python / **cpython** Public

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

Commit 57f5981

3 people authored on Sep 2, 2025 · ✕ 58 / 63 · Verified

```
[3.10] gh-130577: tarfile now validates archives to ensure member offsets are
non-negative (GH-137027) (#137644)

gh-130577: tarfile now validates archives to ensure member offsets are non-negative (GH-
137027)


(cherry picked from commit 7040aa5)

Co-authored-by: Alexander Urieles <aeurielesn@users.noreply.github.com>
Co-authored-by: Gregory P. Smith <greg@krypto.org>
```

3.10 (#137644) · v3.10.20 v3.10.19

1 parent 1df5d00 commit 57f5981

3 files changed +162

Top

Filter files...

- Lib
  - tarfile.py
  - test
    - test_tarfile.py
- Misc/NEWS.d/next/Library
  - 2025-07-23-00-35-29.gh-issue-130577.c7ElTy.rst

Search within code

Lib/tarfile.py

```
@@ -1613,6 +1613,9 @@ def _block(self, count):
1613 1613          """Round up a byte count by BLOCKSIZE and return it,
1614 1614             e.g. _block(834) => 1024.
1615 1615          """
     1616 +        # Only non-negative offsets are allowed
     1617 +        if count < 0:
     1618 +            raise InvalidHeaderError("invalid offset")
1616 1619          blocks, remainder = divmod(count, BLOCKSIZE)
1617 1620          if remainder:
1618 1621              blocks += 1
```

Lib/test/test_tarfile.py

```
@@ -49,6 +49,7 @@ def sha256sum(data):
49 49  xzname = os.path.join(TEMPDIR, "testtar.tar.xz")
50 50  tmpname = os.path.join(TEMPDIR, "tmp.tar")
51 51  dotlessname = os.path.join(TEMPDIR, "testtar")
    52 + SPACE = b" "
52 53
53 54  sha256_regtype = (
54 55      "e09e4bc8b3c9d9177e77256353b36c159f5f040531bbd4b024a8f9b9196c71ce"
@@ -4273,6 +4274,161 @@ def valueerror_filter(tarinfo, path):
4273 4274              self.expect_exception(TypeError)  # errorlevel is not int
4274 4275
4275 4276
    4277 + class OffsetValidationTests(unittest.TestCase):
    4278 +     tarname = tmpname
    4279 +     invalid_posix_header = (
    4280 +         # name: 100 bytes
    4281 +         tarfile.NUL * tarfile.LENGTH_NAME
    4282 +         # mode, space, null terminator: 8 bytes
    4283 +         + b"000755" + SPACE + tarfile.NUL
    4284 +         # uid, space, null terminator: 8 bytes
    4285 +         + b"000001" + SPACE + tarfile.NUL
    4286 +         # gid, space, null terminator: 8 bytes
    4287 +         + b"000001" + SPACE + tarfile.NUL
    4288 +         # size, space: 12 bytes
    4289 +         + b"\xff" * 11 + SPACE
    4290 +         # mtime, space: 12 bytes
```

```
4291 +          + tarfile.NUL * 11 + SPACE
4292 +          # chksum: 8 bytes
4293 +          + b"0011407" + tarfile.NUL
4294 +          # type: 1 byte
4295 +          + tarfile.REGTYPE
4296 +          # linkname: 100 bytes
4297 +          + tarfile.NUL * tarfile.LENGTH_LINK
4298 +          # magic: 6 bytes, version: 2 bytes
4299 +          + tarfile.POSIX_MAGIC
4300 +          # uname: 32 bytes
4301 +          + tarfile.NUL * 32
4302 +          # gname: 32 bytes
4303 +          + tarfile.NUL * 32
4304 +          # devmajor, space, null terminator: 8 bytes
4305 +          + tarfile.NUL * 6 + SPACE + tarfile.NUL
4306 +          # devminor, space, null terminator: 8 bytes
4307 +          + tarfile.NUL * 6 + SPACE + tarfile.NUL
4308 +          # prefix: 155 bytes
4309 +          + tarfile.NUL * tarfile.LENGTH_PREFIX
4310 +          # padding: 12 bytes
4311 +          + tarfile.NUL * 12
4312 +      )
4313 +      invalid_gnu_header = (
4314 +          # name: 100 bytes
4315 +          tarfile.NUL * tarfile.LENGTH_NAME
4316 +          # mode, null terminator: 8 bytes
4317 +          + b"0000755" + tarfile.NUL
4318 +          # uid, null terminator: 8 bytes
4319 +          + b"0000001" + tarfile.NUL
4320 +          # gid, space, null terminator: 8 bytes
4321 +          + b"0000001" + tarfile.NUL
4322 +          # size, space: 12 bytes
4323 +          + b"\xff" * 11 + SPACE
4324 +          # mtime, space: 12 bytes
4325 +          + tarfile.NUL * 11 + SPACE
4326 +          # chksum: 8 bytes
4327 +          + b"0011327" + tarfile.NUL
4328 +          # type: 1 byte
4329 +          + tarfile.REGTYPE
4330 +          # linkname: 100 bytes
```

```
4331 +             + tarfile.NUL * tarfile.LENGTH_LINK
4332 +             # magic: 8 bytes
4333 +             + tarfile.GNU_MAGIC
4334 +             # uname: 32 bytes
4335 +             + tarfile.NUL * 32
4336 +             # gname: 32 bytes
4337 +             + tarfile.NUL * 32
4338 +             # devmajor, null terminator: 8 bytes
4339 +             + tarfile.NUL * 8
4340 +             # devminor, null terminator: 8 bytes
4341 +             + tarfile.NUL * 8
4342 +             # padding: 167 bytes
4343 +             + tarfile.NUL * 167
4344 +         )
4345 +         invalid_v7_header = (
4346 +             # name: 100 bytes
4347 +             tarfile.NUL * tarfile.LENGTH_NAME
4348 +             # mode, space, null terminator: 8 bytes
4349 +             + b"000755" + SPACE + tarfile.NUL
4350 +             # uid, space, null terminator: 8 bytes
4351 +             + b"000001" + SPACE + tarfile.NUL
4352 +             # gid, space, null terminator: 8 bytes
4353 +             + b"000001" + SPACE + tarfile.NUL
4354 +             # size, space: 12 bytes
4355 +             + b"\xff" * 11 + SPACE
4356 +             # mtime, space: 12 bytes
4357 +             + tarfile.NUL * 11 + SPACE
4358 +             # chksum: 8 bytes
4359 +             + b"0010070" + tarfile.NUL
4360 +             # type: 1 byte
4361 +             + tarfile.REGTYPE
4362 +             # linkname: 100 bytes
4363 +             + tarfile.NUL * tarfile.LENGTH_LINK
4364 +             # padding: 255 bytes
4365 +             + tarfile.NUL * 255
4366 +         )
4367 +         valid_gnu_header = tarfile.TarInfo("filename").tobuf(tarfile.GNU_FORMAT)
4368 +         data_block = b"\xff" * tarfile.BLOCKSIZE
4369 +
4370 +         def _write_buffer(self, buffer):
```

```
4371 +         with open(self.tarname, "wb") as f:
4372 +             f.write(buffer)
4373 +
4374 +     def _get_members(self, ignore_zeros=None):
4375 +         with open(self.tarname, "rb") as f:
4376 +             with tarfile.open(
4377 +                 mode="r", fileobj=f, ignore_zeros=ignore_zeros
4378 +             ) as tar:
4379 +                 return tar.getmembers()
4380 +
4381 +     def _assert_raises_read_error_exception(self):
4382 +         with self.assertRaisesRegex(
4383 +             tarfile.ReadError, "file could not be opened successfully"
4384 +         ):
4385 +             self._get_members()
4386 +
4387 +     def test_invalid_offset_header_validations(self):
4388 +         for tar_format, invalid_header in (
4389 +             ("posix", self.invalid_posix_header),
4390 +             ("gnu", self.invalid_gnu_header),
4391 +             ("v7", self.invalid_v7_header),
4392 +         ):
4393 +             with self.subTest(format=tar_format):
4394 +                 self._write_buffer(invalid_header)
4395 +                 self._assert_raises_read_error_exception()
4396 +
4397 +     def test_early_stop_at_invalid_offset_header(self):
4398 +         buffer = self.valid_gnu_header + self.invalid_gnu_header +
     self.valid_gnu_header
4399 +         self._write_buffer(buffer)
4400 +         members = self._get_members()
4401 +         self.assertEqual(len(members), 1)
4402 +         self.assertEqual(members[0].name, "filename")
4403 +         self.assertEqual(members[0].offset, 0)
4404 +
4405 +     def test_ignore_invalid_archive(self):
4406 +         # 3 invalid headers with their respective data
4407 +         buffer = (self.invalid_gnu_header + self.data_block) * 3
4408 +         self._write_buffer(buffer)
4409 +         members = self._get_members(ignore_zeros=True)
```

```
4410  +         self.assertEqual(len(members), 0)
4411  +
4412  +     def test_ignore_invalid_offset_headers(self):
4413  +         for first_block, second_block, expected_offset in (
4414  +             (
4415  +                 (self.valid_gnu_header),
4416  +                 (self.invalid_gnu_header + self.data_block),
4417  +                 0,
4418  +             ),
4419  +             (
4420  +                 (self.invalid_gnu_header + self.data_block),
4421  +                 (self.valid_gnu_header),
4422  +                 1024,
4423  +             ),
4424  +         ):
4425  +             self._write_buffer(first_block + second_block)
4426  +             members = self._get_members(ignore_zeros=True)
4427  +             self.assertEqual(len(members), 1)
4428  +             self.assertEqual(members[0].name, "filename")
4429  +             self.assertEqual(members[0].offset, expected_offset)
4430  +
4431  +
4276  4432    def setUpModule():
4277  4433        os_helper.unlink(TEMPDIR)
4278  4434        os.makedirs(TEMPDIR)
```

...5-07-23-00-35-29.gh-issue-130577.c7EITy.rst

```
•••   @@ -0,0 +1,3 @@
1   + :mod:`tarfile` now validates archives to ensure member offsets are
2   + non-negative.  (Contributed by Alexander Enrique Urieles Nieto in
3   + :gh:`130577`.)
```

Comments 0