» Issue35121



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/79302

classification

Title:	[CVE-2018-20852] Cookie domain check returns incorrect results		
Type:	security	**Stage**:	resolved
Components:	Library (Lib)	**Versions**:	Python 3.8, Python 3.7, Python 3.6, Python 3.4, Python 3.5, Python 2.7

process

Status:	closed	**Resolution**:	fixed
Dependencies:		**Superseder**:	
Assigned To:	benjamin.peterson	**Nosy List**:	Windson Yang, benjamin.peterson, larry, lukasz.langa, martin.panter, miss-islington, ned.deily, orsenthil, rschiron, serhiy.storchaka, vstinner, xtreak, 西田雄治
Priority:	release blocker	**Keywords**:	patch, security_issue

Created on **2018-10-31 06:52** by **西田雄治**, last changed **2022-04-11 14:59** by **admin**. This issue is now **closed**.

Pull Requests

URL	Status	Linked	Edit
PR 10258	merged	xtreak, 2018-10-31 10:29	
PR 12260	merged	miss-islington, 2019-03-10 02:10	
PR 12261	merged	miss-islington, 2019-03-10 02:36	
PR 12279	merged	xtreak, 2019-03-11 15:54	
PR 12281	merged	xtreak, 2019-03-11 16:02	
PR 13426	merged	xtreak, 2019-05-19 19:09	

Messages (30)

msg328973 - (view) **Author: 西田雄治 (西田雄治)** **Date: 2018-10-31 06:52**

```
http.cookiejar.DefaultPolicy.domain_return_ok returns incorrect results.

So, HTTP clients send cookies which issued from wrong server.
```

```
policy = http.cookiejar.DefaultCookiePolicy()
req = urllib.request.Request('https://xxxfoo.co.jp/')
print(policy.domain_return_ok('foo.co.jp', req)   # should be False, but it returns
True
```

msg328975 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2018-10-31 08:15**

The current set of tests are at https://github.com/python/cpython/blob/0353b4eaaf451ad463ce7eb3074f6b62d332f401/Lib/test/test_http_cookiejar.py#L406 . A simple set of tuple that can be added based on the report as below :

```
("http://barfoo.com", ".foo.com", False)
("http://barfoo.com", "foo.com", False) # Fails on master
```

The check is done at https://github.com/python/cpython/blob/0353b4eaaf451ad463ce7eb3074f6b62d332f401/Lib/http/cookiejar.py#L1176 . There is no check to add '.' before domain if absent. Hence it performs a substring match with the values req_host = ".barfoo.com" and erhn = ".barfoo.com" and domain = "foo.com" so the condition `not (req_host.endswith(domain) or erhn.endswith(domain))` fails and doesn't return False. I would suggest adding a check to make sure domain also starts with '.' similar to req_host and erhn thus fixing the issue. I tried the fix and existing tests along with the reported case works fine.

```
diff --git a/Lib/http/cookiejar.py b/Lib/http/cookiejar.py
index 0ba8200f32..da7462701b 100644
--- a/Lib/http/cookiejar.py
+++ b/Lib/http/cookiejar.py
@@ -1173,6 +1173,8 @@ class DefaultCookiePolicy(CookiePolicy):
             req_host = "."+req_host
         if not erhn.startswith("."):
             erhn = "."+erhn
+        if not domain.startswith("."):
+            domain = "."+domain
         if not (req_host.endswith(domain) or erhn.endswith(domain)):
             #_debug("   request domain %s does not match cookie domain %s",
             #       req_host, domain)
```

```
("http://barfoo.com", ".foo.com", False)
("http://barfoo.com", "foo.com", False) # Tests pass with fix
```

Also tried the script attached in the report

```
$ cat ../backups/bpo35121.py

import urllib
from http.cookiejar import DefaultCookiePolicy

policy = DefaultCookiePolicy()
req = urllib.request.Request('https://xxxfoo.co.jp/')
print(policy.domain_return_ok('foo.co.jp', req))

# without fix

$ ./python.exe ../backups/bpo35121.py
```

```
True

# With domain fix

$ ./python.exe ../backups/bpo35121.py
False
```

The check was added in 2004 with commit **2a6ba9097ee3942ae328befaf074ce9722b93ca0** .
If my fix is correct I am willing to raise a PR for this with test.

Hope it helps!

msg328981 - (view) **Author: 西田雄治 (西田雄治)** **Date: 2018-10-31 10:24**

I think that is desired result. thanks!

msg328985 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2018-10-31 11:27**

Thanks for the confirmation. I have created a PR (**https://github.com/python/cpython/pull/10258**) with test and added 3.8 as affected version. Please add in if I have missed anything in the PR.

msg329176 - (view) **Author: Windson Yang (Windson Yang) *** **Date: 2018-11-03 02:12**

I wonder
https://github.com/python/cpython/blob/master/Lib/test/test_http_cookiejar.py#L420

("**http://foo.bar.com/**", "com", True),
("**http://foo.com/**", "com", True),

are expected behavior?

msg329179 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2018-11-03 04:02**

Good catch Windson! I overlooked the tests. There is also a comment that it's liberal in the test function. Since the code was added in 2006 I don't if it's ok broken to fix this or not. I will let the reviewers take a call then. There is also domain_match and user_domain_match that perform more tests.

> This is only a rough check for performance reasons, so it's not too critical as long as it's sufficiently liberal.

Thanks for your input!

msg332299 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2018-12-21 20:57**

Looking further into this the domain validation makes it little more stricter and can have wider implications. For example requests library uses cookiejar to maintain cookies between sessions. One more case is that `domain` can be empty so only non-empty domains can be prefixed with dot.

A simple server that sets Cookie with value `A=LDJDSFLKSDJLDSF`

```
import SimpleHTTPServer
import logging
```

```
class MyHTTPRequestHandler(SimpleHTTPServer.SimpleHTTPRequestHandler):
    def do_GET(self):
        self.cookieHeader = self.headers.get('Cookie')
        SimpleHTTPServer.SimpleHTTPRequestHandler.do_GET(self)

    def end_headers(self):
        self.send_my_headers()
        SimpleHTTPServer.SimpleHTTPRequestHandler.end_headers(self)

    def send_my_headers(self):
        self.send_header('Set-Cookie', 'A=LDJDSFLKSDJLDSF')

if __name__ == '__main__':
    SimpleHTTPServer.test(HandlerClass=MyHTTPRequestHandler)


Add below host entry to `/etc/hosts`

127.0.0.1 test.com
127.0.0.1 1.test.com
127.0.0.1 footest.com


Sample script to demonstrate requests behavior change

import requests

with requests.Session() as s:
    cookies = dict(cookies_are='working')
    m = s.get("http://test.com:8000", cookies=cookies)
    print(m.request.headers)
    m = s.get("http://1.test.com:8000", cookies=cookies)
    print(m.request.headers)
    m = s.get("http://footest.com:8000", cookies=cookies)
    print(m.request.headers)


Before patch :


{'User-Agent': 'python-requests/2.11.1', 'Accept-Encoding': 'gzip, deflate',
'Accept': '*/*', 'Connection': 'keep-alive', 'Cookie': 'cookies_are=working'}
{'User-Agent': 'python-requests/2.11.1', 'Accept-Encoding': 'gzip, deflate',
'Accept': '*/*', 'Connection': 'keep-alive', 'Cookie': 'A=LDJDSFLKSDJLDSF;
cookies_are=working'}
{'User-Agent': 'python-requests/2.11.1', 'Accept-Encoding': 'gzip, deflate',
'Accept': '*/*', 'Connection': 'keep-alive', 'Cookie': 'A=LDJDSFLKSDJLDSF;
cookies_are=working'}

After patch :


{'User-Agent': 'python-requests/2.11.1', 'Accept-Encoding': 'gzip, deflate',
'Accept': '*/*', 'Connection': 'keep-alive', 'Cookie': 'cookies_are=working'}
{'User-Agent': 'python-requests/2.11.1', 'Accept-Encoding': 'gzip, deflate',
'Accept': '*/*', 'Connection': 'keep-alive', 'Cookie': 'A=LDJDSFLKSDJLDSF;
```

```
cookies_are=working'}
{'User-Agent': 'python-requests/2.11.1', 'Accept-Encoding': 'gzip, deflate',
'Accept': '*/*', 'Connection': 'keep-alive', 'Cookie': 'cookies_are=working'}
```

As with my patch since the cookie is set on `test.com` while making a request to `footest.com` the cookie is skipped as part of the patch since footest is not a subdomain of test.com but 1.test.com is a subdomain. This is a behavior change to be decided whether worth doing or to document this since in a client with session like requests module connecting to lot of hosts this can potentially pass cookies of test.com to footest.com. A discussion on requests repo on providing the option for user to set a stricter cookie policy :
https://github.com/requests/requests/issues/2576

On testing with curl cookie-jar it seems that the cookies are passed even for the subdomain only when it's set and not as part of top level domain.

msg332576 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2018-12-27 06:58**

Also looking at the docs for different frameworks like [Flask](**http://flask.pocoo.org/docs/1.0/api/#flask.Response.set_cookie**) and [Django](**https://docs.djangoproject.com/en/2.1/ref/request-response/#django.http.HttpResponse.set_cookie**) they recommend setting Domain attribute only for cross-domain cookies.

From Django docs

> Use domain if you want to set a cross-domain cookie. For example, domain="example.com" will set a cookie that is readable by the domains **www.example.com**, blog.example.com, etc. Otherwise, a cookie will only be readable by the domain that set it.

When there is no domain specified then the frameworks seem to set the cookie only for the host only as per [RFC 6265](**https://tools.ietf.org/html/rfc6265#section-5.3**). So domain attribute is set all the time and it's just that if the domain attribute is set explicitly by the server with the set_cookie function in the frameworks then the cookiejar has domain_specified set along with dot prefixed for the domain enabling stricter validations. I don't know about the metrics of setting the domain attribute vs not setting it. Checking with a simple Flask app and set_cookie without domain parameter the cookies are passed to suffix domains. With domain passed to set_cookie has dot prefixed and the cookies are not passed to suffix domains.

I also looked into other implementations

* aiohttp - uses cookiejar but has custom domain checks and update cookie methods at **https://github.com/aio-libs/aiohttp/blob/49261c192ff225372dffb39056c3c311714b12c5/aiohttp/cookiejar.py#L141** . Thus it's not affected when tested.
* golang implementation - **https://github.com/golang/go/blob/50bd1c4d4eb4fac8ddeb5f063c099daccfb71b26/src/net/http/cookiejar/jar.go#L123**

msg332583 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2018-12-27 10:21**

I have come across another behavior change between path checks while using the cookie jar implementation available in Python. This is related to incorrect cookie validation but with respect to path so let me know if this needs a separate ticket. I observed the following difference :

1. Make a request to "/" that sets a cookie with "path=/any"
2. Make a request to "/any" and the set cookie is passed since the path matches
3. Make a request to "/anybad" and cookie with "path=/any" is also passed too.

On using golang stdlib implementation of cookiejar the cookie "path=/any" is not passed when a request to "/anybad" is made. So I checked with RFC 6265 where the path match check is defined in section-5.1.4 . RFC 6265 also obsoletes RFC 2965 upon which cookiejar is based I hope since original implementation of cookiejar is from 2004 and RFC 6265 was standardized later. So I think it's good to enforce RFC 6265 since RFC 2965 is obsolete at least in Python 3.8 unless this is considered as a security issue. I think this is a security issue. The current implementation can potentially cause cookies to be sent to incorrect paths in the same domain that share the same prefix. This is a behavior change with more strict checks but I could see no tests failing with RFC 6265 implementation too.

RFC 2965 also gives a loose definition of path-match without mentioning about / check in the paths based on which Python implementation is based as a simple prefix match.

> For two strings that represent paths, P1 and P2, P1 path-matches P2
> if P2 is a prefix of P1 (including the case where P1 and P2 string-
> compare equal). Thus, the string /tec/waldo path-matches /tec.

RFC 6265 path-match definition : **https://tools.ietf.org/html/rfc6265#section-5.1.4**

   A request-path path-matches a given cookie-path if at least one of
   the following conditions holds:

   o The cookie-path and the request-path are identical.

   o The cookie-path is a prefix of the request-path, and the last
      character of the cookie-path is %x2F ("/").

   o The cookie-path is a prefix of the request-path, and the first
      character of the request-path that is not included in the cookie-
      path is a %x2F ("/") character.

The current implementation in cookiejar is as below :

```
def path_return_ok(self, path, request):
    _debug("- checking cookie path=%s", path)
    req_path = request_path(request)
    if not req_path.startswith(path):
        _debug("  %s does not path-match %s", req_path, path)
        return False
    return True
```

Translating the RFC 6265 steps (a literal translation of go implementation) would have something like below and no tests fail on master. So the python implementation goes in line with the RFC not passing cookies of "path=/any" to /anybody

```
def path_return_ok(self, path, request):
    req_path = request_path(request)
    if req_path == path:
        return True
    elif req_path.startswith(path) and ((path.endswith("/") or req_path[len(path)]
== "/")):
        return True

    return False
```

The golang implementation is as below which is a literal translation of RFC 6265 steps at
https://github.com/golang/go/blob/50bd1c4d4eb4fac8ddeb5f063c099daccfb71b26/src/net/http/cookiejar/jar.go#L130

```
// pathMatch implements "path-match" according to RFC 6265 section 5.1.4.
func (e *entry) pathMatch(requestPath string) bool {
        if requestPath == e.Path {
                return true
        }
        if strings.HasPrefix(requestPath, e.Path) {
                if e.Path[len(e.Path)-1] == '/' {
                        return true // The "/any/" matches "/any/path" case.
                } else if requestPath[len(e.Path)] == '/' {
                        return true // The "/any" matches "/any/path" case.
                }
        }
        return false
}
```

Feel free to correct me if I am wrong on the above.

msg332920 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2019-01-03 08:01**

I have opened ~~**issue35647**~~ for path related checks as a separate report.

msg335386 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2019-02-13 06:30**

This issue affects 2.7 as well along with 3.4 and 3.5. The initial report was notified to **security@python.org** . 2.7.16 release candidate dates were announced at **https://mail.python.org/pipermail/python-dev/2019-February/156266.html**. I have prepared an initial backport of this with tests for 2.7 at **https://github.com/python/cpython/compare/2.7...tirkarthi:bpo35121-27** . Serhiy has approved the PR for master. I have added notes here and on the PR about the issue and implementation in other languages. It would be helpful if someone can double check my analysis since cookiejar has not received much change over the years.

If this is a potential candidate for 2.7 release I can help with that once the changes are merged to master. Adding Benjamin Peterson to this issue to take a call on if it needs to be backported to 2.7. If it's planned for a backport then also to decide on priority if this needs to be part of 2.7.16 or later release.

msg337588 - (view) **Author: Ned Deily (ned.deily) *** **Date: 2019-03-10 02:09**

New changeset **ca7fe5063593958e5efdf90f068582837f07bd14** by Ned Deily (Xtreak) in branch 'master':
~~**bpo-35121**~~: prefix dot in domain for proper subdomain validation (~~**GH-10258**~~)
https://github.com/python/cpython/commit/ca7fe5063593958e5efdf90f068582837f07bd14

msg337590 - (view) **Author: Ned Deily (ned.deily) *** **Date: 2019-03-10 02:58**

New changeset **e5123d81ffb3be35a1b2767d6ced1a097aaf77be** by Ned Deily (Miss Islington (bot)) in branch '3.7':
~~**bpo-35121**~~: prefix dot in domain for proper subdomain validation (~~**GH-10258**~~) (~~**GH-12261**~~)
https://github.com/python/cpython/commit/e5123d81ffb3be35a1b2767d6ced1a097aaf77be

msg337592 - (view) **Author: Ned Deily (ned.deily) *** **Date: 2019-03-10 02:59**

New changeset **b241af861b37e20ad30533bc0b7e2e5491cc470f** by Ned Deily (Miss Islington (bot)) in branch '3.6':
~~**bpo-35121**~~: prefix dot in domain for proper subdomain validation (~~**GH-10258**~~) (~~**GH-12260**~~)
https://github.com/python/cpython/commit/b241af861b37e20ad30533bc0b7e2e5491cc470f

msg337593 - (view) **Author: Ned Deily (ned.deily) *** **Date: 2019-03-10 03:06**

OK, thanks for the initial report and a big thank you to @xtreak for the PR and following up on things. The PR is now merged to master for 3.8.0 and backported as a security fix for release in 3.7.3 and 3.6.9. Reassigning to Benjamin for a decision on backporting to 2.7.

msg337598 - (view) **Author: Serhiy Storchaka (serhiy.storchaka) *** **Date: 2019-03-10 08:21**

What about 3.4 and 3.5?

msg337600 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2019-03-10 08:51**

From my initial tests 3.4 and 3.5 were also affected. 3.4 is going EoL and RC1 is out but there is one another security issue (~~**issue36216**~~) fixed last week with a PR open. If the merge window is open and Larry is okay then I can raise backport PRs if needed. There are less changes made to cookiejar and cherry_picker would also work fine as I tried it locally.

cherry_picker --no-push **ca7fe5063593958e5efdf90f068582837f07bd14** 3.5
🐍 🍒 ⛏

Now backporting '**ca7fe5063593958e5efdf90f068582837f07bd14**' into '3.5'
Switched to a new branch 'backport-ca7fe50-3.5'
Branch 'backport-ca7fe50-3.5' set up to track remote branch '3.5' from 'upstream'.

[backport-ca7fe50-3.5 **fcb2dd85a0**] ~~**bpo-35121**~~: prefix dot in domain for proper subdomain validation (~~**GH-10258**~~)
 Author: Xtreak <**tir.karthi@gmail.com**>
 Date: Sun Mar 10 07:39:48 2019 +0530
 3 files changed, 45 insertions(+), 2 deletions(-)
 create mode 100644 **bpo-35121**.EgHv9k.rst">Misc/NEWS.d/next/Security/2018-10-31-15-39-17.~~**bpo-35121**~~.EgHv9k.rst

Finished cherry-pick **ca7fe5063593958e5efdf90f068582837f07bd14** into backport-ca7fe50-3.5 😀

cherry_picker --no-push **ca7fe5063593958e5efdf90f068582837f07bd14** 3.4
🐍 🍒 ⛏

Now backporting '**ca7fe5063593958e5efdf90f068582837f07bd14**' into '3.4'
Switched to a new branch 'backport-ca7fe50-3.4'
Branch 'backport-ca7fe50-3.4' set up to track remote branch '3.4' from 'upstream'.

Performing inexact rename detection: 100% (639108/639108), done.
[backport-ca7fe50-3.4 **46ea57d6b3**] ~~**bpo-35121**~~: prefix dot in domain for proper subdomain validation (~~**GH-10258**~~)
 Author: Xtreak <**tir.karthi@gmail.com**>
 Date: Sun Mar 10 07:39:48 2019 +0530
 3 files changed, 45 insertions(+), 2 deletions(-)
 create mode 100644 **bpo-35121**.EgHv9k.rst">Misc/NEWS.d/next/Security/2018-10-31-15-39-17.~~**bpo-35121**~~.EgHv9k.rst

Finished cherry-pick **ca7fe5063593958e5efdf90f068582837f07bd14** into backport-ca7fe50-3.4 😀

msg337601 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2019-03-10 09:16**

There are many libraries that use DefaultCookiePolicy and requests library uses it for client where session state needs to be maintained across different requests. Currently, requests doesn't have a documented API to change to cookiejar policy and were not keen on introducing a custom one since this might introduce maintenance burden over keeping it in sync with other changes when made upstream. The team have been informed about this when the issue was created and I also updated the maintainers now about the fix being merged since it's a highly popular library.

So requests will remain affected when ran on versions where this patch is not available in CPython standard library as of now. A potentially simple workaround in the absence of patch on affected versions is to set DomainStrict in the cookiepolicy that would make sure a literal match against domain is made at [0] . The disadvantage I guess would be that cookie set on example.com would not be shared with subdomain which might break workflow. aio-http was not affected since it uses custom cookiejar policy. scrapy also seems to be not affected since they custom policies. Most of the web frameworks don't recommend setting domain explicitly and set them implicitly so it can be reproduced in the default setup of frameworks and Flask was the one I tested which makes me assume this could be easily exploited.

[0] **https://github.com/python/cpython/blob/ca7fe5063593958e5efdf90f068582837f07bd14/Lib/http/cookiejar.py#L1158**

msg338109 - (view) **Author: Larry Hastings (larry) *** **Date: 2019-03-16 22:56**

New changeset **42ad4101d3ba7ca3c371dadf0f8880764c9f15fb** by larryhastings (Xtreak) in branch '3.4':

[3.4] ~~bpo-35121~~: prefix dot in domain for proper subdomain validation (~~GH-10258~~) (~~#12279~~)
https://github.com/python/cpython/commit/42ad4101d3ba7ca3c371dadf0f8880764c9f15fb

msg338114 - (view) **Author: Larry Hastings (larry) *** **Date: 2019-03-17 00:03**

New changeset 4749f1b69000259e23b4cc6f63c542a9bdc62f1b by larryhastings (Xtreak) in branch '3.5':
[3.5] ~~bpo-35121~~: prefix dot in domain for proper subdomain validation (~~GH-10258~~) (~~#12281~~)
https://github.com/python/cpython/commit/4749f1b69000259e23b4cc6f63c542a9bdc62f1b

msg338152 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2019-03-18 01:22**

Larry, I am reopening this since this seems to affects 2.7 and would wait for Benjamin's call on backporting this.

msg344555 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2019-06-04 12:30**

I added this issue to my security website:
https://python-security.readthedocs.io/vuln/cookie-domain-check.html

So it's fixed in Python 3.4.10, 3.5.7 and 3.7.3. Right now, 2.7 and 3.6 are vulnerable (but 3.6 branch is fixed).

msg344556 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2019-06-04 12:37**

Can someone try to backport the fix to Python 2.7?

msg344560 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2019-06-04 12:49**

> Can someone try to backport the fix to Python 2.7?

The backport to 2.7 ~~PR 13426~~ is open. It would be helpful if someone can review it. I am not sure of the commit review process and who needs to review and approve it since this is assigned to Benjamin Peterson. It's a fairly straightforward backport except that http.cookiejar is cookielib in Python 2.

msg345689 - (view) **Author: miss-islington (miss-islington)** **Date: 2019-06-15 15:29**

New changeset 979daae300916adb399ab5b51410b6ebd0888f13 by Miss Islington (bot) (Xtreak) in branch '2.7':
[2.7] ~~bpo-35121~~: prefix dot in domain for proper subdomain validation (~~GH-10258~~) (~~GH-13426~~)
https://github.com/python/cpython/commit/979daae300916adb399ab5b51410b6ebd0888f13

msg345700 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2019-06-15 16:38**

Closing this as resolved since the fix was merged to all branches. Thank you all.

msg345736 - (view) **Author: STINNER Victor (vstinner) *** **Date: 2019-06-16 07:58**

Again, well done Karthikeyan Singaravelan!

msg346748 - (view) **Author: Riccardo Schirone (rschiron)** **Date: 2019-06-27 15:49**

Did anybody request a CVE for this issue? I think it deserves one as it is a security issue and it may leak cookies to wrong domains. Does anybody have anything against assigning a CVE to this issue? If not, I would try to get one from MITRE.

msg346749 - (view) **Author: Karthikeyan Singaravelan (xtreak) *** **Date: 2019-06-27 15:55**

I also reported it to **security@python.org** . Please check with them too to see if there is a CVE request already made. Thanks.

msg347951 - (view) **Author: Riccardo Schirone (rschiron)** **Date: 2019-07-15 08:30**

CVE-2018-20852 has been assigned to this flaw.

History

Date	User	Action	Args
2022-04-11 14:59:07	admin	set	github: 79302
2019-07-15 09:42:01	vstinner	set	title: [CVE-2018-20852] Cookie domain check returns incorrect results -> [CVE-2018-20852] Cookie domain check returns incorrect results
2019-07-15 09:41:35	vstinner	set	title: Cookie domain check returns incorrect results -> [CVE-2018-20852] Cookie domain check returns incorrect results
2019-07-15 08:30:53	rschiron	set	messages: + **msg347951**
2019-06-27 15:55:29	xtreak	set	messages: + **msg346749**
2019-06-27 15:49:06	rschiron	set	nosy: + **rschiron** messages: + **msg346748**
2019-06-16 07:58:28	vstinner	set	messages: + **msg345736**
2019-06-15 16:38:00	xtreak	set	status: open -> closed resolution: fixed messages: + **msg345700** stage: patch review -> resolved
2019-06-15 15:29:46	miss-islington	set	nosy: + **miss-islington** messages: + **msg345689**
2019-06-04 12:49:06	xtreak	set	messages: + **msg344560**
2019-06-04 12:37:47	vstinner	set	messages: + **msg344556**
2019-06-04 12:30:18	vstinner	set	nosy: + **vstinner** messages: + **msg344555**
2019-05-19 19:09:00	xtreak	set	stage: commit review -> patch review pull_requests: + **pull_request13336**
2019-05-10 17:57:59	ned.deily	set	messages: - **msg342111**
2019-05-10 17:36:40	ned.deily	set	messages: + **msg342111**
2019-03-18 01:22:13	xtreak	set	status: closed -> open resolution: fixed -> (no value) messages: + **msg338152** stage: resolved -> commit review

2019-03-17 20:52:04	larry	set	status: open -> closed resolution: fixed stage: patch review -> resolved
2019-03-17 00:03:41	larry	set	messages: + **msg338114**
2019-03-16 22:56:36	larry	set	messages: + **msg338109**
2019-03-11 16:02:58	xtreak	set	pull_requests: + **pull_request12261**
2019-03-11 15:54:07	xtreak	set	stage: commit review -> patch review pull_requests: + **pull_request12259**
2019-03-10 09:16:39	xtreak	set	messages: + **msg337601**
2019-03-10 08:51:02	xtreak	set	messages: + **msg337600**
2019-03-10 08:21:40	serhiy.storchaka	set	nosy: + **larry** messages: + **msg337598** versions: + Python 3.4, Python 3.5
2019-03-10 03:06:58	ned.deily	set	assignee: **benjamin.peterson** stage: patch review -> commit review messages: + **msg337593** versions: + Python 2.7
2019-03-10 03:01:07	ned.deily	set	pull_requests: - **pull_request12244**
2019-03-10 02:59:30	ned.deily	set	messages: + **msg337592**
2019-03-10 02:58:28	ned.deily	set	messages: + **msg337590**
2019-03-10 02:36:45	miss-islington	set	pull_requests: + **pull_request12246**
2019-03-10 02:10:28	miss-islington	set	pull_requests: + **pull_request12245**
2019-03-10 02:10:14	miss-islington	set	pull_requests: + **pull_request12244**
2019-03-10 02:09:53	ned.deily	set	nosy: + **lukasz.langa** messages: + **msg337588**
2019-02-13 06:30:08	xtreak	set	nosy: + **benjamin.peterson** messages: + **msg335386**
2019-01-03 08:01:23	xtreak	set	messages: + **msg332920**
2018-12-27 10:21:23	xtreak	set	messages: + **msg332583**
2018-12-27 06:58:31	xtreak	set	messages: + **msg332576**
2018-12-24 09:59:22	serhiy.storchaka	set	priority: high -> release blocker nosy: + **ned.deily** type: behavior -> security
2018-12-24 03:44:50	ned.deily	set	nosy: + **serhiy.storchaka**
2018-12-24 03:42:28	ned.deily	set	keywords: + **security_issue** priority: normal -> high
2018-12-21 20:57:07	xtreak	set	messages: + **msg332299**

2018-11-03 04:02:30	xtreak	set	messages: + **msg329179**
2018-11-03 02:12:48	Windson Yang	set	nosy: + **Windson Yang** messages: + **msg329176**
2018-10-31 12:36:05	xtreak	set	nosy: + **orsenthil**, **martin.panter**
2018-10-31 11:27:17	xtreak	set	messages: + **msg328985** versions: + Python 3.8
2018-10-31 10:29:16	xtreak	set	keywords: + **patch** stage: patch review pull_requests: + **pull_request9569**
2018-10-31 10:24:45	西田雄治	set	messages: + **msg328981**
2018-10-31 08:15:02	xtreak	set	nosy: + **xtreak** messages: + **msg328975**
2018-10-31 06:52:48	西田雄治	create	