libextractor

GNU libextractor

Log | Files | Refs | Submodules | README | LICENSE

---

commit 04004eb19033e093938138b09befdf31e71e8522
parent 3ee5a54450fff66fb28a77e16636f0659686b2ec
Author: Christian Grothoff <christian@grothoff.org>
Date: Mon, 29 Jun 2026 20:00:09 +0200

fix large stack allocation

Diffstat:

M	ChangeLog	6	++++++
M	src/plugins/dvi_extractor.c	3	+++
M	src/plugins/ole2_extractor.c	89	++-----------------------------

3 files changed, 56 insertions(+), 42 deletions(-)

```
diff --git a/ChangeLog b/ChangeLog
@@ -1,3 +1,9 @@
+Mon Jun 29 07:58:39 PM CEST 2026
+    Fix potential 4 MB on-stack memory allocation that could
+    result in a stack-based buffer overflow in the OLE2 extractor.
+    Thanks to Haitam Lazaar for reporting.
+    Releasing GNU libextractor 1.14. -CG
+
 Mon May 25 12:43:20 PM CEST 2026
        Revive REAL plugin (fixes #2518).
        Revive VLC plugin (fixes #2075).
diff --git a/src/plugins/dvi_extractor.c b/src/plugins/dvi_extractor.c
@@ -158,6 +158,9 @@ getShortAt (const void *data)
  * @param ec extraction context provided to the plugin
  */
 void
+EXTRACTOR_dvi_extract_method (struct EXTRACTOR_ExtractContext *ec);
+
+void
 EXTRACTOR_dvi_extract_method (struct EXTRACTOR_ExtractContext *ec)
 {
   unsigned int klen;
diff --git a/src/plugins/ole2_extractor.c b/src/plugins/ole2_extractor.c
@@ -341,50 +341,51 @@ process_star_office (GsfInput *src,
                      void *proc_cls)
 {
   off_t size = gsf_input_size (src);
+  char buf[1024];

   if ( (size < 0x374) ||
-       (size > 4 * 1024 * 1024) ) /* == 0x375?? */
+       (size > 16 * 1024 * 1024) )
     return 0;
-  {
-    char buf[size];
-
-    gsf_input_read (src, size, (unsigned char*) buf);
-    if ( (buf[0] != 0x0F) ||
-         (buf[1] != 0x0) ||
-         (0 != strncmp (&buf[2],
-                        "SfxDocumentInfo",
-                        strlen ("SfxDocumentInfo"))) ||
-         (buf[0x11] != 0x0B) ||
-         (buf[0x13] != 0x00) || /* pw protected! */
-         (buf[0x12] != 0x00) )
-      return 0;
-    buf[0xd3] = '\0';
-    if ( (buf[0x94] + buf[0x93] > 0) &&
-         (0 != add_metadata (proc, proc_cls,
-                             &buf[0x95],
-                             EXTRACTOR_METATYPE_TITLE)) )
-      return 1;
-    buf[0x114] = '\0';
-    if ( (buf[0xd5] + buf[0xd4] > 0) &&
-         (0 != add_metadata (proc, proc_cls,
```

```
-                                &buf[0xd6],
-                                EXTRACTOR_METATYPE_SUBJECT)) )
-      return 1;
-    buf[0x215] = '\0';
-    if ( (buf[0x115] + buf[0x116] > 0) &&
-         (0 != add_metadata (proc, proc_cls,
-                             &buf[0x117],
-                             EXTRACTOR_METATYPE_COMMENT)) )
-      return 1;
-    buf[0x296] = '\0';
-    if ( (buf[0x216] + buf[0x217] > 0) &&
-         (0 != add_metadata (proc, proc_cls,
-                             &buf[0x218],
-                             EXTRACTOR_METATYPE_KEYWORDS)) )
-      return 1;
-    /* fixme: do timestamps,
-       mime-type, user-defined info's */
-  }
+  if (size > sizeof (buf))
+    size = sizeof (buf);
+  gsf_input_read (src,
+                  size,
+                  (unsigned char*) buf);
+  if ( (buf[0] != 0x0F) ||
+       (buf[1] != 0x0) ||
+       (0 != strncmp (&buf[2],
+                      "SfxDocumentInfo",
+                      strlen ("SfxDocumentInfo"))) ||
+       (buf[0x11] != 0x0B) ||
+       (buf[0x13] != 0x00) ||   /* pw protected! */
+       (buf[0x12] != 0x00) )
+    return 0;
+  buf[0xd3] = '\0';
+  if ( (buf[0x94] + buf[0x93] > 0) &&
+       (0 != add_metadata (proc, proc_cls,
+                           &buf[0x95],
+                           EXTRACTOR_METATYPE_TITLE)) )
+    return 1;
+  buf[0x114] = '\0';
+  if ( (buf[0xd5] + buf[0xd4] > 0) &&
+       (0 != add_metadata (proc, proc_cls,
+                           &buf[0xd6],
+                           EXTRACTOR_METATYPE_SUBJECT)) )
+    return 1;
+  buf[0x215] = '\0';
+  if ( (buf[0x115] + buf[0x116] > 0) &&
+       (0 != add_metadata (proc, proc_cls,
+                           &buf[0x117],
+                           EXTRACTOR_METATYPE_COMMENT)) )
+    return 1;
+  buf[0x296] = '\0';
+  if ( (buf[0x216] + buf[0x217] > 0) &&
+       (0 != add_metadata (proc, proc_cls,
+                           &buf[0x218],
+                           EXTRACTOR_METATYPE_KEYWORDS)) )
+    return 1;
+  /* fixme: do timestamps,
+     mime-type, user-defined info's */
   return 0;
 }

@@ -649,7 +650,8 @@ history_extract (GsfInput *stream,
                                                                 LE_TYPE_INPUT))
 #define LE_INPUT_GET_CLASS(obj)        (G_TYPE_INSTANCE_GET_CLASS ((obj), \
                                                                    LE_TYPE_INPUT, \
-                                                                   LeInputClass))
+                                                                   LeInputClass) \
+                                                                  )

 /**
  * Internal state of an "LeInput" object.
@@ -908,6 +910,9 @@ le_input_new (struct EXTRACTOR_ExtractContext *ec)
  * @param ec extraction context provided to the plugin
  */
 void
+EXTRACTOR_ole2_extract_method (struct EXTRACTOR_ExtractContext *ec);
+
+void
 EXTRACTOR_ole2_extract_method (struct EXTRACTOR_ExtractContext *ec)
```

```
  {
    GsfInput *input;
```