

One of the warnings CodeSonar® issued in analyzing the open-source gnuchess program was the Leak warning shown in the screenshot fragment below.

```
c:\gnuchess-5.07\gnuchess-5.07\src\book.c Options ▾
292 int BookBuilderOpen(void)
293 {
294     FILE *rfp, *wfp;
295     int res;
296
297     if ((rfp = fopen(BOOKRUN, "rb")) != NULL) {
298         printf("Opened existing book!\n");
299         if (!check_magic(rfp)) {
300             fprintf(stderr,
301                 "File %s does not conform to the current format.\n"
302                 "Consider rebuilding your book.\n",
303                 BOOKRUN);
304             return BOOK_EFORMAT;
305         }
306         /*
307          * We have to read the size header, but in book building we
308          * use the maximum-sized hash table, so we discard the value.
309          */
310         digest_bits = MAX_DIGEST_BITS;
311         read_size(rfp);
312         res = read_book(rfp);
313         fclose(rfp);
314         if (res != BOOK_SUCCESS) {
315             [ Lines 315 to 340 omitted. ]
341             BOOKRUN, strerror(errno));
342             return BOOK_EIO;
343         }
344         digest_bits = MAX_DIGEST_BITS;
345         /* We use read_book() here only to allocate memory */
346         if (read_book(wfp) == BOOK_ENOMEM) {
347             return BOOK_ENOMEM;
348         }
349     }
350     return BOOK_SUCCESS;
351 }
Leak
There are no remaining references to the resource fopen("book.dat", "rb") from book.c:297.
• The resource was allocated at book.c:297.
• The last reference was lost at book.c:351.
• The resource was not freed.
The issue can occur if the highlighted code executes.
See related events 1, 2, 4, 5, 6, 8, and 10.
Show: All events | Only primary events
```

This fragment warns of a potential file pointer resource leak. The file pointer opened on line 297 may be leaked when the function returns on line 304. The problem is that the filepointer is stored in a local variable, rfp. When the function returns on line 304, rfp goes out of scope and the handle to the open file pointer is permanently lost.

The path to the point where the leak occurs is shown in red. Users can inspect the check_magic() call on line 299 by clicking on the plus sign, and see that the path taken through write_magic() does not clean up rfp.