python / **cpython** Public

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

Commit c9d9f78

3 people authored on Aug 4, 2025 · ✕ 83 / 86 · Verified

```
[3.12] gh-130577: tarfile now validates archives to ensure member offsets are
non-negative (GH-137027) (#137171)

(cherry picked from commit 7040aa5)

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

3.12 (#137171) · v3.12.13 v3.12.12

1 parent f66c75f commit c9d9f78

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.c7EITy.rst

Search within code

Lib/tarfile.py

```
@@ -1615,6 +1615,9 @@ def _block(self, count):
1615  1615          """Round up a byte count by BLOCKSIZE and return it,
1616  1616             e.g. _block(834) => 1024.
```

```
1617  1617            """
      1618  +         # Only non-negative offsets are allowed
      1619  +         if count < 0:
      1620  +             raise InvalidHeaderError("invalid offset")
1618  1621            blocks, remainder = divmod(count, BLOCKSIZE)
1619  1622            if remainder:
1620  1623                blocks += 1
```

Lib/test/test_tarfile.py

```
@@ -50,6 +50,7 @@ def sha256sum(data):
50    50      xzname = os.path.join(TEMPDIR, "testtar.tar.xz")
51    51      tmpname = os.path.join(TEMPDIR, "tmp.tar")
52    52      dotlessname = os.path.join(TEMPDIR, "testtar")
      53  + SPACE = b" "
53    54
54    55      sha256_regtype = (
55    56          "e09e4bc8b3c9d9177e77256353b36c159f5f040531bbd4b024a8f9b9196c71ce"
@@ -4488,6 +4489,161 @@ def extractall(self, ar):
4488  4489            ar.extractall(self.testdir, filter='fully_trusted')
4489  4490
4490  4491
      4492  + class OffsetValidationTests(unittest.TestCase):
      4493  +     tarname = tmpname
      4494  +     invalid_posix_header = (
      4495  +         # name: 100 bytes
      4496  +         tarfile.NUL * tarfile.LENGTH_NAME
      4497  +         # mode, space, null terminator: 8 bytes
      4498  +         + b"000755" + SPACE + tarfile.NUL
      4499  +         # uid, space, null terminator: 8 bytes
      4500  +         + b"000001" + SPACE + tarfile.NUL
      4501  +         # gid, space, null terminator: 8 bytes
      4502  +         + b"000001" + SPACE + tarfile.NUL
      4503  +         # size, space: 12 bytes
      4504  +         + b"\xff" * 11 + SPACE
      4505  +         # mtime, space: 12 bytes
      4506  +         + tarfile.NUL * 11 + SPACE
      4507  +         # chksum: 8 bytes
      4508  +         + b"0011407" + tarfile.NUL
```

```
4509 +         # type: 1 byte
4510 +         + tarfile.REGTYPE
4511 +         # linkname: 100 bytes
4512 +         + tarfile.NUL * tarfile.LENGTH_LINK
4513 +         # magic: 6 bytes, version: 2 bytes
4514 +         + tarfile.POSIX_MAGIC
4515 +         # uname: 32 bytes
4516 +         + tarfile.NUL * 32
4517 +         # gname: 32 bytes
4518 +         + tarfile.NUL * 32
4519 +         # devmajor, space, null terminator: 8 bytes
4520 +         + tarfile.NUL * 6 + SPACE + tarfile.NUL
4521 +         # devminor, space, null terminator: 8 bytes
4522 +         + tarfile.NUL * 6 + SPACE + tarfile.NUL
4523 +         # prefix: 155 bytes
4524 +         + tarfile.NUL * tarfile.LENGTH_PREFIX
4525 +         # padding: 12 bytes
4526 +         + tarfile.NUL * 12
4527 +     )
4528 +     invalid_gnu_header = (
4529 +         # name: 100 bytes
4530 +         tarfile.NUL * tarfile.LENGTH_NAME
4531 +         # mode, null terminator: 8 bytes
4532 +         + b"0000755" + tarfile.NUL
4533 +         # uid, null terminator: 8 bytes
4534 +         + b"0000001" + tarfile.NUL
4535 +         # gid, space, null terminator: 8 bytes
4536 +         + b"0000001" + tarfile.NUL
4537 +         # size, space: 12 bytes
4538 +         + b"\xff" * 11 + SPACE
4539 +         # mtime, space: 12 bytes
4540 +         + tarfile.NUL * 11 + SPACE
4541 +         # chksum: 8 bytes
4542 +         + b"0011327" + tarfile.NUL
4543 +         # type: 1 byte
4544 +         + tarfile.REGTYPE
4545 +         # linkname: 100 bytes
4546 +         + tarfile.NUL * tarfile.LENGTH_LINK
4547 +         # magic: 8 bytes
4548 +         + tarfile.GNU_MAGIC
```

```
4549 +         # uname: 32 bytes
4550 +         + tarfile.NUL * 32
4551 +         # gname: 32 bytes
4552 +         + tarfile.NUL * 32
4553 +         # devmajor, null terminator: 8 bytes
4554 +         + tarfile.NUL * 8
4555 +         # devminor, null terminator: 8 bytes
4556 +         + tarfile.NUL * 8
4557 +         # padding: 167 bytes
4558 +         + tarfile.NUL * 167
4559 +     )
4560 +     invalid_v7_header = (
4561 +         # name: 100 bytes
4562 +         tarfile.NUL * tarfile.LENGTH_NAME
4563 +         # mode, space, null terminator: 8 bytes
4564 +         + b"000755" + SPACE + tarfile.NUL
4565 +         # uid, space, null terminator: 8 bytes
4566 +         + b"000001" + SPACE + tarfile.NUL
4567 +         # gid, space, null terminator: 8 bytes
4568 +         + b"000001" + SPACE + tarfile.NUL
4569 +         # size, space: 12 bytes
4570 +         + b"\xff" * 11 + SPACE
4571 +         # mtime, space: 12 bytes
4572 +         + tarfile.NUL * 11 + SPACE
4573 +         # chksum: 8 bytes
4574 +         + b"0010070" + tarfile.NUL
4575 +         # type: 1 byte
4576 +         + tarfile.REGTYPE
4577 +         # linkname: 100 bytes
4578 +         + tarfile.NUL * tarfile.LENGTH_LINK
4579 +         # padding: 255 bytes
4580 +         + tarfile.NUL * 255
4581 +     )
4582 +     valid_gnu_header = tarfile.TarInfo("filename").tobuf(tarfile.GNU_FORMAT)
4583 +     data_block = b"\xff" * tarfile.BLOCKSIZE
4584 +
4585 +     def _write_buffer(self, buffer):
4586 +         with open(self.tarname, "wb") as f:
4587 +             f.write(buffer)
4588 +
```

```
4589 +     def _get_members(self, ignore_zeros=None):
4590 +         with open(self.tarname, "rb") as f:
4591 +             with tarfile.open(
4592 +                 mode="r", fileobj=f, ignore_zeros=ignore_zeros
4593 +             ) as tar:
4594 +                 return tar.getmembers()
4595 +
4596 +     def _assert_raises_read_error_exception(self):
4597 +         with self.assertRaisesRegex(
4598 +             tarfile.ReadError, "file could not be opened successfully"
4599 +         ):
4600 +             self._get_members()
4601 +
4602 +     def test_invalid_offset_header_validations(self):
4603 +         for tar_format, invalid_header in (
4604 +             ("posix", self.invalid_posix_header),
4605 +             ("gnu", self.invalid_gnu_header),
4606 +             ("v7", self.invalid_v7_header),
4607 +         ):
4608 +             with self.subTest(format=tar_format):
4609 +                 self._write_buffer(invalid_header)
4610 +                 self._assert_raises_read_error_exception()
4611 +
4612 +     def test_early_stop_at_invalid_offset_header(self):
4613 +         buffer = self.valid_gnu_header + self.invalid_gnu_header +
     self.valid_gnu_header
4614 +         self._write_buffer(buffer)
4615 +         members = self._get_members()
4616 +         self.assertEqual(len(members), 1)
4617 +         self.assertEqual(members[0].name, "filename")
4618 +         self.assertEqual(members[0].offset, 0)
4619 +
4620 +     def test_ignore_invalid_archive(self):
4621 +         # 3 invalid headers with their respective data
4622 +         buffer = (self.invalid_gnu_header + self.data_block) * 3
4623 +         self._write_buffer(buffer)
4624 +         members = self._get_members(ignore_zeros=True)
4625 +         self.assertEqual(len(members), 0)
4626 +
4627 +     def test_ignore_invalid_offset_headers(self):
```

```
4628  +         for first_block, second_block, expected_offset in (
4629  +             (
4630  +                 (self.valid_gnu_header),
4631  +                 (self.invalid_gnu_header + self.data_block),
4632  +                 0,
4633  +             ),
4634  +             (
4635  +                 (self.invalid_gnu_header + self.data_block),
4636  +                 (self.valid_gnu_header),
4637  +                 1024,
4638  +             ),
4639  +         ):
4640  +             self._write_buffer(first_block + second_block)
4641  +             members = self._get_members(ignore_zeros=True)
4642  +             self.assertEqual(len(members), 1)
4643  +             self.assertEqual(members[0].name, "filename")
4644  +             self.assertEqual(members[0].offset, expected_offset)
4645  +
4646  +
4491  4647      def setUpModule():
4492  4648          os_helper.unlink(TEMPDIR)
4493  4649          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