Security

PHP injection vulnerability - no check on text input format.

Sun, 2013-07-28 14:36 — wodenx

Project:	RESTful Web Services
Component:	Code
Category:	bug report
Priority:	Moderately critical
Assigned:	klausi
Status:	Closed (fixed)
View grants:	fago, sepgil

Description

A service client authenticated as a particular user may submit an entity with a formatted text field containing a format (e.g. php_code) which that user doesn't have permission to use. Attached patch (with test) demonstrates the problem and a proposed solution.

*** NOTE - This issue also affects the services_entity project (https://drupal.org/project/services_entity), and a second patch is attached to apply the same solution. Services Entity doesn't have working tests, so no test is supplied with that patch.

It's arguable that this check should be moved directly into entity_api, although I'm not sure how that would work since the access checks don't conventionally receive the data to be set.

I suspect that this vulnerability may exist also in D8 Rest services, but I have not had a chance to investigate that fully.

Attachment	Size
restws-text-format-access.patch REVIEW SIMPLYTEST.ME	4.07 KB
services_entity-text-format-access.patch REVIEW SIMPLYTEST.ME	1.93 KB

Contrib XSS, CSRF, or SQL Injection

Comments

#1

Sun, 2013-07-28 14:54 — wodenx

Patch for services entity revised to apply to the ServicesEntityResourceControllerClean class as well.

Attachment	Size
services_entity-text-format-access-2.patch REVIEW SIMPLYTEST.ME	2.79 KB

reply

#2

Sun, 2013-07-28 21:33 — klausi

Assigned to:	Anonymous	» klausi
Status:	Needs team response	» Needs maintainer response
View grants:		+fago, +sepgil

Thanks for spotting this and the test case!

Adding the other RESTWS maintainers.

The patch looks mostly good, I'm going to fix some coding standard violations and I want to investigate if we can avoid looping twice over all entity fields on update/PUT requests.

@fago: do you know any other way to check for filter format access with the entity API?

Regarding services_entity: that module has no stable release yet, so according to our policy https://drupal.org/security-advisory-policy we are not fixing this privately for that module. Please keep this confidential until we manage to push out new RESTWS releases, then we can open a critical bug report in the public services_entity issue queue.

edit reply

#3

Sun, 2013-07-28 22:12 — Damien Tournoud

Ew. Isn't that totally the job of the field validation function to ensure that type of things? I would argue that this is a core bug. We don't want crazy special cases popping up everywhere.

reply

#4

Mon, 2013-07-29 00:52 — Dave Reid

Field validation should run for programmatic (non-UI) entity saves, but it depends on the caller invoking field_attach_validate() *prior* to calling the entity save function.

reply

#5

Mon, 2013-07-29 00:54 — Dave Reid

That being said, text_field_validate() does not contain any check for 'does the user have access to this format'. From my experience most field validation callbacks don't check stuff like that because it's not what core does. I'm not sure where the responsibility would lie here, but probably entity API in D7.

reply

#6

Mon, 2013-07-29 01:35 — Damien Tournoud

I'm of the opinion that if `restws` and `entity_service` properly call field-level validation functions (they probably don't, but let's suppose they do for the sake of argument), it's up to the field types to implement the proper validations. Pushing this validation to the upper layer is just a case of layer violation (one of those that make Drupal as a CMS so annoying to work with).

So there are probably two bugs here:

- Core should implement proper validation checks for text formats
- `restws` and `entity_service` should properly call field-level validation functions

reply

#7

Mon, 2013-07-29 01:35 — wodenx

Will do.
Out of curiosity - what standards violations did you find? I ran it through coder and it didn't squawk.
Also - do you know if this is also a potential issue in D8?
Thanks.

reply

#8

Mon, 2013-07-29 02:07 — wodenx

Just saw the rest of the thread.

Re: the validation arguments - I considered this - but I'm not sure it's the place of a *validator* to check *access*. It's very easy to imagine a case where a module might want to write values it know are safe, even though the currently logged in user isn't authorized to use the format. I would argue that the behavior of a field validator (though not a form validator) should not depend on the state of the current session.

That being said - it would be easy enough to add the format access check to core as you describe along the lines of the attached (untested) patch. If that's deemed the desired approach, I'm happy to work it up along with a test case.

BTW - services_entity and restws do, in fact, call field level validation indirectly via entity_api EntityDrupalWrapper::set() method.

Attachment	Size
93428-field-format-access-untested.patch REVIEW SIMPLYTEST.ME	1.22 KB

reply

#9

Mon, 2013-07-29 03:49 — wodenx

whoops - patch above has some stupid errors.

Attachment	Size
93428-field-format-access-untested-9.patch REVIEW SIMPLYTEST.ME	1.24 KB

reply

#10

Mon, 2013-07-29 10:32 — fago

>restws and entity_service should properly call field-level validation functions
yep, I agree that they should. However I'm not sure I agree in doing access checks as apart of the field validation - that is intended to validate the data only, not the form submission of a current user.

Imho, it would make sense to check that as part of field access - (if it isn't already?). Then, before saving we can do access checks besides validation?

reply

#11

Mon, 2013-07-29 10:37 — Damien Tournoud

@fago: field access is part of the Entity API module right? Do we have implementations for core fields in there?

reply

#12

Mon, 2013-07-29 13:45 — wodenx

@fago As I said in #1, I agree that it might be better to do this on the field access level in entity, and that's how I originally approached the solution. But as I also said, the issue is that access callbacks don't typically get passed the data to be set, but rather the entity on which they are being set, so this involves a change in usage patterns that could be far reaching. All the same, here's the work I did on that approach, including a test case. Note that this crosses paths with https://drupal.org/node/1780646 and will have to be reconciled.

*** EDIT This patch is incomplete .. full patch in next post.

Attachment	Size
93428-12-field-format-access-entity.patch REVIEW SIMPLYTEST.ME	3.69 KB

reply

#13

Mon, 2013-07-29 13:49 — wodenx

Here is the full patch.

Attachment	Size
93428-13-field-format-access-entity.patch REVIEW SIMPLYTEST.ME	4.82 KB

reply

#14

Mon, 2013-07-29 14:25 — Damien Tournoud

Project:	RESTful Web Services	» Entity API
Status:	Needs maintainer response	» Patch (code needs review)

Not a review of the actual patch, but this approach sounds way more reasonable to me. Reassigning to the Entity API module.

reply

#15

Tue, 2013-07-30 10:33 — klausi

Project:	Entity API	» RESTful Web Services

RESTWS does not call field level access functions on write operations right now, which we should fix. The entity API patch looks interesting, some questions:

If we call ->access() on the field before setting the data this will not work. An existing text format could be set that the current user has access to, so ->access() returns TRUE and then we can again set php_code as unauthorized user.

If we call ->access() on the field after setting the value this could also be a problem, since it could be set to php_code before, then be changed to filtered_html and ->access() returns TRUE. But as far as I remember text fields that are set to a higher level text format cannot be edited and are locked for unprivileged users.

So we could call ->access() twice for the field, once before setting the new value(s) and once after that. If either one returns FALSE we deny access. Does that make sense?

Anyway, since this is a private security discussion we should fix the actual vulnerability with a less optimal solution in one module (RESTWS) and then continue with entity API improvements in public. That guarantees us to react in a timely fashion without interdependent security releases (would the entity API fix be a security release?).

Proposed solution for RESTWS:
* Implement ->access() on fields before setting incoming values
* Implement ->access() on fields after setting incoming values
* Call field_attach_validate() before saving data.
* Implement the hack to cover text format access in RESTWS, as we had it in the first patches.
* Write test cases for those 4 points
* Provide patches for RESTWS 1.x and 2.x

What do you think?

edit reply

#16

Tue, 2013-07-30 16:54 — wodenx

@klausi that all looks right to me.

One thought regarding the double ->access() check - i wonder if it's actually necessary to prevent a user from "lowering" the format -- even if the field ui does lock an input field if a user isn't authorized to use its format, do we have to do the same? To me, lack of the "use text format x" permission doesn't imply that i can't change a field which already has format x to another format which i do have permission to use, as long as i have access to modify that field in the first place.

In the field UI case, it makes sense bc the user is *editing* the text in a form -- so there has to be an editable representation of the field BEFORE it's updated. But for RESTws (and services_entity), this isn't the case - the service client is providing completely new text and format -- we never have to represent the existing text with a different format than intended.

reply

#17

Wed, 2013-07-31 11:01 — klausi

Makes sense. I find it confusing that we call field access AFTER we set a values - I would have expected to call it before setting values on the field.

In Drupal 8 I implemented hook_entity_field_access() for a REST test case with the assumption that it will be called before setting values (entity_test.module). So we need to change that there, too (after fixing this).

But yeah, the ->access() method should probably deal with the new values, it can always use entity_load_unchanged() to get the old field values.

Proposed solution for RESTWS:
* Implement ->access() on fields after setting incoming values
* Call field_attach_validate() before saving data.
* Implement the hack to cover text format access in RESTWS, as we had it in the first patches.
* Write test cases for those 3 points
* Provide patches for RESTWS 1.x and 2.x

edit reply

#18

Wed, 2013-07-31 13:47 — klausi

I tried to implement field access for creation, but unfortunately the access callback entity_metadata_no_hook_node_access() for nodes is severely broken. See https://drupal.org/node/1780646#comment-7555461

So I tried to simply override that access callback, but Entity API won't let me since its entity info alter hook always runs last :-(

Any ideas how to work around that?

Attachment	Size
restws-field-access-93428.patch REVIEW SIMPLYTEST.ME	8.08 KB

edit reply

#19

Wed, 2013-07-31 14:37 — wodenx

Ha I just ran into the same issue. We can try to pre-empt entity's hook_module_implements_alter since we're heavier by default -- seeing fi that will work now.
I also ran into some other issues with calling field level validation on every property -- new patch will follow shortly.

reply

#20

Wed, 2013-07-31 16:15 — wodenx

OK - this combines your patch and mine.
- Uses hook_module_implements_alter() to make our entity_info_alter() implementation run after entity_api.
- Doesn't check access on entity keys for the create operation (comment inline explains why)
- Has a hack to allow our node_access callback to work properly on 'update' operations when there's no nid. This is necessary bc entity_api doesn't distinguish between 'create' and 'update' access when checking entity_access for the parent of a field. It should properly be fixed there, along the lines:

```
diff --git a/includes/entity.wrapper.inc b/includes/entity.wrapper.inc
index 4ee1c3a..f49a558 100644
--- a/includes/entity.wrapper.inc
+++ b/includes/entity.wrapper.inc
@@ -512,8 +512,11 @@ class EntityStructureWrapper extends EntityMetadataWrapper implements IteratorAg
       while (!($entity instanceof EntityDrupalWrapper) && isset($entity->info['parent'])) {
         $entity = $entity->info['parent'];
       }
-      if ($entity instanceof EntityDrupalWrapper && !$entity->entityAccess('update', $account)) {
-        return FALSE;
+      if ($entity instanceof EntityDrupalWrapper) {
+        $entity_op = $entity->getIdentifier() ? 'update' : 'create';
+        if (!$entity->entityAccess($entity_op, $account)) {
+          return FALSE;
+        }
       }
```

```
    }
    if (!empty($info['access callback'])) {
```

Or, even better, entity_api should add a 'create' op to field level access.

Attachment	Size
93428-20-restws-field-access.patch **REVIEW** **SIMPLYTEST.ME**	10.08 KB

reply

#21

Wed, 2013-07-31 22:49 — klausi

Status:	Patch (code needs review)	» Patch (code needs work)

```
+++ b/restws.entity.inc
@@ -156,6 +156,20 @@ class RestWSEntityResourceController implements RestWSQueryResourceControllerInt
       $wrapper = entity_property_values_create_entity($this->entityType, $values);
+      $info = entity_get_info($this->entityType);
```

Can't we get the entity info from the wrapper?

```
+++ b/restws.entity.inc
@@ -156,6 +156,20 @@ class RestWSEntityResourceController implements RestWSQueryResourceControllerInt
+        if (!in_array($name, $entity_keys)) {
+          // @todo Replace this with simple $property->access() once entity_api supports text format access checks.
```

Comment exceeds 80 characters and "entity_api" should be "Entity API".

```
+++ b/restws.entity.inc
@@ -357,4 +375,34 @@ class RestWSEntityResourceController implements RestWSQueryResourceControllerInt
+  /**
+   * Helper method to check access on a property.
+   * @todo Remove this once entity_api properly handles text-format access.
+   *
+   * @param $wrapper
+   *   The wrapped entity whose fields are to be checked.
+   * @param $name
+   *   The name of the property to set.
+   * @param value
+   *   The value to set.
+   * @throws RestWSException (403).
+   *   If there is an attempt to set an unauthorized property.
+   */
```

data types are missing, @return is missing, short description should be its own paragraph.

```
+++ b/restws.module
@@ -433,7 +435,8 @@ function restws_permission() {
-   if ($hook == 'menu_alter') {
+   // @todo remove entity_info_alter once https://drupal.org/node/1780646 is fixed. ¶
```

Trailing white space. And this will only work randomly, because "restws" comes after "entity" in the alphabet, but if the module weights are different we will fail to register our access callback. Still looks like this is the best we can do :-(

```
+++ b/restws.test
@@ -99,6 +99,91 @@ class RestWSTestCase extends DrupalWebTestCase {
+    $json = drupal_json_encode($new_node);
+    $result = $this->httpRequest('node', 'POST', $account, $json);
+    $node = entity_load('node', FALSE, array('title' => $title));
+    $this->assertEqual(count($node), 0, "Node with unauthorized input format wasn't created");
+
+    $this->assertResponse('403');
```

The response code is the first thing we should check after the request, so this should be moved up.

```
+++ b/restws.test
@@ -99,6 +99,91 @@ class RestWSTestCase extends DrupalWebTestCase {
+
+    $this->checkPermissions(array(), TRUE); // Reset the cache of valid permissions.
+    $account = $this->drupalCreateUser(array('access content', 'bypass node access', 'access resource node', 'administer
users', 'use text format php_code'));
```

Comments should be on their own line.

Looks like this is the best we can do on this issue, any other comments on the overall approach?

TODO:
* Call field_attach_validate() before saving data + test case.

edit reply

#22

Thu, 2013-08-01 00:36 — wodenx

> Can't we get the entity info from the wrapper?

yes, but what's the difference? The info is statically cached anyway. But I made this change.

> Comment exceeds 80 characters

It is on a single line bc it's a @todo, and breaking it up would orphan part of the task (at least in eclipse)

> And this will only work randomly, because "restws" comes after "entity" in the alphabet

yep - but hopefully the linked issue will land soon and we can get rid of this hack. The alternative would be to wait to deploy the field access part until after the Entity API thing is fixed. What do you think?

> TODO:
> * Call field_attach_validate() before saving data + test case.

Is this really necessary? EntityMetadataWrapper::set() calls $this->validate() - I'd hope that invokes the field api validators...

I think I addressed the other issues you spotted. Thanks.

Attachment	Size
93428-22-restws-field-access.patch REVIEW SIMPLYTEST.ME	10 KB

reply

#23

Thu, 2013-08-01 10:35 — klausi

Patch looks good!

No, the ->validate() method does not call any field validators, it just verifies the data type of primitives and executes a validation callback. And that is not generically set for fields.

Anyway, I think we are dealing with enough changes already for this security fix, so let's ignore field_attach_validate() for this patch unless someone can describe a concrete vulnerability regarding that.

TODO:
* testResourceArray() test case is broken and needs to be fixed
* Update/PUT test case for the field access test and the input format test
* Patch for RESTWS 1.x
* Security Advisory draft at https://security.drupal.org/node/add/advisory

edit reply

#24

Fri, 2013-08-02 17:04 — klausi

Status:	Patch (code needs work)	» Patch (code needs review)

Working my way through the test cases as this uncovers a couple of things.

* Added fix for the node author, so that unprivileged users can still set themselves as author, so checkPropertyAccess() is now even a bit more customized.

Not finished with all test cases yet, just posting an intermediate patch.

Attachment	Size
restws-field-access-93428-24.patch REVIEW SIMPLYTEST.ME	17.18 KB
restws-field-access-93428-24-interdiff.txt REVIEW	12.33 KB

edit reply

#25

Fri, 2013-08-02 17:34 — wodenx

That looks right, though it's too bad we need all the special cases. Entity API makes some rather unilateral decisions about what permissions are needed to modify various node properties.

I see what you mean about field_attach_validate. It looks like Field doesn't provide a clean mechanism for validating an individual field - just all the fields on an entity. I wonder if that can/should be refactored?

Regarding the access - looking into this more closely, I think the proper place for the access check is not in Entity API, but in Text. Entity APIalready invokes field_access, so it's really the responsibility of the text module to implement hook_field_access and check the format. I'm working on that patch now, but it should render the one in #13 unnecessary.

reply

#26

Fri, 2013-08-02 22:31 — klausi

Finished with all the test cases, added tests for PUT requests for field access and the PHP format. I also removed the HTTP request verbosity for better debugging and directly committed that non-security related part to 7.x-2.x.

This patch should now be ready for 7.x-2.x.

I'm planning to release this next Wednesday, any reviews would be appreciated.

TODO:
* Patch for RESTWS 1.x
* Security Advisory draft at https://security.drupal.org/node/add/advisory

Attachment	Size
restws-field-access-93428-26.patch REVIEW SIMPLYTEST.ME	15.79 KB
restws-field-access-93428-26-interdiff.txt REVIEW	10.09 KB

edit reply

#27

Sun, 2013-08-04 14:51 — wodenx

Attached core patch follows up on #25 - posting here for preliminary review - will submit to core issue queue once this security issue is resolved.

Attachment	Size
text-field-format-access.patch REVIEW SIMPLYTEST.ME	3.51 KB

reply

#28

Mon, 2013-08-05 11:45 — klausi

Status:	Patch (code needs review)	» Ready for release

And here is the patch for the RESTWS 1.x branch, merged in the test cases accordingly and verified the functionality against a clone of a production site.

Advisory draft: https://security.drupal.org/node/93798

Attachment	Size
restws-field-access-1.x-93428-28.patch REVIEW SIMPLYTEST.ME	15.48 KB

edit reply

#29

Wed, 2013-08-07 17:32 — klausi

Status:	Ready for release	» Closed (fixed)

This will go out later today at https://drupal.org/node/2059603

Thanks everyone!

edit reply

#1

Issue title: *
PHP injection vulnerability - no check on text input format.

Project: *	**Component:** *	**Assigned:**
RESTful Web Services	Code	klausi

Category: *	**Priority:**	**Status:**
bug report	Moderately critical	Closed (fixed)

Create commit message
Make sticky

Input format

Access control

Attach new file:
Browse... No file selected.
The maximum upload size is *1 MB*. Only files with the following extensions may be uploaded: *jpg jpeg gif png pdf odt ods odp tar gz zip patch diff txt*.

Attach

Save Preview Restore previously entered data