

Stored XSS in blob viewer

[HackerOne report #806571](#) by [yvvdwF](#) on 2020-02-27, assigned to [@jeremymatos](#):

Summary

I found a Stored-XSS in blob viewer when viewing a json file.

In particular, when viewing an openapi file, [openapi_viewer](#) is called to transfer the file's data to [SwaggerUIBundle](#) to render.

SwaggerUIBundle does its job when rendering graphical representation of the openapi's content. It also allows *html tags and attributes* in the description of the openapi. Although it removes malicious tags and attributes, but this is not enough in gitlab's context:

1. `class` and `style` attributes allow attackers to arbitrarily present their disposition. My demo below uses `class` attribute to create a transparent layer that fulfills the document to intercept any user's clicks.
2. `data-*` attributes, under the help of [jQueryUI](#), allows attackers to create any requests to server when user clicking (not only `GET`, but also, `PUT`, `DELETE`, `HEAD`) with arbitrary parameters
3. The current CSP is easily by passed by [QueryGlobalEval](#). In my demo below, you should see an `alert` after clicking anywhere

Steps to reproduce

1. In any project, create a file naming `xss-openapi1.json`, then put the following content:

```
{
  "swagger" : "2.0",
  "info" : {
    "description" : "<a href=https://gitlab.com/yvvdwF/data/-/wikis/alert.md data-type=script style='cursor:default' data-r"
  }
}
```

2. Click anywhere on the document view, you should see an alert.

Impact

There are three impacts as in the Summary above. The most important impact is the stored-XSS allowing attackers to perform any action on behalf of users at the client side.

Examples

(This repository is in private mode, please let me know if you cannot access it)

<https://gitlab.com/yvvdwF/xss/-/blob/master/xss-openapi1.json>

What is the current bug behavior?

Gitlab does not check the result generated by SwaggerUIBundle

What is the expected correct behavior?

Should remove any inappropriate html attributes, such as, `data-*`, `style`, `class`.


Output of checks

This bug happens on GitLab.com

Impact

The stored-XSS allows attackers to perform any action on behalf of users at the client side.

Edited 2 years ago by [Nick Thomas](#)

 Drag your designs here or [click to upload](#).

Tasks  0


No tasks are currently assigned. Use tasks to break down this issue into smaller parts.


Linked items  0

Link issues together to show that they're related or that one is blocking others. [Learn more](#).

Activity

 [GitLab SecurityBot](#) added [HackerOne](#) [security](#) labels 2 years ago

 [GitLab SecurityBot](#) added [priority 3](#) [severity 3](#) scoped labels 2 years ago

 [GitLab SecurityBot](#) @[gitlab-security-bot](#) · 2 years ago Author Reporter

[HackerOne comment](#) by [jmatos_bgTVF](#):


Hello @yvvdwF,

Thank you for this extra report.

I have created a private project containing `xss-openapi1.js` and its content described in Step1. But when I access <https://gitlab.com/thenamespace/project/-/blob/master/xss-openapi1.js>, I cannot have the XSS triggered by clicking on the document.

Could you please provide more information about Step2 ?

Best regards, GitLab Security Team


 [GitLab SecurityBot](#) @[gitlab-security-bot](#) · 2 years ago Author Reporter

[HackerOne comment](#) by [yvvdwF](#):

Hi,

A mistake in filename, it should be ended by `.json`, for example, `xss-openapi1.json` (not `xss-openapi1.js`)


Best regards,

 [GitLab SecurityBot](#) @[gitlab-security-bot](#) · 2 years ago Author Reporter

[HackerOne comment](#) by [jmatos_bgTVF](#):

Thank you for the quick feedback, I can now reproduce it. Before assessing the impact, are you able to trigger it without any user interaction, i.e. without clicking ?


Best regards, GitLab Security Team


 [GitLab SecurityBot](#) @[gitlab-security-bot](#) · 2 years ago Author Reporter

[HackerOne comment](#) by [yvvdwF](#):

are you able to trigger it without any user interaction, i.e. without clicking ?

No, I see no way to auto trigger js executions (because CSP will block them)

 [Jeremy Matos](#) changed due date to May 22, 2020 2 years ago

 [Jeremy Matos](#) @[jeremymatos](#) · 2 years ago Contributor

[@ohikaj](#) [@dsatchar](#) Can you confirm this is under your group ?

I could reproduce it, this is a variant of Stored XSS: when user visits <https://gitlab.com/thenamespace/project/-/blob/master/xss-openapi1.json>, nothing happens. But as soon as he clicks anywhere in the UI, the XSS payload gets triggered.

 [Kai Armstrong](#) @[ohikaj](#) · 2 years ago Developer

[@jeremymatos](#) This would be [group: source code](#). Adding [@m.qli](#) and [@jamsav](#) for visibility

 [Jeremy Matos](#) @[jeremymatos](#) · 2 years ago Contributor

Thank you

Michelle Gill @m.gill · 2 years ago

@jeremymatos is this a duplicate of #35871 (closed)?

Developer

Jeremy Matos @jeremymatos · 2 years ago

@m.gill no it's different, it's abusing Swagger description and you don't need to click on a link, but anywhere in the UI

Contributor

Please [register](#) or [sign in](#) to reply

✓

Jeremy Matos added group editor / discuss create scoped labels 2 years ago

✓

Kai Armstrong added group source code scoped label and automatically removed group editor label 2 years ago

🕒

Michelle Gill changed milestone to %13.1 2 years ago

✓

Michelle Gill added backlog label 2 years ago

✓

GitLab Bot added Accepting merge requests label 2 years ago

👤

Michelle Gill assigned to @kerrizor 2 years ago

Michelle Gill @m.gill · 2 years ago

@kerrizor this issue is tentatively scheduled for %13.1. Please investigate the issue and follow the process according to create-stage#12641 (closed) before 2020-05-17. When you are finished investigating and have added your comments:

- add the workflow ready for development label
- assign/reassign a weight
- unassign yourself from the issue unless you prefer to take ownership over it

Developer

✓

GitLab Bot removed Accepting merge requests label 2 years ago

✓

Michelle Gill added workflow ready for development scoped label 2 years ago

✓

Michelle Gill added Deliverable label 2 years ago

Michelle Gill @m.gill · 2 years ago

I'm going to estimate 3 here, but please change if that is inaccurate.

Developer

👤

Michelle Gill changed weight to 3 2 years ago

👤

Michelle Gill assigned to @nick.thomas and unassigned @kerrizor 2 years ago

✓

Nick Thomas changed the description 2 years ago

✓

Nick Thomas added workflow in dev scoped label and automatically removed workflow ready for development label 2 years ago

Nick Thomas @nick.thomas · 2 years ago

Contributor

The test case no longer works - although we still send the malicious request, interpreting the response is blocked by:

Cross-Origin Read Blocking (CORB) blocked cross-origin response <URL> with MIME type text/plain. See <URL> for more details.

This is still worth fixing in a security release though, since we're relying on the target server to be non-malicious here. Also, just being able to make the request is sometimes sufficient to wreak havoc.

This is a [bug](#) -heavy issue, but I'll see what I can do.

Edited by Nick Thomas 2 years ago

✓

Nick Thomas added Security label 2 years ago

Nick Thomas @nick.thomas · 2 years ago

Contributor

OK, bumping to the latest version of swagger-ui-dist does not resolve the problem:

- "swagger-ui-dist": "~3.24.3",

+ "swagger-ui-dist": "~3.25.5",

The way this feature works is that we give the raw blob path (e.g. /root/test-118n-Imported/-/raw/master/xss-openapi.json) as data-endpoint here. It is picked up in JS here, and passed on to the swagger library unmodified. It requests the raw JSON, produces the output itself, and injects it into the page.

So there are a few things things we could do, and only of them involves any backend work 🤖

We could introduce a new endpoint that does pre-filtering on the swagger json file, and give that endpoint to swagger-ui. This would involve loading the whole swagger file into memory and filtering every field.

We could capture the output of the swagger-ui-dist process (right now it's injected directly into a named element, I assume it's possible to intercept) and do further filtering of it in JS.

Is there a way we can wrap the injected swagger-ui element inside something that says data-* attributes have no meaning inside here? A sort of lightweight sandboxing?

Or we could follow the security process for swagger-ui-dist, get them to fix the holes mentioned here, perhaps contributing the fix ourselves and update to the fixed version once it's released. That's documented a bit here: <https://github.com/swagger-api/swagger-ui#security-contact>

I doubt we're the only people using swagger-ui inside a page where data-* URLs can have meaning, so the first three options shares the disadvantage of disclosing a vulnerability that make have wider impact. I'd really prefer to see us follow the fourth option first, looking at the others only if we can't get satisfaction that way.

I guess we could also disable this particular viewer as part of a security release, if we decide the last option is what we want to do but we can't make it happen quickly.

I'd like to poll @jeremymatos and @lamphill for opinions on this - WDYT? Are we ok to email security@swagger.io with details of this issue?

Edited by Nick Thomas 2 years ago

Jeremy Matos @jeremymatos · 2 years ago

Contributor

@nick.thomas I would also recommend the option to fix directly swagger-ui, which is clearly the best for the open source community.

Nick Thomas @nick.thomas · 2 years ago

Contributor

OK, I'll send them an email 🙌

Nick Thomas @nick.thomas · 2 years ago

Contributor

OK, Swagger seem keen on fixing this upstream lol/

we'll do our best to put it in next week's release. We'll keep you posted on when it happens.

So I'll keep https://gitlab.com/gitlab-org/security/gitlab/-/merge_requests/572 as a backup option, but hopefully we'll just be able to bump package.json to fix this. I'll check back on the 12th to see where we are.

Edited by Nick Thomas 2 years ago

Nick Thomas @nick.thomas · 2 years ago

Contributor

OK, a new release of swagger-ui has been pushed out, and it fixes the problem: <https://gitlab.com/gitlab-org/security/gitlab/-/issues/170> will track our upgrade to this version.

Please [register](#) or [sign in](#) to reply

Nick Thomas

@nickthomas · 2 years ago

I have a WIP MR for option 3: https://github.com/qitlab-oro/security/qitlab-/merge_requests/577

I don't really like it. In particular, it does nothing to address point 1 in the issue description - we no longer run AJAX queries for the attacker, but they can still control presentation of the whole page.

Still, it might make sense as a defence in depth measure more broadly - if we did something like this to every element that is going to be populated with sanitised user input, it would keep bad parsers from being a security issues, I think.

Edited by Nick Thomas · 2 years ago

Phil Humber

@amphill · 2 years ago

Maintain...

I'm ok with this change, it solves the weird global rails remote data-* attributes we use.

I think in order to solve point 1 we would need to intercept the response with <https://github.com/swaggor-api/swaggor-api/blob/master/docs/usage/configuration.md#network> and then sanitize the response to remove what we don't want.

if we did something like this to every element that is going to be populated with sanitised user input

This I agree with. Though I would like to go 1 step further and just remove the `data-remote` attributes, though that is a much larger issue.

Please [register](#) or [sign in](#) to reply

Nick Thomas

changed health status to at risk · 2 years ago

Nick Thomas

@nickthomas · 2 years ago

Contributor

swaggor-ui does have a comprehensive plugin system. Perhaps we could use that to sanitize user input to remove styles, etc?

Nick Thomas

@nickthomas · 2 years ago

Contributor

What we want to modify is this function: <https://github.com/swaggor-api/swaggor-api/blob/master/src/core/components/providers/markdown.jsx#L52> - it feels like it's possible, judging from the documentation, but I can't quite work out the component system at present 🤔

Nick Thomas

added [swatflow](#) [in review](#) · scoped label and automatically removed [swatflow](#) [in dev](#) · label [2 years ago](#)

GitLab SecurityBot

added [security-issue-escalated](#) · label [2 years ago](#)

GitLab SecurityBot

@qitlab-securitybot · 2 years ago

[@danielqueso](#) [@m_gill](#) [@librouillon](#) This -53 [seconds](#) · issue's milestone has expired.

About this automation: [AppSec Escalation Engine](#)

Nick Thomas

added [swatflow](#) [verification](#) · scoped label and automatically removed [swatflow](#) [in review](#) · label [2 years ago](#)

Nick Thomas

@nickthomas · 2 years ago

Contributor

This issue will be resolved with GitLab versions 13.1.2, 13.0.8, and 12.10.13.

Nick Thomas

closed · 2 years ago

GitLab SecurityBot

removed [security-issue-escalated](#) · label [2 years ago](#)

GitLab SecurityBot

@qitlab-securitybot · 2 years ago

[Author](#) [Report](#)

This [swatflow](#) [security](#) · issue was closed 30 days ago and may become public.

Please ensure the following items are true and add a ☒ reaction:

- Issue description and comments do not contain sensitive data belonging to GitLab.
- Issue does not reveal private information of the reporter (i.e. session IDs, passwords).

If the issue needs to stay confidential, please add the [swatflow-confidential](#) label.

If you removed confidential data from the issue description before making it public, make sure that the description history entry is deleted.

Jeremy Matos

@jeremymatos · 2 years ago

[Contributor](#)

Making issue public

Please [register](#) or [sign in](#) to reply

Jeremy Matos

made the issue visible to everyone · 2 years ago

GitLab SecurityBot

@qitlab-securitybot · 2 years ago

[Author](#) [Report](#)

[HackerOne report #806571](#) was disclosed on 2020-08-04 @ 09:46.

- Bounty awarded: \$2200

Costel Maxim

mentioned in issue [#296857](#) (closed) · 1 year ago

Please [register](#) or [sign in](#) to reply