python / **cpython** Public

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

Commit `ae99fe3`

3 people authored on Mar 17 · ✕ 86 / 91 · Partially verified

```
[3.13] gh-141707: Skip TarInfo DIRTYPE normalization during GNU long name
handling (GH-145818)

(cherry picked from commit 42d754e)

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

3.13 (AcreetionOS-Linux/python#2, #145818) · v3.13.13

1 parent 2705364 commit ae99fe3

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

```
@@ -1267,6 +1267,20 @@ def _create_pax_generic_header(cls, pax_headers,
type, encoding):
1267 1267         @classmethod
1268 1268         def frombuf(cls, buf, encoding, errors):
1269 1269             """Construct a TarInfo object from a 512 byte bytes object.
     1270 +
     1271 +         To support the old v7 tar format AREGTYPE headers are
     1272 +         transformed to DIRTYPE headers if their name ends in '/'.
     1273 +         """
     1274 +         return cls._frombuf(buf, encoding, errors)
     1275 +
     1276 +     @classmethod
     1277 +     def _frombuf(cls, buf, encoding, errors, *, dircheck=True):
     1278 +         """Construct a TarInfo object from a 512 byte bytes object.
     1279 +
     1280 +         If ``dircheck`` is set to ``True`` then ``AREGTYPE`` headers will
     1281 +         be normalized to ``DIRTYPE`` if the name ends in a trailing slash.
     1282 +         ``dircheck`` must be set to ``False`` if this function is called
     1283 +         on a follow-up header such as ``GNUTYPE_LONGNAME``.
1270 1284             """
1271 1285             if len(buf) == 0:
1272 1286                 raise EmptyHeaderError("empty header")
@@ -1297,7 +1311,7 @@ def frombuf(cls, buf, encoding, errors):
1297 1311
1298 1312             # Old V7 tar format represents a directory as a regular
1299 1313             # file with a trailing slash.
1300      -          if obj.type == AREGTYPE and obj.name.endswith("/"):
     1314 +          if dircheck and obj.type == AREGTYPE and obj.name.endswith("/"):
1301 1315                 obj.type = DIRTYPE
1302 1316
1303 1317             # The old GNU sparse format occupies some of the unused
@@ -1332,8 +1346,15 @@ def fromtarfile(cls, tarfile):
1332 1346             """Return the next TarInfo object from TarFile object
1333 1347                tarfile.
1334 1348             """
     1349 +          return cls._fromtarfile(tarfile)
     1350 +
     1351 +     @classmethod
```

```
                1352 +     def _fromtarfile(cls, tarfile, *, dircheck=True):
                1353 +         """
                1354 +         See dircheck documentation in _frombuf().
                1355 +         """
1335            1356           buf = tarfile.fileobj.read(BLOCKSIZE)
1336            -              obj = cls.frombuf(buf, tarfile.encoding, tarfile.errors)
                1357 +         obj = cls._frombuf(buf, tarfile.encoding, tarfile.errors,
                               dircheck=dircheck)
1337            1358           obj.offset = tarfile.fileobj.tell() - BLOCKSIZE
1338            1359           return obj._proc_member(tarfile)
1339            1360
                               @@ -1391,7 +1412,7 @@ def _proc_gnulong(self, tarfile):
1391            1412
1392            1413           # Fetch the next header and process it.
1393            1414           try:
1394            -              next = self.fromtarfile(tarfile)
                1415 +             next = self._fromtarfile(tarfile, dircheck=False)
1395            1416           except HeaderError as e:
1396            1417               raise SubsequentHeaderError(str(e)) from None
1397            1418
                               @@ -1526,7 +1547,7 @@ def _proc_pax(self, tarfile):
1526            1547
1527            1548           # Fetch the next header.
1528            1549           try:
1529            -              next = self.fromtarfile(tarfile)
                1550 +             next = self._fromtarfile(tarfile, dircheck=False)
1530            1551           except HeaderError as e:
1531            1552               raise SubsequentHeaderError(str(e)) from None
1532            1553
```

Lib/test/test_tarfile.py

```
                               @@ -1216,6 +1216,25 @@ def test_longname_directory(self):
1216            1216                   self.assertIsNotNone(tar.getmember(longdir))
1217            1217
                               self.assertIsNotNone(tar.getmember(longdir.removesuffix('/')))
1218            1218
                1219 +     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)
+
+
1219 1238  class GNUReadTest(LongnameTest, ReadTest, unittest.TestCase):
1220 1239
1221 1240      subdir = "gnu"
```

Misc/ACKS

```
@@ -1512,6 +1512,7 @@ Ashwin Ramaswami
1512 1512  Jeff Ramnani
1513 1513  Grant Ramsay
1514 1514  Bayard Randel
     1515 + Eashwar Ranganathan
1515 1516  Varpu Rantala
1516 1517  Brodie Rao
1517 1518  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