

python / cpython Public

<> Code Issues 5k+ Pull requests 2.1k Actions Projects Security and q

# Commit 42d754e



sethmlarson and e-nodem authored last month · ✓ 69 / 72 · Verified



gh-141707: Skip TarInfo DIRTYE normalization during GNU long name handling

Co-authored-by: Eashwar Ranganathan <eashwar@eashwar.com>

main (#143934) · v3.15.0a8

1 parent ce1abaf commit 42d754e

4 files changed +47 -4 lines changed

↑ 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.DBmQly.rst

4 files changed +47 -4 lines changed

Search within code ⚙️

Lib/tarfile.py

```

@@ -1276,6 +1276,20 @@ def _create_pax_generic_header(cls, pax_headers,
type, encoding):
1276 1276     @classmethod
1277 1277     def frombuf(cls, buf, encoding, errors):

```

```

1278 1278         """Construct a TarInfo object from a 512 byte bytes object.
1279 +
1280 +         To support the old v7 tar format AREGTYPE headers are
1281 +         transformed to DIRTYE headers if their name ends in '/'.
1282 +         """
1283 +         return cls._frombuf(buf, encoding, errors)
1284 +
1285 +         @classmethod
1286 +         def _frombuf(cls, buf, encoding, errors, *, dircheck=True):
1287 +             """Construct a TarInfo object from a 512 byte bytes object.
1288 +
1289 +             If ``dircheck`` is set to ``True`` then ``AREGTYPE`` headers will
1290 +             be normalized to ``DIRTYE`` if the name ends in a trailing slash.
1291 +             ``dircheck`` must be set to ``False`` if this function is called
1292 +             on a follow-up header such as ``GNUTYPE_LONGNAME``.
1293 +
1294 +             """
1295 +             if len(buf) == 0:
1296 +                 raise EmptyHeaderError("empty header")
1297 +
1298 +             @@ -1306,7 +1320,7 @@ def frombuf(cls, buf, encoding, errors):
1299 +
1300 +             # Old V7 tar format represents a directory as a regular
1301 +             # file with a trailing slash.
1302 +
1303 +             - if obj.type == AREGTYPE and obj.name.endswith("/"):
1304 +             + if dircheck and obj.type == AREGTYPE and obj.name.endswith("/"):
1305 +
1306 +                 obj.type = DIRTYE
1307 +
1308 +             # The old GNU sparse format occupies some of the unused
1309 +
1310 +             @@ -1341,8 +1355,15 @@ def fromtarfile(cls, tarfile):
1311 +
1312 +             """Return the next TarInfo object from TarFile object
1313 +             tarfile.
1314 +             """
1315 +
1316 +             return cls._fromtarfile(tarfile)
1317 +
1318 +             @classmethod
1319 +             def _fromtarfile(cls, tarfile, *, dircheck=True):
1320 +                 """
1321 +                 See dircheck documentation in _frombuf().
1322 +                 """

```

```

1344 1365         buf = tarfile.fileobj.read(BLOCKSIZE)
1345 -         obj = cls.frombuf(buf, tarfile.encoding, tarfile.errors)
1366 +         obj = cls._frombuf(buf, tarfile.encoding, tarfile.errors,
dircheck=dircheck)
1346 1367         obj.offset = tarfile.fileobj.tell() - BLOCKSIZE
1347 1368         return obj._proc_member(tarfile)
1348 1369
@@ -1400,7 +1421,7 @@ def _proc_gnulong(self, tarfile):
1400 1421
1401 1422         # Fetch the next header and process it.
1402 1423         try:
1403 -             next = self.fromtarfile(tarfile)
1424 +             next = self._fromtarfile(tarfile, dircheck=False)
1404 1425         except HeaderError as e:
1405 1426             raise SubsequentHeaderError(str(e)) from None
1406 1427
@@ -1535,7 +1556,7 @@ def _proc_pax(self, tarfile):
1535 1556
1536 1557         # Fetch the next header.
1537 1558         try:
1538 -             next = self.fromtarfile(tarfile)
1559 +             next = self._fromtarfile(tarfile, dircheck=False)
1539 1560         except HeaderError as e:
1540 1561             raise SubsequentHeaderError(str(e)) from None
1541 1562

```

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):
1238 +         # Test reading a longname file and ensure it is not handled as a
directory
1239 +         # Issue #141707
1240 +         buf = io.BytesIO()

```

```

1241 +         with tarfile.open(mode='w', fileobj=buf, format=self.format) as tar:
1242 +             ti = tarfile.TarInfo()
1243 +             ti.type = tarfile.AREGTYPE
1244 +             ti.name = ('a' * 99) + '/' + ('b' * 3)
1245 +             tar.addfile(ti)
1246 +
1247 +             expected = {t.name: t.type for t in tar.getmembers()}
1248 +
1249 +             buf.seek(0)
1250 +             with tarfile.open(mode='r', fileobj=buf) as tar:
1251 +                 actual = {t.name: t.type for t in tar.getmembers()}
1252 +
1253 +             self.assertEqual(expected, actual)
1254 +
1255 +

```

```

1237 1256     class GNUReadTest(LongnameTest, ReadTest, unittest.TestCase):
1238 1257
1239 1258         subdir = "gnu"

```



▼ Misc/ACKS



@@ -1557,6 +1557,7 @@ Ashwin Ramaswami

```

1557 1557     Jeff Ramnani
1558 1558     Grant Ramsay
1559 1559     Bayard Randel
1560 + Eashwar Ranganathan
1560 1561     Varpu Rantala
1561 1562     Brodie Rao
1562 1563     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 ``DIRTYE`` when
   parsing
2 + GNU long name or link headers.

```

Comments 0