**SUMMARY**

A concurrency bug in 1.8 branch version

**DETAILS**

Some details can also be found at: https://bugzilla.mozilla.org/show\_bug.cgi?id=342577

This bug is due to a data race.

nsZipArchive::SeekToItem can be called with the same nsZipItem on two different threads simultaneously. This creates a race if SeekToItem has never been called on that nsZipItem previously to update its offset member (and set hasDataOffset).

|  |  |
| --- | --- |
| Thread1 () | Thread2 () |
| PRInt32 nsZipArchive::SeekToItem(const nsZipItem\* aItem, PRFileDesc\* aFd)  {  ..  if (!(aItem->flags & ZIFLAG\_DATAOFFSET)) {  ..  ((nsZipItem\*)aItem)->offset += ZIPLOCAL\_SIZE +  xtoint(Local.filename\_len) +  xtoint(Local.extrafield\_len);  ((nsZipItem\*)aItem)->flags |= ZIFLAG\_DATAOFFSET;  .. | PRInt32 nsZipArchive::SeekToItem(const nsZipItem\* aItem, PRFileDesc\* aFd)  {  ..  if (!(aItem->flags & ZIFLAG\_DATAOFFSET)) {  ..  ((nsZipItem\*)aItem)->offset += ZIPLOCAL\_SIZE +  xtoint(Local.filename\_len) +  xtoint(Local.extrafield\_len);  ((nsZipItem\*)aItem)->flags |= ZIFLAG\_DATAOFFSET;  .. |
| 0-lock | 0-lock |

thr 0x2254000 (A): PR\_OpenFile(..., 1, \00) = 23

thr 0x2254000 (A): nsZipArchive::SeekToItem(0x219f558, [23]) enter

thr 0x2254000 (A): \_MD\_lseek: lseek(23, 492557, 0) = 0

thr 0x2262000 (B): PR\_OpenFile(..., 1, \00) = 24

thr 0x2262000 (B): nsZipArchive::SeekToItem(0x219f558, [24]) enter

thr 0x2262000 (B): \_MD\_lseek: lseek(24, 492557, 0) = 0

thr 0x2254000 (A): \_MD\_lseek: lseek(23, 492620, 0) = 0

thr 0x2262000 (B): \_MD\_lseek: lseek(24, 492683, 0) = 0

thr 0x2254000 (A): nsZipArchive::SeekToItem: PR\_Seek([23], 492683, 0) = 492620

(The letters A and B were added to make it easier to identify the two threads. In this particular case, ... is omitted, but the file is "/path/to/MinefieldDebug.app/Contents/MacOS/chrome/browser.jar" on both threads, and aItem->name is "content/browser/utilityOverlay.js". This behavior also occurs with other members and other jar files.)