» Issue35907



This issue tracker **has been migrated to GitHub**, and is currently **read-only**.
For more information, **see the GitHub FAQs in the Python's Developer Guide.**



This issue has been migrated to GitHub:
https://github.com/python/cpython/issues/80088

classification

Title:	[security][CVE-2019-9948] Unnecessary URL scheme exists to allow local_file:// reading file in urllib		
Type:	security	**Stage**:	resolved
Components:	Library (Lib)	**Versions**:	Python 3.8, Python 3.7, Python 2.7

process

Status:	closed	**Resolution**:	fixed
Dependencies:		**Superseder**:	
Assigned To:		**Nosy List**:	Petter S, christian.heimes, cstratak, larry, martin.panter, matrixise, ned.deily, push0ebp, vstinner, ware, xtreak
Priority:	normal	**Keywords**:	patch

Created on **2019-02-06 08:19** by **push0ebp**, last changed **2022-04-11 14:59** by **admin**. This issue is now **closed**.

Pull Requests

URL	Status	Linked	Edit
PR 11842	merged	push0ebp, 2019-02-13 17:11	
PR 13474	closed	vstinner, 2019-05-21 21:29	
PR 13505	merged	vstinner, 2019-05-22 20:24	
PR 13506	merged	vstinner, 2019-05-22 20:32	
PR 13510	merged	vstinner, 2019-05-22 21:30	
PR 13513	merged	vstinner, 2019-05-22 22:19	
PR 13523	merged	vstinner, 2019-05-23 11:39	
PR 13557	merged	vstinner, 2019-05-24 20:10	
PR 13558	merged	vstinner, 2019-05-24 20:27	
PR 13559	merged	vstinner, 2019-05-24 20:31	

Messages (27)

msg334905 - (view) **Author: Sihoon Lee (push0ebp) *** **Date: 2019-02-06 08:19**

```
The Unnecessary scheme exists in urlopen() urllib

when people would protect to read file system in HTTP request of urlopen(), they
often filter like this against SSRF.

# Vulnerability PoC
import urllib
print urllib.urlopen('local_file:///etc/passwd').read()[:30]
the result is
##
# User Database
#
# Note t


but if we use a scheme like this, parsing URL cannot parse scheme with urlparse()
this is the parsed result.
ParseResult(scheme='', netloc='', path='local_file:/etc/passwd', params='',
query='', fragment='')


def request(url):
    from urllib import urlopen
    from urlparse import urlparse

    result = urlparse(url)
    scheme = result.scheme
    if not scheme:
        return False #raise Exception("Required scheme")
    if scheme == 'file':
        return False #raise Exception("Don't open file")
    res = urlopen(url)
    content = res.read()
    print url, content[:30]
    return True

assert request('file:///etc/passwd') == False
assert request(' file:///etc/passwd') == False
assert request('File:///etc/passwd') == False
assert request('http://www.google.com') != False

if they filter only file://, this mitigation can be bypassed against SSRF.
with this way.

assert request('local-file:/etc/passwd') == True
ParseResult(scheme='local-file', netloc='', path='/etc/passwd', params='',
query='', fragment='')
parseing URL also can be passed.


# Attack scenario
this is the unnecessary URL scheme("local_file").
even if it has filtering, An Attacker can read arbitrary files as bypassing with
it.
```

```
# Root Cause

URLopener::open in urllib.py
from 203 lin

name = 'open_' + urltype
self.type = urltype
name = name.replace('-', '_') #it can also allows local-file
if not hasattr(self, name): #passed here hasattr(URLopener, 'open_local_file')
    if proxy:
        return self.open_unknown_proxy(proxy, fullurl, data)
    else:
        return self.open_unknown(fullurl, data)
try:
    if data is None:
        return getattr(self, name)(url)
    else:
        return getattr(self, name)(url, data) #return URLopener::open_local_file

it may be just trick because people usually use whitelist (allow only http or
https.
Even if but anyone may use blacklist like filtering file://, they will be
affected with triggering SSRF
```

msg334923 - (view) **Author: Christian Heimes (christian.heimes) *** **Date: 2019-02-06 10:53**

```
Thanks for your report. I'm having a hard time understanding your English. If I
understand you correctly, your bug report is about the open_local_file() method
and the surprising fact that urllib supports the local_file schema.

I agree, this looks like an implementation artefact. urllib should not expose the
local_file schema. In Python 3 refuses local_file:// (tested with 3.4 to 3.7).

>>> import urllib.request
>>> urllib.request.urlopen('local_file:///etc/passwd').read()[:30]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/usr/lib64/python3.6/urllib/request.py", line 223, in urlopen
    return opener.open(url, data, timeout)
  File "/usr/lib64/python3.6/urllib/request.py", line 526, in open
    response = self._open(req, data)
  File "/usr/lib64/python3.6/urllib/request.py", line 549, in _open
    'unknown_open', req)
  File "/usr/lib64/python3.6/urllib/request.py", line 504, in _call_chain
    result = func(*args)
  File "/usr/lib64/python3.6/urllib/request.py", line 1388, in unknown_open
    raise URLError('unknown url type: %s' % type)
urllib.error.URLError: <urlopen error unknown url type: local_file>
```

msg334925 - (view) **Author: Christian Heimes (christian.heimes) *** **Date: 2019-02-06 10:55**

```
Only the Python 2 urllib module is affected. Python 2.7's urllib2 also correctly
fails with local_file://
```

```
>>> import urllib2
>>> urllib2.urlopen('local_file:///etc/passwd').read()[:30]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/home/heimes/dev/python/2.7/Lib/urllib2.py", line 154, in urlopen
    return opener.open(url, data, timeout)
  File "/home/heimes/dev/python/2.7/Lib/urllib2.py", line 429, in open
    response = self._open(req, data)
  File "/home/heimes/dev/python/2.7/Lib/urllib2.py", line 452, in _open
    'unknown_open', req)
  File "/home/heimes/dev/python/2.7/Lib/urllib2.py", line 407, in _call_chain
    result = func(*args)
  File "/home/heimes/dev/python/2.7/Lib/urllib2.py", line 1266, in unknown_open
    raise URLError('unknown url type: %s' % type)
urllib2.URLError: <urlopen error unknown url type: local_file>
```

msg334927 - (view) **Author: Sihoon Lee (push0ebp) *** **Date: 2019-02-06 11:28**

Sorry for my bad English.
Yes, exactly. Only python 2.7 has been affected. not python3.
So I chose only Python2.7 version.

msg334928 - (view) **Author: Sihoon Lee (push0ebp) *** **Date: 2019-02-06 11:29**

and only urllib, not urllib2.

msg334929 - (view) **Author: Christian Heimes (christian.heimes) *** **Date: 2019-02-06 11:33**

I'm not a native English speaker either. I wasn't sure if I understood you correctly. Thanks!

msg334930 - (view) **Author: Sihoon Lee (push0ebp) *** **Date: 2019-02-06 11:42**

I am not also native English speaker. It's OK. Thank you for reading my report

msg339664 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2019-04-08 17:52**

This issue seems to have been assigned CVE-2019-9948 (**https://nvd.nist.gov/vuln/detail/CVE-2019-9948**) as noted in **https://github.com/python/cpython/pull/11842#issuecomment-480930608**

msg342334 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2019-05-13 14:47**

Christian:
> I agree, this looks like an implementation artefact. urllib should not expose the local_file schema. In Python 3 refuses local_file:// (tested with 3.4 to 3.7).

I'm not sure that I understand well the issue. urllib accepts various scheme by design: HTTP, HTTPS, FTP, FILE, etc.

For example, file:// scheme is legit and works as expected. Python 3 example:

```
import urllib.request
```

```
req = urllib.request.Request('file:///etc/passwd')
print(f"URL scheme: {req.type}")
fp = urllib.request.urlopen(req)
print(fp.read()[:30])
fp.close()
---

Output with Python 3:
---
URL scheme: file
b'root:x:0:0:root:/root:/bin/bas'
---


I get a similar output with this Python 2 example:
---
import urllib
req = urllib.urlopen('file:///etc/passwd')
print(req.read()[:30])
req.close()
---


Christian:
> I agree, this looks like an implementation artefact. urllib should not expose
the local_file schema.

I understand that Python 2 handles local_file://url as file://url. Ok. But is
this a security issue? If you care of security, you ensure that the url scheme is
HTTP or HTTPS, not only forbid FILE, no?

I'm asking because of:

Karthikeyan Singaravelan:
> This issue seems to have been assigned CVE-2019-9948
(https://nvd.nist.gov/vuln/detail/CVE-2019-9948) ...
```

msg342336 - (view) **Author: Christian Heimes (christian.heimes) *** **Date: 2019-05-13 14:53**

```
The issue is not about whether "file://" schema or not.

It's about the fact that urllib on Python 2 has two schemas that allow local file
access. There is the well-known "file://" schema and there is the implementation
artifact "local_file://". A careful, security-minded developer knows about the
file:// schema and also knows how to block it. But the "local_file://" schema is
a surprising side-effect of the implementation.
```

msg342337 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2019-05-13 15:09**

```
If you use directly the URLopener class, Python 3 has a similar issue:
---
import urllib.request
req = urllib.request.URLopener().open('local_file:///etc/passwd')
print(req.read()[:30])
```

```
req.close()
---
```

msg342363 - (view) **Author: Sihoon Lee (push0ebp) *** **Date: 2019-05-13 17:10**

If developers allow only http:// or https:// as whitelist, it has no problem.
But, If someone blocks only one file://, attacker can bypass it.
This issue may provides attacker with bypassing method as new scheme.

msg343098 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2019-05-21 21:12**

New changeset **b15bde8058e821b383d81fcae68b335a752083ca** by Victor Stinner (SH) in branch '2.7':
~~**bpo-35907**~~, CVE-2019-9948: urllib rejects local_file:// scheme (~~**GH-11842**~~)
https://github.com/python/cpython/commit/b15bde8058e821b383d81fcae68b335a752083ca

msg343233 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2019-05-22 20:15**

New changeset **0c2b6a3943aa7b022e8eb4bfd9bffcddebf9a587** by Victor Stinner in branch 'master':
~~**bpo-35907**~~, CVE-2019-9948: urllib rejects local_file:// scheme (~~**GH-13474**~~)
https://github.com/python/cpython/commit/0c2b6a3943aa7b022e8eb4bfd9bffcddebf9a587

msg343239 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2019-05-22 21:28**

New changeset **942c31dffbe886ff02e25a319cc3891220b8c641** by Victor Stinner in branch '2.7':
~~**bpo-35907**~~: Complete test_urllib.test_local_file_open() (~~**GH-13506**~~)
https://github.com/python/cpython/commit/942c31dffbe886ff02e25a319cc3891220b8c641

msg343241 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2019-05-22 21:28**

New changeset **34bab215596671d0dec2066ae7d7450cd73f638b** by Victor Stinner in branch '3.7':
~~**bpo-35907**~~, CVE-2019-9948: urllib rejects local_file:// scheme (~~**GH-13474**~~) (~~**GH-13505**~~)
https://github.com/python/cpython/commit/34bab215596671d0dec2066ae7d7450cd73f638b

msg343424 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2019-05-24 20:06**

New changeset **deffee57749cf29ba17f50f11fb2a8cbc3e3752d** by Victor Stinner in branch 'master':
~~**bpo-35907**~~: Clarify the NEWS entry (~~**GH-13523**~~)
https://github.com/python/cpython/commit/deffee57749cf29ba17f50f11fb2a8cbc3e3752d

msg343427 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2019-05-24 21:06**

New changeset **1c9debd2366a21525769aaa99ce334092033a963** by Victor Stinner in branch 'master':
~~**bpo-35907**~~: Fix typo in the NEWS entry (~~**GH-13559**~~)
https://github.com/python/cpython/commit/1c9debd2366a21525769aaa99ce334092033a963

msg343431 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2019-05-24 21:28**

New changeset **d9d1045837e5356331b6d5e24cbd1286acb62b5d** by Victor Stinner in branch '2.7':
~~**bpo-35907**~~: Clarify the NEWS entry (~~**GH-13557**~~)
https://github.com/python/cpython/commit/d9d1045837e5356331b6d5e24cbd1286acb62b5d

msg343432 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2019-05-24 21:29**

New changeset **cee4ac8135fe9cf99de4ceca52d1f53e14b69dba** by Victor Stinner in branch '3.7':
~~**bpo-35907**~~: Clarify the NEWS entry (~~**GH-13558**~~)
https://github.com/python/cpython/commit/cee4ac8135fe9cf99de4ceca52d1f53e14b69dba

msg343856 - (view) **Author: Ned Deily (ned.deily) *** **Date: 2019-05-29 02:30**

New changeset **4f06dae5d8d4400ba38d8502da620f07d4a5696e** by Ned Deily (Victor Stinner) in branch '3.6':
~~**bpo-35907**~~, CVE-2019-9948: urllib rejects local_file:// scheme (~~**GH-13513**~~)
https://github.com/python/cpython/commit/4f06dae5d8d4400ba38d8502da620f07d4a5696e

msg347867 - (view) **Author: Larry Hastings (larry) *** **Date: 2019-07-14 07:04**

New changeset **4fe82a8eef7aed60de05bfca0f2c322730ea921e** by larryhastings (Victor Stinner) in branch '3.5':
~~**bpo-35907**~~, CVE-2019-9948: urllib rejects local_file:// scheme (~~**GH-13474**~~) (~~**GH-13505**~~) (~~**#13510**~~)
https://github.com/python/cpython/commit/4fe82a8eef7aed60de05bfca0f2c322730ea921e

msg368011 - (view) **Author: Petter S (Petter S) *** **Date: 2020-05-04 06:49**

We should whitelist the protocols. The current solution with `getattr` is really fragile.

For example, this crashes with a `TypeError`:
`URLopener().open("unknown_proxy://test")`

msg368063 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2020-05-04 16:31**

> We should whitelist the protocols. The current solution with `getattr` is really fragile. For example, this crashes with a `TypeError`:
`URLopener().open("unknown_proxy://test")`

Would you mind to elaborate why do you consider that the solution is incomplete? Your issue doesn't show that Python is vulnerable. TypeError *is* the expected behavior.

Would you prefer another error message? If yes, please open a seperated issue.

msg368080 - (view) **Author: Petter S (Petter S) *** **Date: 2020-05-04 20:20**

The solution is incomplete because it fixes just this single security issue, not the inherent fragility of this file.

If, in the future someone happens to add another method starting with open to

this class, we are at risk of having the same problem again.

As for the error message, it is of course a minor issue, but I don't think it is expected that "unknown_proxy://" and "something_else://" raise different exceptions, right?

msg369228 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2020-05-18 14:16**

> The solution is incomplete because it fixes just this single security issue, not the inherent fragility of this file.

If you want to propose a change to make the file "less fragile", please open a *new* separated issue.

The issue is about an exact vulnerability, the "local_file://" scheme, which has been fixed. I close again the issue.

msg369292 - (view) **Author: Petter S (Petter S) *** **Date: 2020-05-18 21:31**

OK: **https://bugs.python.org/issue40673**

History

Date	User	Action	Args
2022-04-11 14:59:10	admin	set	github: 80088
2020-05-18 21:31:43	Petter S	set	messages: + **msg369292**
2020-05-18 14:16:27	vstinner	set	status: open -> closed resolution: fixed messages: + **msg369228** stage: patch review -> resolved
2020-05-04 20:20:05	Petter S	set	messages: + **msg368080**
2020-05-04 16:31:12	vstinner	set	messages: + **msg368063**
2020-05-04 06:49:11	Petter S	set	nosy: + **Petter S** messages: + **msg368011**
2019-07-14 07:04:22	larry	set	nosy: + **larry** messages: + **msg347867**
2019-05-29 02:30:52	ned.deily	set	nosy: + **ned.deily** messages: + **msg343856**
2019-05-24 21:29:13	vstinner	set	messages: + **msg343432**
2019-05-24 21:28:59	vstinner	set	messages: + **msg343431**
2019-05-24 21:06:28	vstinner	set	messages: + **msg343427**
2019-05-24 20:31:37	vstinner	set	pull_requests: + **pull_request13470**
2019-05-24 20:27:15	vstinner	set	pull_requests: + **pull_request13469**

2019-05-24 20:10:41	vstinner	set	pull_requests: + **pull_request13468**
2019-05-24 20:06:37	vstinner	set	messages: + **msg343424**
2019-05-23 11:39:29	vstinner	set	pull_requests: + **pull_request13438**
2019-05-22 22:19:41	vstinner	set	pull_requests: + **pull_request13430**
2019-05-22 21:30:39	vstinner	set	pull_requests: + **pull_request13426**
2019-05-22 21:28:30	vstinner	set	messages: + **msg343241**
2019-05-22 21:28:06	vstinner	set	messages: + **msg343239**
2019-05-22 20:32:22	vstinner	set	pull_requests: + **pull_request13422**
2019-05-22 20:24:17	vstinner	set	pull_requests: + **pull_request13421**
2019-05-22 20:15:18	vstinner	set	messages: + **msg343233**
2019-05-21 21:29:51	vstinner	set	pull_requests: + **pull_request13386**
2019-05-21 21:12:44	vstinner	set	messages: + **msg343098**
2019-05-13 17:10:33	push0ebp	set	messages: + **msg342363**
2019-05-13 15:15:56	christian.heimes	set	title: [security][CVE-2019-9948] Unnecessary URL scheme exists to allow file:// reading file in urllib -> [security][CVE-2019-9948] Unnecessary URL scheme exists to allow local_file:// reading file in urllib
2019-05-13 15:09:49	vstinner	set	messages: + **msg342337** versions: + Python 3.7, Python 3.8
2019-05-13 14:53:28	christian.heimes	set	messages: + **msg342336**
2019-05-13 14:47:47	vstinner	set	nosy: + **vstinner** messages: + **msg342334**
2019-05-13 12:59:28	vstinner	set	title: Unnecessary URL scheme exists to allow file:// reading file in urllib -> [security][CVE-2019-9948] Unnecessary URL scheme exists to allow file:// reading file in urllib
2019-04-29 10:58:10	cstratak	set	nosy: + **cstratak**
2019-04-09 15:46:06	ware	set	nosy: + **ware**
2019-04-08 17:52:36	xtreak	set	nosy: + **xtreak** messages: + **msg339664**
2019-02-13 17:11:04	push0ebp	set	keywords: + **patch** stage: needs patch -> patch review pull_requests: + **pull_request11872**
2019-02-06 11:42:22	push0ebp	set	messages: + **msg334930**
2019-02-06 11:33:37	christian.heimes	set	messages: + **msg334929**

2019-02-06 11:29:51	push0ebp	set	messages: + **msg334928**
2019-02-06 11:28:42	push0ebp	set	messages: + **msg334927**
2019-02-06 10:55:51	christian.heimes	set	messages: + **msg334925**
2019-02-06 10:53:52	christian.heimes	set	messages: + **msg334923** stage: needs patch
2019-02-06 08:57:57	matrixise	set	nosy: + **christian.heimes**, **martin.panter**, **matrixise**
2019-02-06 08:19:51	push0ebp	create	