python / **cpython** Public

Code | Issues 5k+ | Pull requests 2.2k | Actions | Projects | Security and q

Commit 7ad3093

3 people authored on Mar 17 · ✕ 81 / 87 · Partially verified

```
[3.14] gh-141707: Skip TarInfo DIRTYPE normalization during GNU long name
handling (GH-145819)

(cherry picked from commit 42d754e)

Co-authored-by: Seth Michael Larson <seth@python.org>
Co-authored-by: Eashwar Ranganathan <eashwar@eashwar.com>
```

3.14 (#145819) · v3.14.4

1 parent 4601007 commit 7ad3093

4 files changed +47 -4

Top

Filter files…

- Lib
  - tarfile.py
  - test
    - test_tarfile.py
- Misc
  - ACKS
  - NEWS.d/next/Library
    - 2025-11-18-06-35-53.gh-issue-141707.DBmQIy.rst

Search within code

Lib/tarfile.py

```
@@ -1278,6 +1278,20 @@ def _create_pax_generic_header(cls, pax_headers,
type, encoding):
1278 1278         @classmethod
1279 1279         def frombuf(cls, buf, encoding, errors):
1280 1280             """Construct a TarInfo object from a 512 byte bytes object.
     1281 +
     1282 +         To support the old v7 tar format AREGTYPE headers are
     1283 +         transformed to DIRTYPE headers if their name ends in '/'.
     1284 +         """
     1285 +         return cls._frombuf(buf, encoding, errors)
     1286 +
     1287 +     @classmethod
     1288 +     def _frombuf(cls, buf, encoding, errors, *, dircheck=True):
     1289 +         """Construct a TarInfo object from a 512 byte bytes object.
     1290 +
     1291 +         If ``dircheck`` is set to ``True`` then ``AREGTYPE`` headers will
     1292 +         be normalized to ``DIRTYPE`` if the name ends in a trailing slash.
     1293 +         ``dircheck`` must be set to ``False`` if this function is called
     1294 +         on a follow-up header such as ``GNUTYPE_LONGNAME``.
1281 1295             """
1282 1296             if len(buf) == 0:
1283 1297                 raise EmptyHeaderError("empty header")
@@ -1308,7 +1322,7 @@ def frombuf(cls, buf, encoding, errors):
1308 1322
1309 1323             # Old V7 tar format represents a directory as a regular
1310 1324             # file with a trailing slash.
1311      -         if obj.type == AREGTYPE and obj.name.endswith("/"):
     1325 +         if dircheck and obj.type == AREGTYPE and obj.name.endswith("/"):
1312 1326                 obj.type = DIRTYPE
1313 1327
1314 1328             # The old GNU sparse format occupies some of the unused
@@ -1343,8 +1357,15 @@ def fromtarfile(cls, tarfile):
1343 1357             """Return the next TarInfo object from TarFile object
1344 1358                tarfile.
1345 1359             """
     1360 +         return cls._fromtarfile(tarfile)
     1361 +
     1362 +     @classmethod
```

```
                1363 +     def _fromtarfile(cls, tarfile, *, dircheck=True):
                1364 +         """
                1365 +         See dircheck documentation in _frombuf().
                1366 +         """
1346            1367           buf = tarfile.fileobj.read(BLOCKSIZE)
1347             -         obj = cls.frombuf(buf, tarfile.encoding, tarfile.errors)
                1368 +         obj = cls._frombuf(buf, tarfile.encoding, tarfile.errors,
                                dircheck=dircheck)
1348            1369           obj.offset = tarfile.fileobj.tell() - BLOCKSIZE
1349            1370           return obj._proc_member(tarfile)
1350            1371
                              @@ -1402,7 +1423,7 @@ def _proc_gnulong(self, tarfile):
1402            1423
1403            1424           # Fetch the next header and process it.
1404            1425           try:
1405             -             next = self.fromtarfile(tarfile)
                1426 +             next = self._fromtarfile(tarfile, dircheck=False)
1406            1427           except HeaderError as e:
1407            1428               raise SubsequentHeaderError(str(e)) from None
1408            1429
                              @@ -1537,7 +1558,7 @@ def _proc_pax(self, tarfile):
1537            1558
1538            1559           # Fetch the next header.
1539            1560           try:
1540             -             next = self.fromtarfile(tarfile)
                1561 +             next = self._fromtarfile(tarfile, dircheck=False)
1541            1562           except HeaderError as e:
1542            1563               raise SubsequentHeaderError(str(e)) from None
1543            1564
```

Lib/test/test_tarfile.py

```
                              @@ -1234,6 +1234,25 @@ def test_longname_directory(self):
1234            1234                   self.assertIsNotNone(tar.getmember(longdir))
1235            1235
                                self.assertIsNotNone(tar.getmember(longdir.removesuffix('/')))
1236            1236
                1237 +     def test_longname_file_not_directory(self):
```

```
+         # Test reading a longname file and ensure it is not handled as a directory
+         # Issue #141707
+         buf = io.BytesIO()
+         with tarfile.open(mode='w', fileobj=buf, format=self.format) as tar:
+             ti = tarfile.TarInfo()
+             ti.type = tarfile.AREGTYPE
+             ti.name = ('a' * 99) + '/' + ('b' * 3)
+             tar.addfile(ti)
+
+             expected = {t.name: t.type for t in tar.getmembers()}
+
+         buf.seek(0)
+         with tarfile.open(mode='r', fileobj=buf) as tar:
+             actual = {t.name: t.type for t in tar.getmembers()}
+
+         self.assertEqual(expected, actual)
+
+
1237 1256  class GNUReadTest(LongnameTest, ReadTest, unittest.TestCase):
1238 1257
1239 1258      subdir = "gnu"
```

Misc/ACKS

```
@@ -1535,6 +1535,7 @@ Ashwin Ramaswami
1535 1535  Jeff Ramnani
1536 1536  Grant Ramsay
1537 1537  Bayard Randel
     1538 + Eashwar Ranganathan
1538 1539  Varpu Rantala
1539 1540  Brodie Rao
1540 1541  Rémi Rampin
```

...5-11-18-06-35-53.gh-issue-141707.DBmQIy.rst

```
@@ -0,0 +1,2 @@
1 + Don't change :class:`tarfile.TarInfo` type from ``AREGTYPE`` to ``DIRTYPE`` when parsing
```

```
2  + GNU long name or link headers.
```

Comments 0