create_umech;
        cryptodebug;
        cryptodebug_init;
        cryptoerror;
        cryptoerror_off;
        cryptoerror_on;
        free_uentry;
        free_uentrylist;
        free_umechlist;
        getent_uef;
        get_fullpath;
        get_metaslot_info;
        get_pkcs11conf_info;
        hexstr_to_bytes;
        open_nointr;
        pkcs11_close_random;
        pkcs11_close_urandom;
        pkcs11_close_urandom_seed;
        pkcs11_default_token;
```



```
        pkcs11_free_uri;
        pkcs11_get_nzero_urandom;
        pkcs11_get_pass;
        pkcs11_get_random;
        pkcs11_get_urandom;
        pkcs11_mech2keytype;
        pkcs11_mech2keygen;
        pkcs11_mech2str;
        pkcs11_parse_uri;
        pkcs11_read_data;
        pkcs11_seed_random;
        pkcs11_seed_urandom;
        pkcs11_str2mech;
        pkcs11_strerror;
        readn_nointr;
        str2lifetime;
        tohexstr;
        writen_nointr;
    local:
        *;
};
______unchanged_portion_omitted_
```

```
**********************************************************
    9151 Thu Jun  5 21:16:34 2014
new/usr/src/lib/libcryptoutil/common/random.c
1667 pkcs11 may deadlock when multi-threaded consumers fork
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */

/*
 * Copyright (c) 2003, 2010, Oracle and/or its affiliates. All rights reserved.
 * Copyright 2014, OmniTI Computer Consulting, Inc. All rights reserved.
 */

#include <stdio.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>
#include <fcntl.h>
#include <locale.h>
#include <stdarg.h>
#include <cryptoutil.h>
#include <pthread.h>

#pragma init(pkcs11_random_init)

static pthread_mutex_t  random_mutex = PTHREAD_MUTEX_INITIALIZER;
static pthread_mutex_t  urandom_mutex = PTHREAD_MUTEX_INITIALIZER;

static pthread_mutex_t  random_seed_mutex = PTHREAD_MUTEX_INITIALIZER;
static pthread_mutex_t  urandom_seed_mutex = PTHREAD_MUTEX_INITIALIZER;

#define RANDOM_DEVICE           "/dev/random"   /* random device name */
#define URANDOM_DEVICE          "/dev/urandom"  /* urandom device name */

static int      random_fd = -1;
static int      urandom_fd = -1;

static int      random_seed_fd = -1;
static int      urandom_seed_fd = -1;


/*
 * Equivalent of open(2) insulated from EINTR.
 * Also sets close-on-exec.
 */
int
open_nointr(const char *path, int oflag, ...)
{
```



```
        int     fd;
        mode_t  pmode;
        va_list alist;

        va_start(alist, oflag);
        pmode = va_arg(alist, mode_t);
        va_end(alist);

        do {
                if ((fd = open(path, oflag, pmode)) >= 0) {
                        (void) fcntl(fd, F_SETFD, FD_CLOEXEC);
                        break;
                }
                /* errno definitely set by failed open() */
        } while (errno == EINTR);
        return (fd);
}
______unchanged_portion_omitted_

static void
void
pkcs11_close_random(void)
{
        pkcs11_close_common(&random_fd, &random_mutex);
}

static void
void
pkcs11_close_urandom(void)
{
        pkcs11_close_common(&urandom_fd, &urandom_mutex);
}
______unchanged_portion_omitted_

static void
void
pkcs11_close_urandom_seed(void)
{
        pkcs11_close_common(&urandom_seed_fd, &urandom_seed_mutex);
}
______unchanged_portion_omitted_

/*
 * Same as pkcs11_get_urandom but ensures non zero data.
 */
int
pkcs11_get_nzero_urandom(void *dbuf, size_t dlen)
{
        char    extrarand[32];
        size_t  bytesleft = 0;
        size_t  i = 0;

        /* Start with some random data */
        if (pkcs11_get_urandom(dbuf, dlen) < 0)
                return (-1);

        /* Walk through data replacing any 0 bytes with more random data */
        while (i < dlen) {
                if (((char *)dbuf)[i] != 0) {
                        i++;
                        continue;
                }

                if (bytesleft == 0) {
                        bytesleft = sizeof (extrarand);
                        if (pkcs11_get_urandom(extrarand, bytesleft) < 0)
```

```
374                                     return (-1);
375                 }
376                 bytesleft--;

378                 ((char *)dbuf)[i] = extrarand[bytesleft];
379         }
380         return (0);
381 }

383 static void
384 pkcs11_random_prepare(void)
385 {
386         /*
387          * NOTE - None of these are acquired more than one at a time.
388          * I can therefore acquire all four without fear of deadlock.
389          */
390         (void) pthread_mutex_lock(&random_mutex);
391         (void) pthread_mutex_lock(&urandom_mutex);
392         (void) pthread_mutex_lock(&random_seed_mutex);
393         (void) pthread_mutex_lock(&urandom_seed_mutex);
394 }

396 static void
397 pkcs11_random_parent_post(void)
398 {
399         /* Drop the mutexes and get back to work! */
400         (void) pthread_mutex_unlock(&urandom_seed_mutex);
401         (void) pthread_mutex_unlock(&random_seed_mutex);
402         (void) pthread_mutex_unlock(&urandom_mutex);
403         (void) pthread_mutex_unlock(&random_mutex);
404 }

406 static void
407 pkcs11_random_child_post(void)
408 {
409         pkcs11_random_parent_post();

411         /* Also, close the FDs, just in case. */
412         pkcs11_close_random();
413         pkcs11_close_urandom();
414         pkcs11_close_random_seed();
415         pkcs11_close_urandom_seed();
416 }

418 static void
419 pkcs11_random_init(void)
420 {
421         (void) pthread_atfork(pkcs11_random_prepare, pkcs11_random_parent_post,
422             pkcs11_random_child_post);
423 }
```

*_____unchanged_portion_omitted_*

****
6905 Thu Jun 5 21:16:34 2014
new/usr/src/lib/pkcs11/libpkcs11/common/metaGeneral.c
1667 pkcs11 may deadlock when multi-threaded consumers fork
****

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */
/*
 * Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 *
 * Copyright 2014, OmniTI Computer Consulting, Inc. All rights reserved.
 */

/*
 * General-Purpose Functions
 * (as defined in PKCS#11 spec section 11.4)
 */

#include <unistd.h>
#include <errno.h>
#include <string.h>
#include "metaGlobal.h"

extern meta_session_t *meta_sessionlist_head;

struct CK_FUNCTION_LIST metaslot_functionList = {
        { 2, 20 },      /* version */
        meta_Initialize,
        meta_Finalize,
        meta_GetInfo,
        meta_GetFunctionList,
        meta_GetSlotList,
        meta_GetSlotInfo,
        meta_GetTokenInfo,
        meta_GetMechanismList,
        meta_GetMechanismInfo,
        meta_InitToken,
        meta_InitPIN,
        meta_SetPIN,
        meta_OpenSession,
        meta_CloseSession,
        meta_CloseAllSessions,
        meta_GetSessionInfo,
        meta_GetOperationState,
        meta_SetOperationState,
        meta_Login,
        meta_Logout,
```



```
        meta_CreateObject,
        meta_CopyObject,
        meta_DestroyObject,
        meta_GetObjectSize,
        meta_GetAttributeValue,
        meta_SetAttributeValue,
        meta_FindObjectsInit,
        meta_FindObjects,
        meta_FindObjectsFinal,
        meta_EncryptInit,
        meta_Encrypt,
        meta_EncryptUpdate,
        meta_EncryptFinal,
        meta_DecryptInit,
        meta_Decrypt,
        meta_DecryptUpdate,
        meta_DecryptFinal,
        meta_DigestInit,
        meta_Digest,
        meta_DigestUpdate,
        meta_DigestKey,
        meta_DigestFinal,
        meta_SignInit,
        meta_Sign,
        meta_SignUpdate,
        meta_SignFinal,
        meta_SignRecoverInit,
        meta_SignRecover,
        meta_VerifyInit,
        meta_Verify,
        meta_VerifyUpdate,
        meta_VerifyFinal,
        meta_VerifyRecoverInit,
        meta_VerifyRecover,
        meta_DigestEncryptUpdate,
        meta_DecryptDigestUpdate,
        meta_SignEncryptUpdate,
        meta_DecryptVerifyUpdate,
        meta_GenerateKey,
        meta_GenerateKeyPair,
        meta_WrapKey,
        meta_UnwrapKey,
        meta_DeriveKey,
        meta_SeedRandom,
        meta_GenerateRandom,
        meta_GetFunctionStatus,
        meta_CancelFunction,
        meta_WaitForSlotEvent
};
```

_____***unchanged_portion_omitted_***

```
/*
 * meta_Finalize
 *
 * Called by uCF only, "pReserved" argument is ignored.
 */
/*ARGSUSED*/
CK_RV
meta_Finalize(CK_VOID_PTR pReserved)
{
        CK_RV rv = CKR_OK;
        meta_object_t *delay_free_obj, *tmpo;
        meta_session_t *delay_free_ses, *tmps;

        if (pReserved != NULL)
```

```
196                 return (CKR_ARGUMENTS_BAD);

198         (void) pthread_mutex_lock(&initmutex);

200         /*
201          * There used to be calls to cleanup libcryptoutil here.  Given that
202          * libcryptoutil can be linked and invoked independently of PKCS#11,
203          * cleaning up libcryptoutil here makes no sense.  Decoupling these
204          * two also prevent deadlocks and other artificial dependencies.
205          */
199         pkcs11_close_urandom();
200         pkcs11_close_urandom_seed();

207         meta_objectManager_finalize();

209         meta_sessionManager_finalize();

211         meta_mechManager_finalize();

213         meta_slotManager_finalize();

215         /*
216          * free all entries in the delay_freed list
217          */
218         delay_free_obj = obj_delay_freed.first;
219         while (delay_free_obj != NULL) {
220                 tmpo = delay_free_obj->next;
221                 free(delay_free_obj);
222                 delay_free_obj = tmpo;
223         }
224         (void) pthread_mutex_destroy(&obj_delay_freed.obj_to_be_free_mutex);

226         delay_free_ses = ses_delay_freed.first;
227         while (delay_free_ses != NULL) {
228                 tmps = delay_free_ses->next;
229                 free(delay_free_ses);
230                 delay_free_ses = tmps;
231         }
232         (void) pthread_mutex_destroy(&ses_delay_freed.ses_to_be_free_mutex);

234         (void) pthread_mutex_unlock(&initmutex);

236         return (rv);
237 }
```

_____***unchanged_portion_omitted_***

```
**********************************************************
   14235 Thu Jun  5 21:16:34 2014
new/usr/src/lib/pkcs11/pkcs11_softtoken/common/softGeneral.c
1667 pkcs11 may deadlock when multi-threaded consumers fork
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */
/*
 * Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 *
 * Copyright 2014, OmniTI Computer Consulting, Inc. All rights reserved.
 */

#include <strings.h>
#include <errno.h>
#include <cryptoutil.h>
#include <unistd.h> /* for pid_t */
#include <pthread.h>
#include <security/cryptoki.h>
#include "softGlobal.h"
#include "softSession.h"
#include "softObject.h"
#include "softKeystore.h"
#include "softKeystoreUtil.h"

#pragma init(softtoken_init)
#pragma fini(softtoken_fini)

extern soft_session_t token_session; /* for fork handler */

static struct CK_FUNCTION_LIST functionList = {
        { 2, 20 },      /* version */
        C_Initialize,
        C_Finalize,
        C_GetInfo,
        C_GetFunctionList,
        C_GetSlotList,
        C_GetSlotInfo,
        C_GetTokenInfo,
        C_GetMechanismList,
        C_GetMechanismInfo,
        C_InitToken,
        C_InitPIN,
        C_SetPIN,
        C_OpenSession,
        C_CloseSession,
        C_CloseAllSessions,
```



```
        C_GetSessionInfo,
        C_GetOperationState,
        C_SetOperationState,
        C_Login,
        C_Logout,
        C_CreateObject,
        C_CopyObject,
        C_DestroyObject,
        C_GetObjectSize,
        C_GetAttributeValue,
        C_SetAttributeValue,
        C_FindObjectsInit,
        C_FindObjects,
        C_FindObjectsFinal,
        C_EncryptInit,
        C_Encrypt,
        C_EncryptUpdate,
        C_EncryptFinal,
        C_DecryptInit,
        C_Decrypt,
        C_DecryptUpdate,
        C_DecryptFinal,
        C_DigestInit,
        C_Digest,
        C_DigestUpdate,
        C_DigestKey,
        C_DigestFinal,
        C_SignInit,
        C_Sign,
        C_SignUpdate,
        C_SignFinal,
        C_SignRecoverInit,
        C_SignRecover,
        C_VerifyInit,
        C_Verify,
        C_VerifyUpdate,
        C_VerifyFinal,
        C_VerifyRecoverInit,
        C_VerifyRecover,
        C_DigestEncryptUpdate,
        C_DecryptDigestUpdate,
        C_SignEncryptUpdate,
        C_DecryptVerifyUpdate,
        C_GenerateKey,
        C_GenerateKeyPair,
        C_WrapKey,
        C_UnwrapKey,
        C_DeriveKey,
        C_SeedRandom,
        C_GenerateRandom,
        C_GetFunctionStatus,
        C_CancelFunction,
        C_WaitForSlotEvent
};
```

*unchanged_portion_omitted_*

```
/*
 * finalize_common() does the work for C_Finalize.  soft_giant_mutex
 * must be held before calling this function.
 */
static CK_RV
finalize_common(boolean_t force, CK_VOID_PTR pReserved) {

        CK_RV rv = CKR_OK;
        struct object *delay_free_obj, *tmpo;
        struct session *delay_free_ses, *tmps;
```

```
331         if (!softtoken_initialized) {
332                 return (CKR_CRYPTOKI_NOT_INITIALIZED);
333         }

335         /* Check to see if pReseved is NULL */
336         if (pReserved != NULL) {
337                 return (CKR_ARGUMENTS_BAD);
338         }

340         (void) pthread_mutex_lock(&soft_sessionlist_mutex);
341         /*
342          * Set all_sessions_closing flag so any access to any
343          * existing sessions will be rejected.
344          */
345         all_sessions_closing = 1;
346         (void) pthread_mutex_unlock(&soft_sessionlist_mutex);

348         /* Delete all the sessions and release the allocated resources */
349         rv = soft_delete_all_sessions(force);

351         (void) pthread_mutex_lock(&soft_sessionlist_mutex);
352         /* Reset all_sessions_closing flag. */
353         all_sessions_closing = 0;
354         (void) pthread_mutex_unlock(&soft_sessionlist_mutex);

356         softtoken_initialized = B_FALSE;
357         softtoken_pid = 0;

359         /*
360          * There used to be calls to cleanup libcryptoutil here.  Given that
361          * libcryptoutil can be linked and invoked independently of PKCS#11,
362          * cleaning up libcryptoutil here makes no sense.  Decoupling these
363          * two also prevent deadlocks and other artificial dependencies.
364          */
357         pkcs11_close_urandom();
358         pkcs11_close_urandom_seed();
359         pkcs11_close_random();

366         /* Destroy the session list lock here */
367         (void) pthread_mutex_destroy(&soft_sessionlist_mutex);

369         /*
370          * Destroy token object related stuffs
371          * 1. Clean up the token object list
372          * 2. Destroy slot mutex
373          * 3. Destroy mutex in token_session
374          */
375         soft_delete_all_in_core_token_objects(ALL_TOKEN);
376         (void) pthread_mutex_destroy(&soft_slot.slot_mutex);
377         (void) pthread_mutex_destroy(&soft_slot.keystore_mutex);
378         (void) soft_destroy_token_session();

380         /*
381          * free all entries in the delay_freed list
382          */
383         delay_free_obj = obj_delay_freed.first;
384         while (delay_free_obj != NULL) {
385                 tmpo = delay_free_obj->next;
386                 free(delay_free_obj);
387                 delay_free_obj = tmpo;
388         }

390         soft_slot.keystore_load_status = KEYSTORE_UNINITIALIZED;
391         (void) pthread_mutex_destroy(&obj_delay_freed.obj_to_be_free_mutex);
```



```
393         delay_free_ses = ses_delay_freed.first;
394         while (delay_free_ses != NULL) {
395                 tmps = delay_free_ses->next;
396                 free(delay_free_ses);
397                 delay_free_ses = tmps;
398         }
399         (void) pthread_mutex_destroy(&ses_delay_freed.ses_to_be_free_mutex);

401         return (rv);
402 }
```

_____***unchanged_portion_omitted_***