

[b-c-ds / CVE-2021-27291-pygments.txt](#)

Created last year

☆ Star

<> Code Revisions 1 ☆ Stars 1 🍴 Forks 1

cve-2021-27291

CVE-2021-27291-pygments.txt

```
1 Doyensec Vulnerability Advisory
2 CVE-2021-27291
3 =====
4 * Regular Expression Denial of Service (REDoS) in pygments
5 * Affected Product: pygments v1.1+, fixed in 2.7.4
6 * Vendor: https://github.com/pygments
7 * Severity: Medium
8 * Vulnerability Class: Denial of Service
9 * Status: Fixed
10 * Author(s): Ben Caller (Doyensec)
11 =====
12
13 === SUMMARY ===
14
15 In pygments, the lexers used to parse programming languages rely heavily on regular expressions.
16 Some of the regular expressions have exponential or cubic worst-case complexity and are vulnerable to Regular Expression Denial of Service
17 By crafting malicious input, an attacker can cause Denial of Service.
18
19 === TECHNICAL DESCRIPTION ===
20
21 The vulnerable regular expressions are below. Line numbers refer to pygments version 2.7.3.
22
23 pygments/lexers/archetype.py #61
24 Pattern: [+-]?(\d+)*\.\d+%?
25 Complexity: exponential
26 Example: '0' * 3456
27 Repeated character: \d
28 Languages: ODIN, CADL, ADL
29
30 The above shows that the python code
31
32     re.match(r"[+-]?(\d+)*\.\d+%?", "0" * 123)
33
34 will run approximately forever.
35
36 pygments/lexers/factor.py #268
37 Pattern: ""\s+(?:\.\n)*?\s+""
38 Complexity: cubic
39 Repeated character: \s
40 Example: '""' + ' ' * 3456
41 Languages: Factor
42
43 pygments/lexers/factor.py #325
44 Pattern: (\{(\s+)(\s+)(\s+)[^}]+\s+\\)\s)
45 Complexity: cubic
46 Repeated character: \s
47 Example: '{ 0' + ' ' * 3456
48 Languages: Factor
49
50 pygments/lexers/jvm.py #984
51 Pattern: ".*`.*`.*"
52 Complexity: cubic
53 Repeated character: \x60 (')
54 Example: '""' + ' ' * 3456
55 Languages: Ceylon
56
57 pygments/lexers/matlab.py #140
58 pygments/lexers/matlab.py #641
59 pygments/lexers/matlab.py #713
60 Pattern: (\s*)(?:\.(+)(\s*)(=)(\s*))?(.(+)(\s*)(\s*))(\s*)
61 Complexity: cubic
62 Repeated character: \s
63 Example: ' ' * 3456
64 Languages: Matlab, Octave, Scilab
65
66 pygments/lexers/objective.py #264
67 Pattern: (%config)(\s*(\s*)(w+)(\s*=\s*)(.??)(\s*)(\s*))
68 Complexity: cubic
69 Repeated character: \s
70 Example: '%config(a=' + ' ' * 3456
71 Languages: Logos
72
73 pygments/lexers/objective.py #268
74 Pattern: (%new)(\s*)(\s*)(\s*.*?\s*)(\s*)
75 Complexity: cubic
76 Repeated character: \s
77 Example: '%new(' + ' ' * 3456
78 Languages: Logos
79
80 pygments/lexers/templates.py #1408
```

```

81 Pattern: (\$(evoque|overlay)(\{((%))(\s*[\w\-\'\.]+[^\=,%])+\})?(.*?)(?(4)%\}))
82 Complexity: cubic
83 Repeated character: [22:",23:#,27:',aa,2d:-,2e:.,b5,ba,[f8-ff],[a-z],[A-Z],[c0-d6],[d8-f6]]
84 Example: '$evoque{' + 'a' * 3456
85 Languages: Evoque
86
87 pygments/lexers/varnish.py #64
88 Pattern: (\.\w+\b)(\s*=\s*)([^\s;]*)\s*;
89 Complexity: cubic
90 Repeated character: \s
91 Example: '.a=' + ' ' * 3456
92
93
94 === REPRODUCTION STEPS ===
95
96 In some cases, the lexer will only use the vulnerable regex when a prefix is added to the input.
97 As an example, causing REDoS via the ODIN / CADL lexer requires a '<' before the long string of digits.
98
99 Create a file redos.odin containing:
100
101 <00000000000000000000000000000000
102
103 Run 'pygmentize redos.odin'. It will run for a very long time.
104 As the complexity is exponential, adding one extra digit will double the processing time.
105 For cubic complexity REDoS, doubling the length of the repeating section makes processing take 8 times as long.
106
107 Below are recipes for creating source code files which cause REDoS:
108
109 ADL: 'language\n <' + '0' * 30
110 CADL / ODIN: '<' + '0' * 30
111 Ceylon: '"' + ' ' * 3456
112 Evoque: '$evoque{' + 'a' * 3456
113 Factor: '""'+ " " * 3456
114 Logos: '%new(' + ' ' * 3456
115 Matlab: 'function' + ' ' * 3456
116 Varnish VCL: 'backend x{.a=' + ' ' * 3456
117
118
119 === REMEDIATION ===
120
121 Fix the regular expressions to avoid overlapping capture groups.
122
123 === DISCLOSURE TIMELINE ===
124
125 2020-12-29: Vulnerability disclosed via email to maintainer
126 2021-01-11: Fixed in https://github.com/pygments/pygments/commit/2e7e8c4a7b318f4032493773732754e418279a14
127 2021-01-12: Patched version 2.7.4 released
128
129 =====
130
131 Doyensec (www.doyensec.com) is an independent security research
132 and development company focused on vulnerability discovery and
133 remediation. We work at the intersection of software development
134 and offensive engineering to help companies craft secure code.
135
136 Copyright 2020 by Doyensec LLC. All rights reserved.
137
138 Permission is hereby granted for the redistribution of this
139 advisory, provided that it is not altered except by reformatting
140 it, and that due credit is given. Permission is explicitly given
141 for insertion in vulnerability databases and similar, provided
142 that due credit is given. The information in the advisory is
143 believed to be accurate at the time of publishing based on
144 currently available information, and it is provided as-is,
145 as a free service to the community by Doyensec LLC. There are
146 no warranties with regard to this information, and Doyensec LLC
147 does not accept any liability for any direct, indirect, or
148 consequential loss or damage arising from use of, or reliance
149 on, this information.

```

b-c-ds commented on May 11, 2021

Author

Discovered with [regexploit](#).
[Blog](#)