jmacd / **xdelta** Public

Code | Issues 113 | Pull requests 5 | Actions | Projects | Security | Insights

Add appheader tests; fix buffer overflow in main_get_appheader

Browse files

release3_1_apl
v3.1.0 ··· v3.0.9

jmacd committed on Oct 12, 2014 — 1 parent 7b6ff92 — commit ef93ff74203e030073b898c05e8b4860b5d09ef2

Showing **2 changed files** with **108 additions** and **28 deletions**.

Unified | Split

5 xdelta3/xdelta3-main.h

```diff
 
     if (appheadsz > 0)
       {
+        const int kMaxArgs = 4;
         char *start = (char*)apphead;
         char *slash;
         int   place = 0;
-        char *parsed[4];
+        char *parsed[kMaxArgs];
 
         memset (parsed, 0, sizeof (parsed));
 
-        while ((slash = strchr (start, '/')) != NULL)
+        while ((slash = strchr (start, '/')) != NULL && place < (kMaxArgs-1))
     {
       *slash = 0;
```

```
        parsed[place++] = start;
```

131 xdelta3/xdelta3-test.h

```
@@ -1,5 +1,5 @@
  /* xdelta 3 - delta compression tools and library Copyright (C) 2001,
-  * 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012. 
+  * 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012.
   * Joshua P. MacDonald
   *
   *  This program is free software; you can redistribute it and/or modify
      /* only MSBs of the array mt[].                        */
      /* 2002/01/09 modified by Makoto Matsumoto             */
      mt->mt_buffer_[i] =
-    (1812433253UL * (mt->mt_buffer_[i-1] ^ 
+    (1812433253UL * (mt->mt_buffer_[i-1] ^
                 (mt->mt_buffer_[i-1] >> 30)) + i);
    }
  }
      int kk;

      for (kk = 0; kk < MT_LEN - MT_IA; kk++) {
-       y = (mt->mt_buffer_[kk] & UPPER_MASK) | 
+       y = (mt->mt_buffer_[kk] & UPPER_MASK) |
    (mt->mt_buffer_[kk + 1] & LOWER_MASK);
-       mt->mt_buffer_[kk] = mt->mt_buffer_[kk + MT_IA] ^ 
+       mt->mt_buffer_[kk] = mt->mt_buffer_[kk + MT_IA] ^
    (y >> 1) ^ mag01[y & 0x1UL];
      }
      for (;kk < MT_LEN - 1; kk++) {
-       y = (mt->mt_buffer_[kk] & UPPER_MASK) | 
+       y = (mt->mt_buffer_[kk] & UPPER_MASK) |
    (mt->mt_buffer_[kk + 1] & LOWER_MASK);
-       mt->mt_buffer_[kk] = mt->mt_buffer_[kk + (MT_IA - MT_LEN)] ^ 
+      mt->mt_buffer_[kk] = mt->mt_buffer_[kk + (MT_IA - MT_LEN)] ^
     (y >> 1) ^ mag01[y & 0x1UL];
      }
-     y = (mt->mt_buffer_[MT_LEN - 1] & UPPER_MASK) | 
+     y = (mt->mt_buffer_[MT_LEN - 1] & UPPER_MASK) |
       (mt->mt_buffer_[0] & LOWER_MASK);
-     mt->mt_buffer_[MT_LEN - 1] = mt->mt_buffer_[MT_IA - 1] ^ 
+     mt->mt_buffer_[MT_LEN - 1] = mt->mt_buffer_[MT_IA - 1] ^
       (y >> 1) ^ mag01[y & 0x1UL];
      mt->mt_index_ = 0;
    }
    {
      stream->msg = "abnormal command termination";
    }
-      return XD3_INTERNAL;
+      return ret;
      }
    return 0;
  }
  static int
  test_make_inputs (xd3_stream *stream, xoff_t *ss_out, xoff_t *ts_out)
  {
-   usize_t ts = (mt_random (&static_mtrand) % TEST_FILE_MEAN) + TEST_FILE_MEAN / 2;
-   usize_t ss = (mt_random (&static_mtrand) % TEST_FILE_MEAN) + TEST_FILE_MEAN / 2;
+   usize_t ts = (mt_random (&static_mtrand) % TEST_FILE_MEAN) +
+     TEST_FILE_MEAN / 2;
+   usize_t ss = (mt_random (&static_mtrand) % TEST_FILE_MEAN) +
+     TEST_FILE_MEAN / 2;
    uint8_t *buf = (uint8_t*) malloc (ts + ss), *sbuf = buf, *tbuf = buf + ss;
    usize_t sadd = 0, sadd_max = (usize_t)(ss * TEST_ADD_RATIO);
    FILE  *tf = NULL, *sf = NULL;
    {
      if (obuf[i] != rbuf[i])
        {
```

```diff
-         XPR(NT "byte %u (read %u @ %"Q"u) %d != %d\n", 
+         XPR(NT "byte %u (read %u @ %"Q"u) %d != %d\n",
           (int)i, (int)oc, offset, obuf[i], rbuf[i]);
          diffs++;
          return XD3_INTERNAL;

      fclose (orig);
      fclose (recons);
-     if (diffs != 0) 
+     if (diffs != 0)
        {
    return XD3_INTERNAL;
        }
      return 0;
  }

  static int
- test_save_copy (const char *origname)
+ test_copy_to (const char *from, const char *to)
  {
    char buf[TESTBUFSIZE];
    int ret;

-   snprintf_func (buf, TESTBUFSIZE, "cp -f %s %s", origname, TEST_COPY_FILE);
+   snprintf_func (buf, TESTBUFSIZE, "cp -f %s %s", from, to);

    if ((ret = system (buf)) != 0)
      {
    return 0;
  }

+ static int
+ test_save_copy (const char *origname)
+ {
```

```
+   return test_copy_to(origname, TEST_COPY_FILE);
+ }
+
  static int
  test_file_size (const char* file, xoff_t *size)
  {
    inp = buf->base;
    max = buf->base + buf->next - trunto;

-   if ((ret = xd3_read_uint32_t (stream, & inp, max, & rval)) != 
+   if ((ret = xd3_read_uint32_t (stream, & inp, max, & rval)) !=
        XD3_INVALID_INPUT ||
        !MSG_IS (msg))
      {
    if ((buf = (uint8_t*) malloc (TWO_MEGS_AND_DELTA)) == NULL) { return ENOMEM; }

    memset (buf, 0, TWO_MEGS_AND_DELTA);
-   for (i = 0; i < (2 << 20); i += 256) 
+   for (i = 0; i < (2 << 20); i += 256)
      {
        int j;
        int off = mt_random(& static_mtrand) % 10;
-       for (j = 0; j < 256; j++) 
+       for (j = 0; j < 256; j++)
     {
       buf[i + j] = j + off;
     }
      }

    /* Test transfer of exactly 32bits worth of data. */
-   if ((ret = test_streaming (stream, 
-                 buf, 
-                 buf + (1 << 20), 
-                 buf + (2 << 20), 
-                 1 << 12)))
+   if ((ret = test_streaming (stream,
+                 buf,
+                 buf + (1 << 20),
+                 buf + (2 << 20),
+                 1 << 12)))
      {
        goto fail;
      }
      }

    /* First encode */
-   snprintf_func (ecmd, TESTBUFSIZE, "%s %s -f %s %s %s %s %s %s %s",
+   snprintf_func (ecmd, TESTBUFSIZE, "%s %s -f %s %s %s %s %s %s %s",
        program_name, test_softcfg_str,
        has_adler32 ? "" : "-n ",
        has_apphead ? "-A=encode_apphead " : "-A= ",
    snprintf_func (recmd, TESTBUFSIZE,
        "%s recode %s -f %s %s %s %s %s", program_name, test_softcfg_str,
        recoded_adler32 ? "" : "-n ",
-       !change_apphead ? "" :
+       !change_apphead ? "" :
            (recoded_apphead ? "-A=recode_apphead " : "-A= "),
        recoded_secondary ? "-S djw " : "-S none ",
        TEST_DELTA_FILE,
    return 0;
  }

+ /* This tests that the default appheader works */
+ static int
+ test_appheader (xd3_stream *stream, int ignore)
+ {
+   int i;
+   int ret;
+   char buf[TESTBUFSIZE];
+   char bogus[TESTBUFSIZE];
+   xoff_t ssize, tsize;
+   test_setup ();
+
+   if ((ret = test_make_inputs (stream, &ssize, &tsize))) { return ret; }
+
+   snprintf_func (buf, TESTBUFSIZE, "%s -q -f -e -s %s %s %s", program_name,
+      TEST_SOURCE_FILE, TEST_TARGET_FILE, TEST_DELTA_FILE);
+   if ((ret = do_cmd (stream, buf))) { return ret; }
+
+   if ((ret = test_copy_to (program_name, TEST_RECON2_FILE))) { return ret; }
+
+   snprintf_func (buf, TESTBUFSIZE, "chmod 0700 %s", TEST_RECON2_FILE);
+   if ((ret = do_cmd (stream, buf))) { return ret; }
+
+   if ((ret = test_save_copy (TEST_TARGET_FILE))) { return ret; }
+   if ((ret = test_copy_to (TEST_SOURCE_FILE, TEST_TARGET_FILE))) { return ret; }
+
+   if ((ret = test_compare_files (TEST_TARGET_FILE, TEST_COPY_FILE)) == 0)
+    {
+     return XD3_INVALID;  // I.e., files are different!
+    }
+
+   // Test that the target file is restored.
+   snprintf_func (buf, TESTBUFSIZE, "(cd /tmp && %s -q -f -d %s)",
+      TEST_RECON2_FILE,
+      TEST_DELTA_FILE);
+   if ((ret = do_cmd (stream, buf))) { return ret; }
+
+   if ((ret = test_compare_files (TEST_TARGET_FILE, TEST_COPY_FILE)) != 0)
+    {
+     return ret;
+    }
```

```
+
+   // Test a malicious string w/ entries > 4 in the appheader by having
+   // the encoder write it:
+   for (i = 0; i < TESTBUFSIZE / 4; ++i)
+     {
+       bogus[2*i] = 'G';
+       bogus[2*i+1] = '/';
+     }
+   bogus[TESTBUFSIZE/2-1] = 0;
+
+   snprintf_func (buf, TESTBUFSIZE,
+        "%s -q -f -A=%s -e -s %s %s %s", program_name, bogus,
+        TEST_SOURCE_FILE, TEST_TARGET_FILE, TEST_DELTA_FILE);
+   if ((ret = do_cmd (stream, buf))) { return ret; }
+   // Then read it:
+   snprintf_func (buf, TESTBUFSIZE, "(cd /tmp && %s -q -f -d %s)",
+        TEST_RECON2_FILE,
+        TEST_DELTA_FILE);
+   if ((ret = do_cmd (stream, buf)) == 0)
+     {
+       return XD3_INVALID;  // Impossible
+     }
+   if (!WIFEXITED(ret))
+     {
+       return XD3_INVALID;  // Must have crashed!
+     }
+
+   return 0;
+ }
+
  /***********************************************************************
   Source identical optimization
   ***********************************************************************/
        default: CHECK(0);
```

```
         }

-      snprintf_func (rptr, rbuf+TESTBUFSIZE-rptr, "%d/%d", 
+      snprintf_func (rptr, rbuf+TESTBUFSIZE-rptr, "%d/%d",
              inst->pos, inst->size);
       rptr += strlen (rptr);

    DO_TEST (force_behavior, 0, 0);
    DO_TEST (stdout_behavior, 0, 0);
    DO_TEST (no_output, 0, 0);
+   DO_TEST (appheader, 0, 0);
    DO_TEST (command_line_arguments, 0, 0);

  #if EXTERNAL_COMPRESSION
```

0 comments on commit `ef93ff7`