**SUMMARY**

A concurrency bug in MailNews Core

**DETAILS**

Some details can also be found at: https://bugzilla.mozilla.org/show\_bug.cgi?id=270689

This bug is due to a data race.

The creation of list->lock is not thread-safe.

Two other threads (t@2 and t@5) were also in pk11\_GetObjectFromList at that time. They were at line 2095 of pkcs11u.c.

|  |  |
| --- | --- |
| Thread1 (pkcs11u.c) | Thread2 (pkcs11u.c) |
| pk11\_GetObjectFromList(PRBool \*hasLocks, PRBool optimizeSpace,  PK11ObjectFreeList \*list, unsigned int hashSize, PRBool isSessionOb  ject)  {  PK11Object \*object;  int size = 0;  if (!optimizeSpace) {  if (list->lock == NULL) {  list->lock = PZ\_NewLock(nssILockObject);  }  PK11\_USE\_THREADS(PZ\_Lock(list->lock));  object = list->head;  if (object) {  list->head = object->next;  list->count--;  }  PK11\_USE\_THREADS(PZ\_Unlock(list->lock)); <===  if (object) {  object->next = object->prev = NULL;  \*hasLocks = PR\_TRUE;  return object;  }  } | pk11\_GetObjectFromList(PRBool \*hasLocks, PRBool optimizeSpace,  PK11ObjectFreeList \*list, unsigned int hashSize, PRBool isSessionOb  ject)  {  PK11Object \*object;  int size = 0;  if (!optimizeSpace) {  if (list->lock == NULL) {  list->lock = PZ\_NewLock(nssILockObject);  }  PK11\_USE\_THREADS(PZ\_Lock(list->lock));  object = list->head;  if (object) {  list->head = object->next;  list->count--;  }  PK11\_USE\_THREADS(PZ\_Unlock(list->lock)); <===  if (object) {  object->next = object->prev = NULL;  \*hasLocks = PR\_TRUE;  return object;  } |
| 0-lock | 0-lock |