

RESTler: Stateful REST API Fuzzing

Vaggelis Atlidakis (Columbia University), Patrice Godefroid
(Microsoft Research), and Marina Polishchuk (Microsoft Research)



Over the past decade

- ❖ Explosion of cloud services (in Azure and AWS)
- ❖ Rapidly evolving ecosystem
- ❖ REST APIs is the standard way to use cloud services

Over the past decade

- ❖ Explosion of cloud services (in Azure and AWS)
 - ❖ Rapidly evolving ecosystem
 - ❖ REST APIs is the standard way to use cloud services
- **What about testing?**

Testing REST APIs

Testing REST APIs

- ❖ Grammar-based fuzzing (e.g., Peach, SPIKE, ...)
 - Requires manual effort
 - New grammar for every new service

Testing REST APIs

- ❖ Grammar-based fuzzing (e.g., Peach, SPIKE, ...)
 - Requires manual effort
 - New grammar for every new service
- ❖ HTTP fuzzers (e.g., Sulley, Burp, ...)
 - Requires live traffic
 - Not Stateful

Testing REST APIs

- ❖ Grammar-based fuzzing (e.g., Peach, SPIKE, ...)
 - Requires manual effort
 - New grammar for every new service
- ❖ HTTP fuzzers (e.g., Sulley, Burp, ...)
 - Requires live traffic
 - Not Stateful
- ❖ Custom tools for specific APIs
 - Labour intensive
 - High maintenance

Our solution

- RESTler: A stateful REST API fuzzer

Our solution

- RESTler: A stateful REST API fuzzer

Key techniques for stateful REST API fuzzing

1. Dependency analysis between request types

Our solution

- RESTler: A stateful REST API fuzzer

Key techniques for stateful REST API fuzzing

1. Dependency analysis between request types
2. Dynamic feedback loop that learns from past tests

Our solution

- RESTler: A stateful REST API fuzzer

Kinds of bugs RESTler can find

- “500 Internal Server Error” (unhandled exceptions) after executing a sequence of API requests

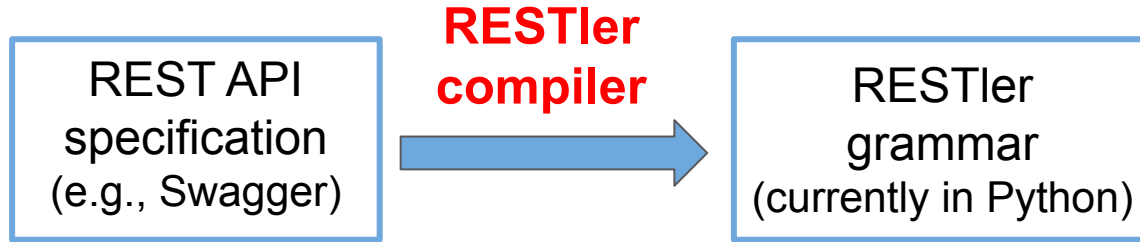
Outline

- ❖ Limitations of existing solutions
- ❖ **System overview**
- ❖ **Evaluation & bugs found**
- ❖ **Experiences with public cloud services**
- ❖ **Conclusions**

System overview

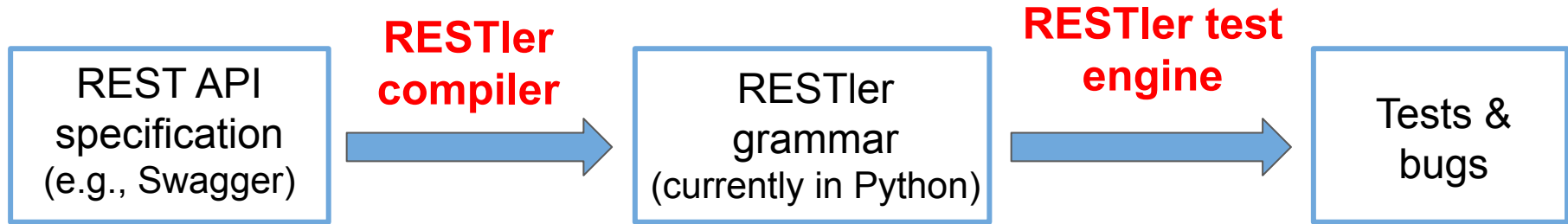
REST API
specification
(e.g., Swagger)

System overview



- ❖ Describe how to fuzz each request type
- ❖ Identify producer/consumer dependencies
- ❖ Generate code to parse responses

System overview



- ❖ Describe how to fuzz each request type
- ❖ Identify producer/consumer dependencies
- ❖ Generate code to parse responses

- ❖ Generate and execute tests: sequences of requests
- ❖ Systematic state-space exploration (breadth first search and others)
- ❖ Analyze test results: Dynamic feedback loop learns from service responses in past tests

RESTler: Stateful REST API Fuzzing

Example

POST	/blog/posts	Creates a new blog post
Response Class (Status 200)		
Success		
Model	Model Schema	
Blog post public {		
body (string): Article content,		
}		
GET	/blog/posts	Returns a list of blog posts
DELETE	/blog/posts/{postId}	Deletes a blog post
GET	/blog/posts/{postId}	Returns a blog post
PUT	/blog/posts/{postId}	Updates a blog post

Sample Swagger specification

```
from restler import requests
from restler import dependencies
```

```
def parse_posts(data):
    post_id = data["id"]
    dependencies.set_var(post_id)

request = requests.Request(
    restler_static("POST"),
    restler_static("/api/blog/posts/"),
    restler_static("HTTP/1.1"),
    restler_static("{}"),
    restler_static("body:"),
    restler_fuzzable("string"),
    restler_static("}"),
    'post_send': {
        'parser': parse_posts,
        'dependencies': [
            post_id.writer(),
        ]
    },
    ...
)
```

RESTler grammar fragment

```
Sending: POST /api/blog/posts/ HTTP/1.1
Accept: application/json
Content-Type: application/json
Host: localhost:8888
{"body": "sampleString"}
```

```
Received: HTTP/1.1 201 CREATED
Content-Type: application/json
Content-Length: 37
Server: Werkzeug/0.14.1 Python/2.7.12
Date: Sun, 01 Apr 2018 05:10:32 GMT
{"body": "sampleString", "id": 5889}
```

Sample test
(request and response)

Outline

- ❖ Limitations of existing solutions
- ❖ System overview
- ❖ **Evaluation & bugs found**
- ❖ Experiences with public cloud services
- ❖ Conclusions

Questions

- Q1: Are tests generated by RESTler exercising deeper service-side logic over time?
- Q2: Can RESTler find bugs in large-scale production services?

Questions

- Q1: Are tests generated by RESTler exercising deeper service-side logic over time?
- Q2: Can RESTler find bugs in large-scale production services?

Case study: Gitlab

- ❖ Open-source self-hosted GIT service (millions of users)
- ❖ ~376 kLOC (Ruby + native libraries)
- ❖ Complex REST API

Deeper service exploration (Q1)

API Family	Total requests	Seq. len.	Cumulative code coverage (lines of code)	Tests
Commits	11	1	598	1
		2	1108	7
		3	1196	250
		4	1760	2220
		5	1760	3667
Branches	7	1	598	1
		2	1089	8
		3	1172	58
		4	1182	576
		5	1185	3644
Issues	22	1	816	37
		2	1163	2444
		3	1163	4156
Repos	10	1	598	1
		2	1117	97
		3	1181	5153

Testing GitLab APIs with RESTler (5h per API family)

Deeper service exploration (Q1)

API Family	Total requests	Seq. len.	Cumulative code coverage (lines of code)	Tests
Commits	11	1	598	1
		2	1108	7
		3	1196	250
		4	1760	2220
		5	1760	3667
Branches	7	1	598	1
		2	1089	8
		3	1172	58
		4	1182	576
		5	1185	3644
Issues	22	1	816	37
		2	1163	2444
		3	1163	4156
Repos	10	1	598	1
		2	1117	97
		3	1181	5153

❖ Longer sequences increase service-side code coverage

Deeper service exploration (Q1)

API Family	Total requests	Seq. len.	Cumulative code coverage (lines of code)	Tests
Commits	11	1	598	1
		2	1108	7
		3	1196	250
		4	1760	2220
		5	1760	3667
Branches	7	1	598	1
		2	1089	8
		3	1172	58
		4	1182	576
		5	1185	3644
Issues	22	1	816	37
		2	1163	2444
		3	1163	4156
Repos	10	1	598	1
		2	1117	97
		3	1181	5153

❖ Longer sequences increase service-side code coverage

❖ Sequences of 3 requests (at least)

Deeper service exploration (Q1)

API Family	Total requests	Seq. len.	Cumulative code coverage (lines of code)	Tests
Commits	11	1	598	1
		2	1108	7
		3	1196	250
		4	1760	2220
		5	1760	3667
Branches	7	1	598	1
		2	1089	8
		3	1172	58
		4	1182	576
		5	1185	3644
Issues	22	1	816	37
		2	1163	2444
		3	1163	4156
Repos	10	1	598	1
		2	1117	97
		3	1181	5153

- ❖ Longer sequences increase service-side code coverage
- ❖ Sequences of 3 requests (at least)
- ❖ Progress in a huge search space

Testing Commits API (5 hours)

- **Brute-force:** 11 request types / 4 renderings on avg / $(11*4)^3 = 85k$ feasible sequences of length 3

Deeper service exploration (Q1)

API Family	Total requests	Seq. len.	Cumulative code coverage (lines of code)	Tests
Commits	11	1	598	1
		2	1108	7
		3	1196	250
		4	1760	2220
		5	1760	3667
Branches	7	1	598	1
		2	1089	8
		3	1172	58
		4	1182	576
		5	1185	3644
Issues	22	1	816	37
		2	1163	2444
		3	1163	4156
Repos	10	1	598	1
		2	1117	97
		3	1181	5153

- ❖ Longer sequences increase service-side code coverage
- ❖ Sequences of 3 requests (at least)
- ❖ Progress in a huge search space

Testing Commits API (5 hours)

- **Brute-force:** 11 request types / 4 renderings on avg / $(11*4)^3 = 85k$ feasible sequences of length 3
- **RESTler:** Seq. Len. 3 / Test generated 250
(feedback + dependencies!)

New bugs found in GitLab (Q2)

API Family	BFS	BFS-Fast	Random-Walk	\cap	U
Commits	5	1	5	1	5
Branches	7	7	7	5	8
Issues	0	1	1	0	1
Repos	2	3	3	2	3
Groups	0	0	2	0	2
Projects	2	1	3	1	3
Total	16	13	21	9	22

Testing GitLab APIs with RESTler (5h per API family)

New bugs found in GitLab (Q2)

API Family	BFS	BFS-Fast	Random-Walk	\cap	U
Commits	5	1	5	1	5
Branches	7	7	7	5	8
Issues	0	1	1	0	1
Repos	2	3	3	2	3
Groups	0	0	2	0	2
Projects	2	1	3	1	3
Total	16	13	21	9	22

❖ 22 new bugs found on Aug. '18
(+6 bugs found on Apr. '18)

Testing GitLab APIs with RESTler (5h per API family)

New bugs found in GitLab (Q2)

API Family	BFS	BFS-Fast	Random-Walk	\cap	U
Commits	5	1	5	1	5
Branches	7	7	7	5	8
Issues	0	1	1	0	1
Repos	2	3	3	2	3
Groups	0	0	2	0	2
Projects	2	1	3	1	3
Total	16	13	21	9	22

- ❖ 22 new bugs found on Aug. '18 (+6 bugs found on Apr. '18)
- ❖ All bugs were disclosed to Gitlab developers

Testing GitLab APIs with RESTler (5h per API family)

New bugs found in GitLab (Q2)

API Family	BFS	BFS-Fast	Random-Walk	\cap	U
Commits	5	1	5	1	5
Branches	7	7	7	5	8
Issues	0	1	1	0	1
Repos	2	3	3	2	3
Groups	0	0	2	0	2
Projects	2	1	3	1	3
Total	16	13	21	9	22

- ❖ 22 new bugs found on Aug. '18 (+6 bugs found on Apr. '18)
- ❖ All bugs were disclosed to Gitlab developers
- ❖ All bugs were easily reproducible, confirmed, and fixed!

Testing GitLab APIs with RESTler (5h per API family)

New bugs found in GitLab (Q2)

API Family	BFS	BFS-Fast	Random-Walk	\cap	U
Commits	5	1	5	1	5
Branches	7	7	7	5	8
Issues	0	1	1	0	1
Repos	2	3	3	2	3
Groups	0	0	2	0	2
Projects	2	1	3	1	3
Total	16	13	21	9	22

❖ Example Bug [#50268]

1. Create a gitlab project
2. Create a repository file with a proper commit message
3. Delete the repository file with an empty commit message

Testing GitLab APIs with RESTler (5h per API family)

New bugs found in GitLab (Q2)

API Family	BFS	BFS-Fast	Random-Walk	\cap	U
Commits	5	1	5	1	5
Branches	7	7	7	5	8
Issues	0	1	1	0	1
Repos	2	3	3	2	3
Groups	0	0	2	0	2
Projects	2	1	3	1	3
Total	16	13	21	9	22

❖ Example Bug [#50268]

1. Create a gitlab project
2. Create a repository file with a proper commit message
3. Delete the repository file with an empty commit message

➤ “500 Internal Server Error”

Testing GitLab APIs with RESTler (5h per API family)

Outline

- ❖ Limitations of existing solutions
- ❖ System overview
- ❖ Evaluation & bugs found
- ❖ **Experiences with public cloud services**
- ❖ Conclusions

Experiences with Azure and Office 365

- ❖ Four production cloud services with open-source specs
 - Resource management Azure services
 - Real-time messaging Office 365 service

Experiences with Azure and Office 365

- ❖ Four production cloud services with open-source specs
 - Resource management Azure services
 - Real-time messaging Office 365 service
- ❖ Needed new features
 - Garbage Collection (resource quotas)
 - Authentication Hooks (short-lived access tokens)
 - Resource-specific mutations (exotic naming schemes)

Experiences with Azure and Office 365

- ❖ Four production cloud services with open-source specs
 - Resource management Azure services
 - Real-time messaging Office 365 service
- ❖ Needed new features
 - Garbage Collection (resource quotas)
 - Authentication Hooks (short-lived access tokens)
 - Resource-specific mutations (exotic naming schemes)
- **RESTler found bugs in all services tested so far!**

Conclusions

- ❖ Build the first stateful REST API fuzzer!
- ❖ Found bugs in Azure and Office 365 cloud services!
- ❖ Found 28 new bugs in Gitlab!

Conclusions

- ❖ Build the first stateful REST API fuzzer!
- ❖ Found bugs in Azure and Office 365 cloud services!
- ❖ Found 28 new bugs in Gitlab!

➤ **Developers are fixing the bugs found with RESTler!**



Thank you!

Paper link

<https://tinyurl.com/yyg5a8je>



Thank you!

Paper link

<https://tinyurl.com/yyg5a8je>

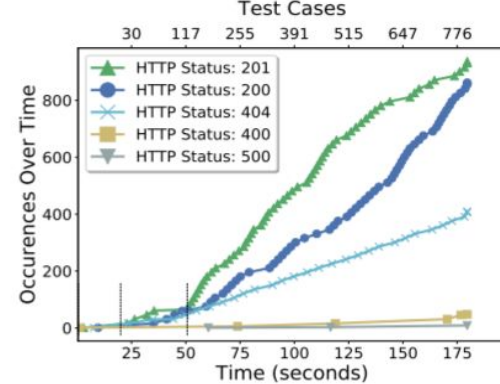
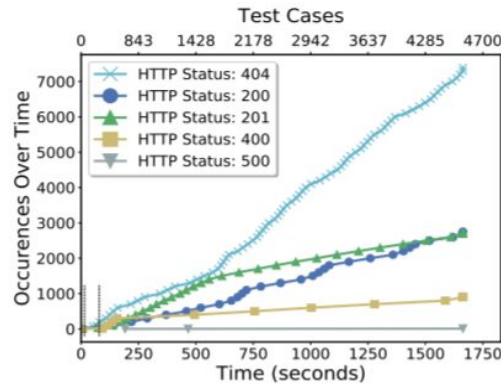
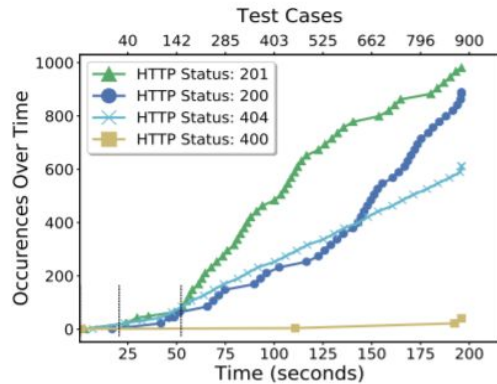
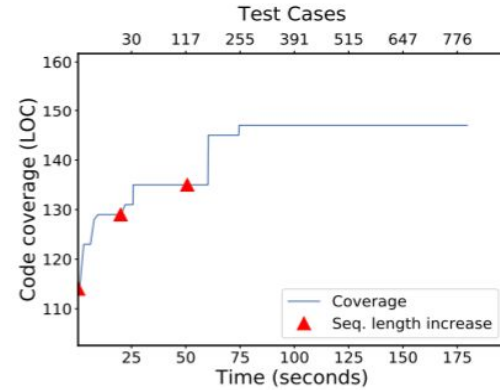
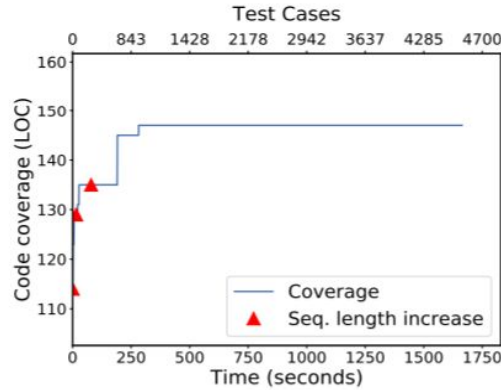
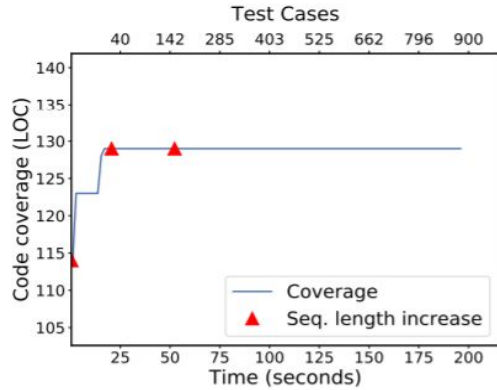


Scalability of state-space exploration strategies

API	Total Requests	Time (hrs)	BFS			BFS-Fast			RandomWalk		
			Len.	Coverage	seqSet	Len.	Coverage	seqSet	Len. (restarts)	Coverage	seqSet
Commits	11 (*11)	1	4	1202		7	1697		13 (16)	1285	
		3	5	1760		9	1731		13 (35)	1295	
		5	5	1760	20679	12	1731	33	13 (56)	1303	1
Branches	7 (*2)	1	5	1182		21	1154		15 (24)	1182	
		3	5	1185		37	1178		19 (92)	1187	
		5	5	1185	5528	47	1178	11	22 (158)	1208	1
Issues	22 (*82)	1	2	1150		2	1086		10 (1)	770	
		3	3	1163		4	1551		10 (1)	770	
		5	3	1163	15658	5	1570	26	16 (2)	847	1
Repos	10 (*24)	1	3	1127		5	1141		10 (29)	1195	
		3	3	1127		7	1141		13 (88)	1231	
		5	3	1181	2194	8	1161	64	13 (142)	1231	1
Groups	50 (*2)	1	2	961		6	1275		19 (41)	1167	
		3	3	1177		11	1275		19 (120)	1250	
		5	3	1177	79518	14	1275	130	22 (186)	1283	1
Projects	48 (*4)	1	2	1006		5	1318		4 (3)	889	
		3	2	1053		11	1319		22 (31)	1024	
		5	3	1203	18173	15	1319	171	22 (45)	1273	1

RESTler: Stateful REST API Fuzzing

Impact of the two key techniques



Extending sequences in Randoop

```
public class A {
    public A() {...}
    public B m1(A a1) {...}
}

public class B {
    public B(int i) {...}
    public void m2(B b, A a) {...}
}
```

sequence s_1	sequence s_2	sequence s_3
B b1 = new B(0);	B b2 = new B(0);	A a1 = new A(); B b3 = a1.m1(a1);

<i>seqs</i>	<i>vals</i>	<i>extend(m2, seqs, vals)</i>
$\langle s_1, s_3 \rangle$	$\langle s_{1.1}, s_{1.1}, s_{3.1} \rangle$ (i.e.: b1, b1, a1)	B b1 = new B(0); A a1 = new A(); B b3 = a1.m1(a1); b1.m2(b1, a1);
$\langle s_3, s_1 \rangle$	$\langle s_{1.1}, s_{1.1}, s_{3.1} \rangle$ (i.e.: b1, b1, a1)	A a1 = new A(); B b3 = a1.m1(a1); B b1 = new B(0); b1.m2(b1, a1);
$\langle s_1, s_2 \rangle$	$\langle s_{1.1}, s_{2.1}, \text{null} \rangle$ (i.e.: b1, b2, null)	B b1 = new B(0); B b2 = new B(0); b1.m2(b2, null);

RESTler: Stateful REST API Fuzzing

Sample bugfix in Gitlab

Showing 3 changed files ▾

+1E

▼ [changelogs/unreleased/api-empty-commit-message.yml](#) 0 → 100644 [🔗](#)

[View file @ ceae58c](#)

```
1 + ---
2 + title: 'API: Catch empty commit messages'
3 + merge_request: 21322
4 + author: Robert Schilling
5 + type: fixed
```

▼ [lib/api/files.rb](#) [🔗](#)

[View file @ ceae58c](#)

```
...   ...   @@ -59,7 +59,7 @@ module API
59     59     params :simple_file_params do
60     60       requires :file_path, type: String, desc: 'The url encoded path to the file. Ex. lib%2Fclass%2Erb'
61     61       requires :branch, type: String, desc: 'Name of the branch to commit into. To create a new branch,
        also provide `start_branch`.'
62     62 -     requires :commit_message, type: String, desc: 'Commit message'
63     63 +     requires :commit_message, type: String, regexp: /\S+/, desc: 'Commit message'
64     64     optional :start_branch, type: String, desc: 'Name of the branch to start the new commit from'
65     65     optional :author_email, type: String, desc: 'The email of the author'
        optional :author_name, type: String, desc: 'The name of the author'
...   ...
```

▼ [spec/requests/api/files_spec.rb](#) [🔗](#)

[View file @ ceae58c](#)

```
...   ...   @@ -337,6 +337,18 @@ describe API::Files do
337   337     expect(response).to have_gitlab_http_status(400)
338   338     end
339   339
340   340 +   it 'returns a 400 bad request if the commit message is empty' do
341   341 +     invalid_params = {
342   342 +       branch: 'master',
343   343 +       content: 'puts 8',
344   344 +       commit_message: ''
345   345 +     }
346   346 +
347   347 +     post api(route(file_path), user), invalid_params
348   348 +
349   349 +     expect(response).to have_gitlab_http_status(400)
350   350 +   end
351   351 +
352   352   it "returns a 400 if editor fails to create file" do
353   353     allow_any_instance_of(Repository).to receive(:create_file)
354   354     .and_raise(Gitlab::Git::CommitError, 'Cannot create file')
...   ...
```

RESTler: Stateful REST API Fuzzing

Developers' Responses

Patrice, thank you for reporting the bugs!
Plz provide instructions on how integrate the tool into the build

Please file VSO for each – these are real bugs. We already fixed a few in ██████████

```
changing 3 changed files with 47 additions and 18 deletions
changelogs/unreleased/api-empty-commit-message.yml
1 + ---
2 + title: 'API: Catch empty commit messages'
3 + merge_request: 21322
4 + author: Robert Schilling
5 + type: fixed

lib/api/files.rb
... @@ -59,7 +59,7 @@ module API
59   params :simple_file_params do
60     requires :file_path, type: String, desc: 'The url encoded path to the file. Ex.
lib%2Fclass%2Erb'
61     requires :branch, type: String, desc: 'Name of the branch to commit into. To create a new
branch, also provide 'start_branch'.'
62 -   requires :commit_message, type: String, desc: 'Commit message'
62 +   requires :commit_message, type: String, allow_blank: false, desc: 'Commit message'
```

Mark Fletcher @markglenfletcher · 1 week ago #50276 Maintainer

@vatlidak1 This one appears to be associated with the `code` attribute rather than `title`? Please can you check the title and description of this issue?

Mark Fletcher @markglenfletcher added `Create` `api` `bug` `devops.create` `snippets` labels · 1 week ago

Mark Fletcher @markglenfletcher · 8 months ago #50677

@vatlidak1 Thanks for the report. I was able to reproduce it.

Let's open this for a ~"Community Contribution" with ~"Accepting Merge Requests"

Mark Fletcher @markglenfletcher added `Accepting merge requests` `Create` `api` `bug` `reproduced on GitLab.com` labels 8 months ago

Mark Fletcher @markglenfletcher · 9 months ago #50272 Maintainer

@vatlidak1 Thanks for the report. I presume that the project is still available on disk at this point and the deferred project execution has not yet been completed

Mark Fletcher @markglenfletcher added `Manage` `api` `bug` `project` `repository` labels 9 months ago

500 Internal Server Error: Delete repository file with empty commit message #50268

1 Related Merge Request

!21322 API: Catch empty commit messages Open

When this merge request is accepted, this issue will be closed automatically.