python / **cpython** Public

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

Commit `fbc2a0c`

3 people authored on Jul 28, 2025 · ✕ 31 / 35 · Verified

```
[3.14] gh-130577: tarfile now validates archives to ensure member offsets are
non-negative (GH-137027) (#137169)

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

3.14 (#137169) · v3.14.5rc1 ••• v3.14.0rc2

1 parent 909a534 commit fbc2a0c

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

```
@@ -1647,6 +1647,9 @@ def _block(self, count):
1647  1647          """Round up a byte count by BLOCKSIZE and return it,
1648  1648             e.g. _block(834) => 1024.
1649  1649          """
```

```
      1650 +         # Only non-negative offsets are allowed
      1651 +         if count < 0:
      1652 +             raise InvalidHeaderError("invalid offset")
1650  1653           blocks, remainder = divmod(count, BLOCKSIZE)
1651  1654           if remainder:
1652  1655               blocks += 1
```

Lib/test/test_tarfile.py

```
                   @@ -55,6 +55,7 @@ def sha256sum(data):
55    55       zstname = os.path.join(TEMPDIR, "testtar.tar.zst")
56    56       tmpname = os.path.join(TEMPDIR, "tmp.tar")
57    57       dotlessname = os.path.join(TEMPDIR, "testtar")
      58 + SPACE = b" "
58    59
59    60       sha256_regtype = (
60    61           "e09e4bc8b3c9d9177e77256353b36c159f5f040531bbd4b024a8f9b9196c71ce"
                   @@ -4602,6 +4603,161 @@ def extractall(self, ar):
4602  4603           ar.extractall(self.testdir, filter='fully_trusted')
4603  4604
4604  4605
      4606 + class OffsetValidationTests(unittest.TestCase):
      4607 +     tarname = tmpname
      4608 +     invalid_posix_header = (
      4609 +         # name: 100 bytes
      4610 +         tarfile.NUL * tarfile.LENGTH_NAME
      4611 +         # mode, space, null terminator: 8 bytes
      4612 +         + b"000755" + SPACE + tarfile.NUL
      4613 +         # uid, space, null terminator: 8 bytes
      4614 +         + b"000001" + SPACE + tarfile.NUL
      4615 +         # gid, space, null terminator: 8 bytes
      4616 +         + b"000001" + SPACE + tarfile.NUL
      4617 +         # size, space: 12 bytes
      4618 +         + b"\xff" * 11 + SPACE
      4619 +         # mtime, space: 12 bytes
      4620 +         + tarfile.NUL * 11 + SPACE
      4621 +         # chksum: 8 bytes
      4622 +         + b"0011407" + tarfile.NUL
      4623 +         # type: 1 byte
```

```
4624 +          + tarfile.REGTYPE
4625 +          # linkname: 100 bytes
4626 +          + tarfile.NUL * tarfile.LENGTH_LINK
4627 +          # magic: 6 bytes, version: 2 bytes
4628 +          + tarfile.POSIX_MAGIC
4629 +          # uname: 32 bytes
4630 +          + tarfile.NUL * 32
4631 +          # gname: 32 bytes
4632 +          + tarfile.NUL * 32
4633 +          # devmajor, space, null terminator: 8 bytes
4634 +          + tarfile.NUL * 6 + SPACE + tarfile.NUL
4635 +          # devminor, space, null terminator: 8 bytes
4636 +          + tarfile.NUL * 6 + SPACE + tarfile.NUL
4637 +          # prefix: 155 bytes
4638 +          + tarfile.NUL * tarfile.LENGTH_PREFIX
4639 +          # padding: 12 bytes
4640 +          + tarfile.NUL * 12
4641 +      )
4642 +      invalid_gnu_header = (
4643 +          # name: 100 bytes
4644 +          tarfile.NUL * tarfile.LENGTH_NAME
4645 +          # mode, null terminator: 8 bytes
4646 +          + b"0000755" + tarfile.NUL
4647 +          # uid, null terminator: 8 bytes
4648 +          + b"0000001" + tarfile.NUL
4649 +          # gid, space, null terminator: 8 bytes
4650 +          + b"0000001" + tarfile.NUL
4651 +          # size, space: 12 bytes
4652 +          + b"\xff" * 11 + SPACE
4653 +          # mtime, space: 12 bytes
4654 +          + tarfile.NUL * 11 + SPACE
4655 +          # chksum: 8 bytes
4656 +          + b"0011327" + tarfile.NUL
4657 +          # type: 1 byte
4658 +          + tarfile.REGTYPE
4659 +          # linkname: 100 bytes
4660 +          + tarfile.NUL * tarfile.LENGTH_LINK
4661 +          # magic: 8 bytes
4662 +          + tarfile.GNU_MAGIC
4663 +          # uname: 32 bytes
```

```
4664 +          + tarfile.NUL * 32
4665 +          # gname: 32 bytes
4666 +          + tarfile.NUL * 32
4667 +          # devmajor, null terminator: 8 bytes
4668 +          + tarfile.NUL * 8
4669 +          # devminor, null terminator: 8 bytes
4670 +          + tarfile.NUL * 8
4671 +          # padding: 167 bytes
4672 +          + tarfile.NUL * 167
4673 +      )
4674 +      invalid_v7_header = (
4675 +          # name: 100 bytes
4676 +          tarfile.NUL * tarfile.LENGTH_NAME
4677 +          # mode, space, null terminator: 8 bytes
4678 +          + b"000755" + SPACE + tarfile.NUL
4679 +          # uid, space, null terminator: 8 bytes
4680 +          + b"000001" + SPACE + tarfile.NUL
4681 +          # gid, space, null terminator: 8 bytes
4682 +          + b"000001" + SPACE + tarfile.NUL
4683 +          # size, space: 12 bytes
4684 +          + b"\xff" * 11 + SPACE
4685 +          # mtime, space: 12 bytes
4686 +          + tarfile.NUL * 11 + SPACE
4687 +          # chksum: 8 bytes
4688 +          + b"0010070" + tarfile.NUL
4689 +          # type: 1 byte
4690 +          + tarfile.REGTYPE
4691 +          # linkname: 100 bytes
4692 +          + tarfile.NUL * tarfile.LENGTH_LINK
4693 +          # padding: 255 bytes
4694 +          + tarfile.NUL * 255
4695 +      )
4696 +      valid_gnu_header = tarfile.TarInfo("filename").tobuf(tarfile.GNU_FORMAT)
4697 +      data_block = b"\xff" * tarfile.BLOCKSIZE
4698 +
4699 +      def _write_buffer(self, buffer):
4700 +          with open(self.tarname, "wb") as f:
4701 +              f.write(buffer)
4702 +
4703 +      def _get_members(self, ignore_zeros=None):
```

```
4704 +         with open(self.tarname, "rb") as f:
4705 +             with tarfile.open(
4706 +                 mode="r", fileobj=f, ignore_zeros=ignore_zeros
4707 +             ) as tar:
4708 +                 return tar.getmembers()
4709 +
4710 +     def _assert_raises_read_error_exception(self):
4711 +         with self.assertRaisesRegex(
4712 +             tarfile.ReadError, "file could not be opened successfully"
4713 +         ):
4714 +             self._get_members()
4715 +
4716 +     def test_invalid_offset_header_validations(self):
4717 +         for tar_format, invalid_header in (
4718 +             ("posix", self.invalid_posix_header),
4719 +             ("gnu", self.invalid_gnu_header),
4720 +             ("v7", self.invalid_v7_header),
4721 +         ):
4722 +             with self.subTest(format=tar_format):
4723 +                 self._write_buffer(invalid_header)
4724 +                 self._assert_raises_read_error_exception()
4725 +
4726 +     def test_early_stop_at_invalid_offset_header(self):
4727 +         buffer = self.valid_gnu_header + self.invalid_gnu_header +
       self.valid_gnu_header
4728 +         self._write_buffer(buffer)
4729 +         members = self._get_members()
4730 +         self.assertEqual(len(members), 1)
4731 +         self.assertEqual(members[0].name, "filename")
4732 +         self.assertEqual(members[0].offset, 0)
4733 +
4734 +     def test_ignore_invalid_archive(self):
4735 +         # 3 invalid headers with their respective data
4736 +         buffer = (self.invalid_gnu_header + self.data_block) * 3
4737 +         self._write_buffer(buffer)
4738 +         members = self._get_members(ignore_zeros=True)
4739 +         self.assertEqual(len(members), 0)
4740 +
4741 +     def test_ignore_invalid_offset_headers(self):
4742 +         for first_block, second_block, expected_offset in (
```

```
4743 +             (
4744 +                 (self.valid_gnu_header),
4745 +                 (self.invalid_gnu_header + self.data_block),
4746 +                 0,
4747 +             ),
4748 +             (
4749 +                 (self.invalid_gnu_header + self.data_block),
4750 +                 (self.valid_gnu_header),
4751 +                 1024,
4752 +             ),
4753 +         ):
4754 +             self._write_buffer(first_block + second_block)
4755 +             members = self._get_members(ignore_zeros=True)
4756 +             self.assertEqual(len(members), 1)
4757 +             self.assertEqual(members[0].name, "filename")
4758 +             self.assertEqual(members[0].offset, expected_offset)
4759 +
4760 +
4605 4761     def setUpModule():
4606 4762         os_helper.unlink(TEMPDIR)
4607 4763         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