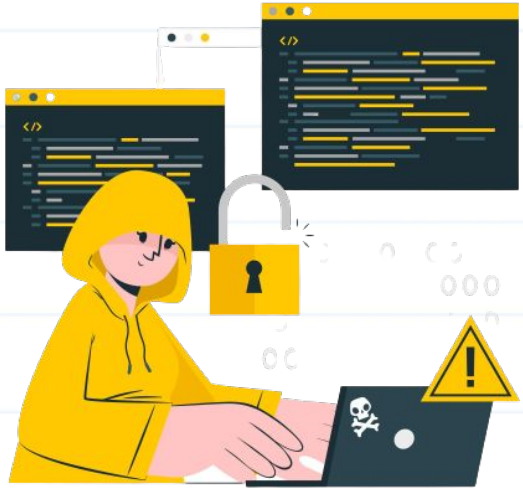
2024 djangocon.us DURHAM

PyGoat

Learn Django security the hard way



Adarsh Divakaran
Co-Founder and Lead Consultant
@ Digievo Labs

Web Application Security

Web application security encompasses a range of measures, technologies, or approaches aimed at **safeguarding** web servers, web applications, and web services, including APIs, against potential attacks **from online threats**.

Web Application Security

- Based on an analysis of 14 million websites, SiteLock reports that websites currently experience an average of **172 attacks every day**
- **Cyberattacks** cost small and medium-Sized Businesses $25k annually on average
- The volume of threats are **doubling** year over year

Source: https://www.sitelock.com/resources/security-report

Web Application Security

For the businesses, these attacks lead to:

- Loss of **credibility**
- **Financial** loss
- Compromise of **user data**

https://haveibeenpwned.com

Oh no — pwned!

Pwned in 4 data breaches and found no pastes (subscribe to search sensitive breaches)

3 Steps to better security

Start using 1Password.com

Step 1 Protect yourself using 1Password to generate and save strong passwords for each website.

Step 2 Enable 2 factor authentication and store the codes inside your 1Password account.

Step 3 Subscribe to notifications for any other breaches. Then just change that unique password.

Why 1Password?

Donate

Breaches you were pwned in

A "breach" is an incident where data has been unintentionally exposed to the public. Using the 1Password password manager helps you ensure all your passwords are strong and unique such that a breach of one service doesn't put your other services at risk.

Ensuring Web Application Security

- Lots of resources are available on building Web applications
- Relatively less guidance and resources for building **Secure Web Applications**.
- Security is often learned the **hard way**.

OWASP

The **Open Worldwide Application Security Project (OWASP)** is an online community that produces freely-available articles, methodologies, documentation, tools, and technologies in the field of web application security.

OWASP provides tools and resources for **security researchers** as well as **developers** to **test and fortify** their applications.

OWASP Top 10

The OWASP Top 10 is a regularly-updated report outlining security concerns for web application security, focusing on the **10 most critical risks**.

The report is put together by a team of security experts from all over the world.

Current version: 2021

https://owasp.org/www-project-top-ten/

RANK	VULNERABILITY
A1	Broken Access Control
A2	Cryptographic Failures
A3	Injection
A4	Insecure Design
A5	Security Misconfiguration
A6	Vulnerable and Outdated Components
A7	Identification and Authentication Failures
A8	Software and Data Integrity Failures
A9	Security Logging and Monitoring Failures
A10	Server-Side Request Forgery (SSRF)

The Pygoat Project

Pygoat is an **intentionally vulnerable** Python Django application that can be used to learn to secure our Django apps.

Pygoat contains (intentionally insecure) **labs** to learn about and exploit the OWASP top 10 vulnerabilities.

A3: Injection

Injection happens when an attacker sneaks in **untrusted data** and **tricks** the interpreter into doing things it shouldn't, like running **unintended commands** or accessing **unauthorized data**.

Eg: SQL, NoSQL, OS command, Object Relational Mapping (ORM) injection

Pygoat Lab 1

Mission: Get unauthorized access as admin

Security Testing Methodology

A security researcher will:

- Perform **enumeration** against the target - Identify the tech stack, various operations and find points of attack.
- They will then proceed with **testing** the web application functionalities (against multiple vulnerabilities) to uncover security bugs - using automated or manual tools.

Our Methodology

- We will look into various **Pygoat labs**
- Each lab has an objective
- Since Pygoat source code is available to us, we can easily spot the security issues and attack each of the labs without much trial and error.

Demo

Pygoat Lab 1

```
SELECT * FROM introduction_login WHERE
user='"+name+"'AND password='"+password+"'"
```

When name is “john” and password is “pass1”, it becomes:

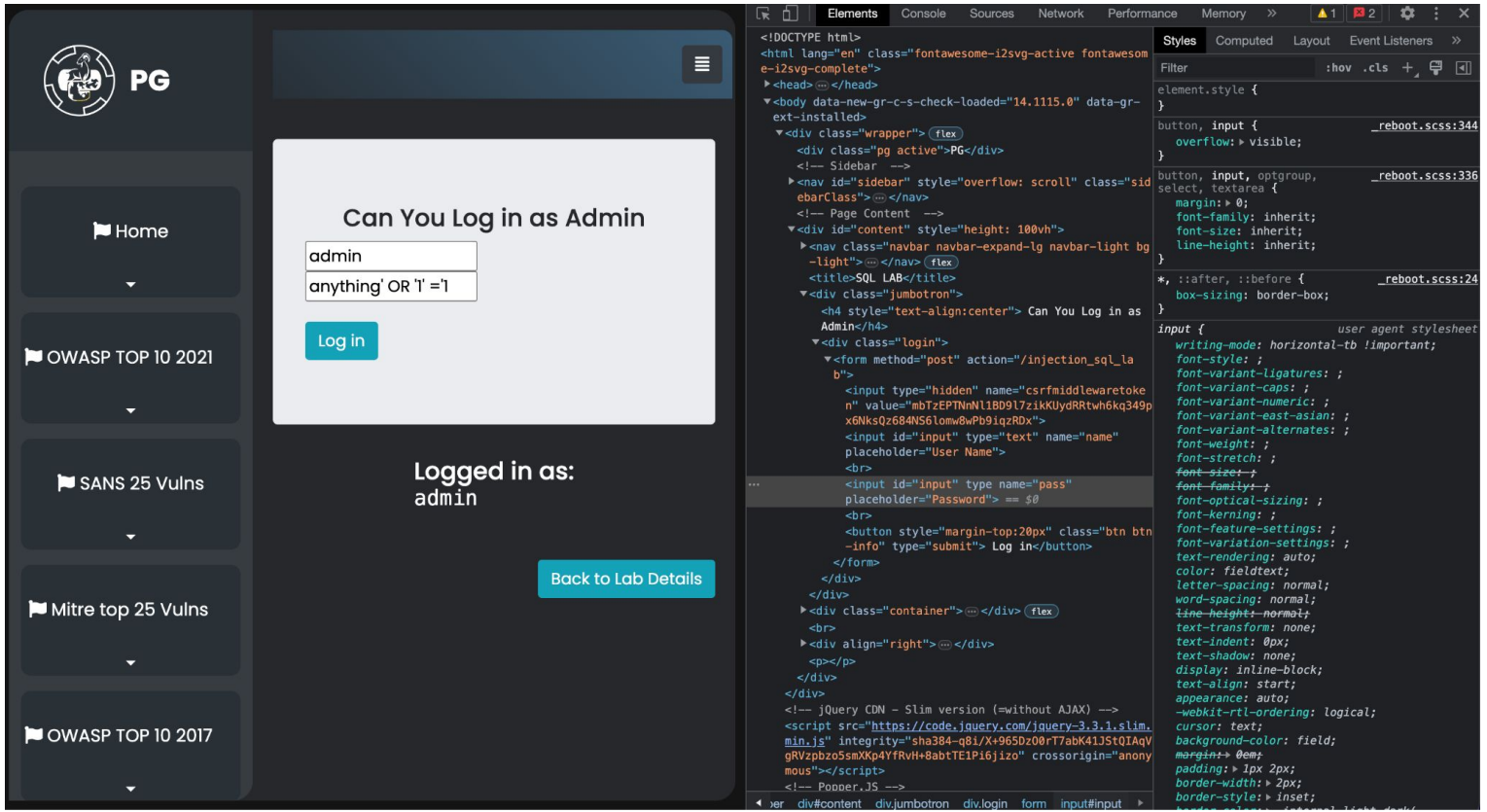
```
SELECT * FROM introduction_login WHERE
user='john' AND password='pass1'
```

Pygoat Lab 1

```
SELECT * FROM introduction_login WHERE
user='"+name+"'AND password='"+password+"'"
```

What if we enter password as: `any' OR '1'='1`

```
SELECT * FROM introduction_login WHERE
user='john' AND password='any' OR '1'='1'
```


PG
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Can You Log in as Admin
admin
anything' OR '1' ='1
Log in
Logged in as:
admin
Back to Lab Details
Elements
Console
Sources
Network
Performance
Memory
Styles
Computed
Layout
Event Listeners
Filter
:hov .cls
element.style {
}
button, input { _reboot.scss:344
overflow: visible;
}
button, input, optgroup, select, textarea { _reboot.scss:336
margin: 0;
font-family: inherit;
font-size: inherit;
line-height: inherit;
}
*, ::after, ::before { _reboot.scss:24
box-sizing: border-box;
}
input { user agent stylesheet
writing-mode: horizontal-tb !important;
font-style: ;
font-variant-ligatures: ;
font-variant-caps: ;
font-variant-numeric: ;
font-variant-east-asian: ;
font-variant-alternates: ;
font-weight: ;
font-stretch: ;
font-size: ;
font-family: ;
font-optical-sizing: ;
font-kerning: ;
font-feature-settings: ;
font-variation-settings: ;
text-rendering: auto;
color: fieldtext;
letter-spacing: normal;
word-spacing: normal;
line-height: normal;
text-transform: none;
text-indent: 0px;
text-shadow: none;
display: inline-block;
text-align: start;
appearance: auto;
-webkit-rtl-ordering: logical;
cursor: text;
background-color: field;
margin: 0em;
padding: 1px 2px;
border-width: 2px;
border-style: inset;

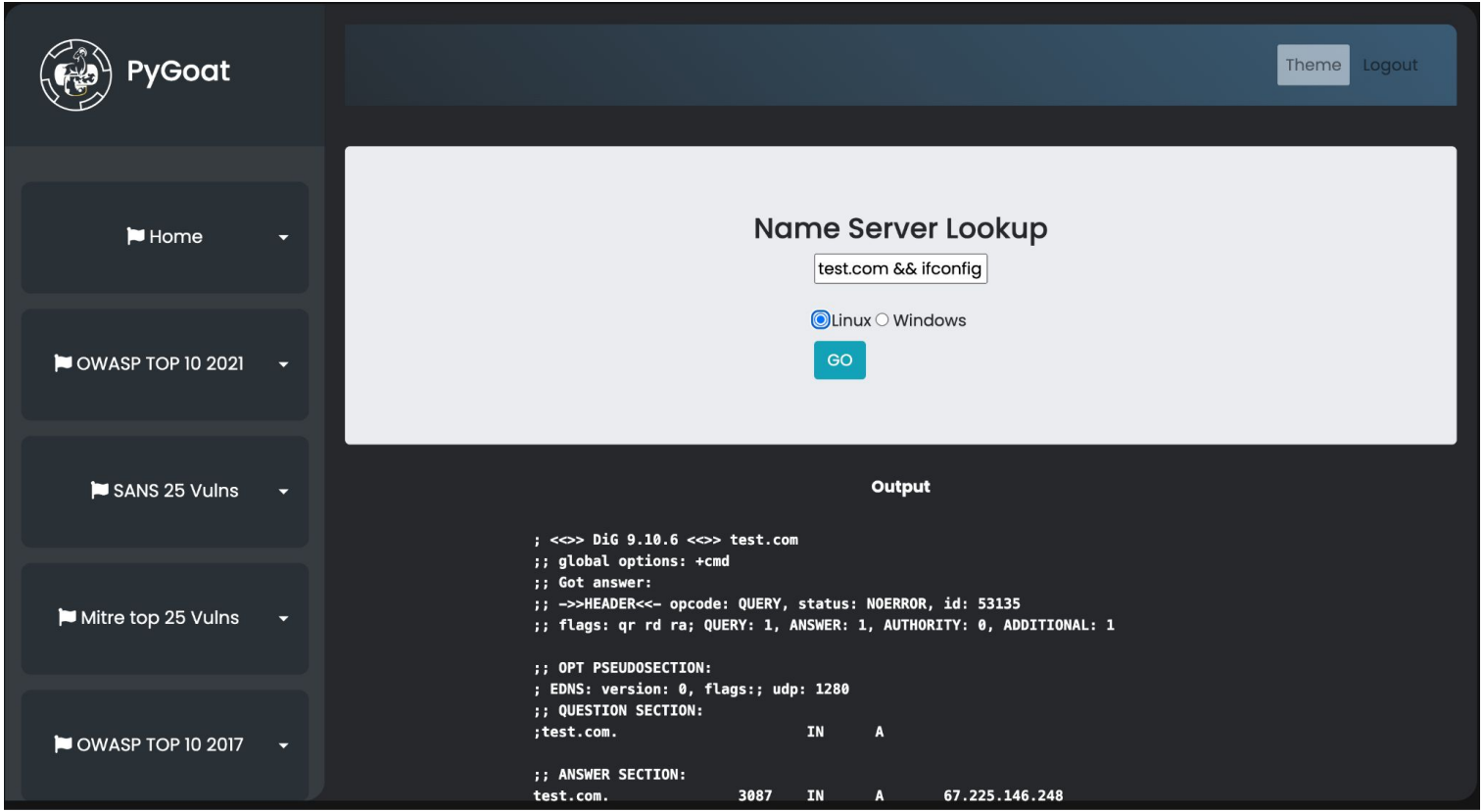
Vulnerable Code

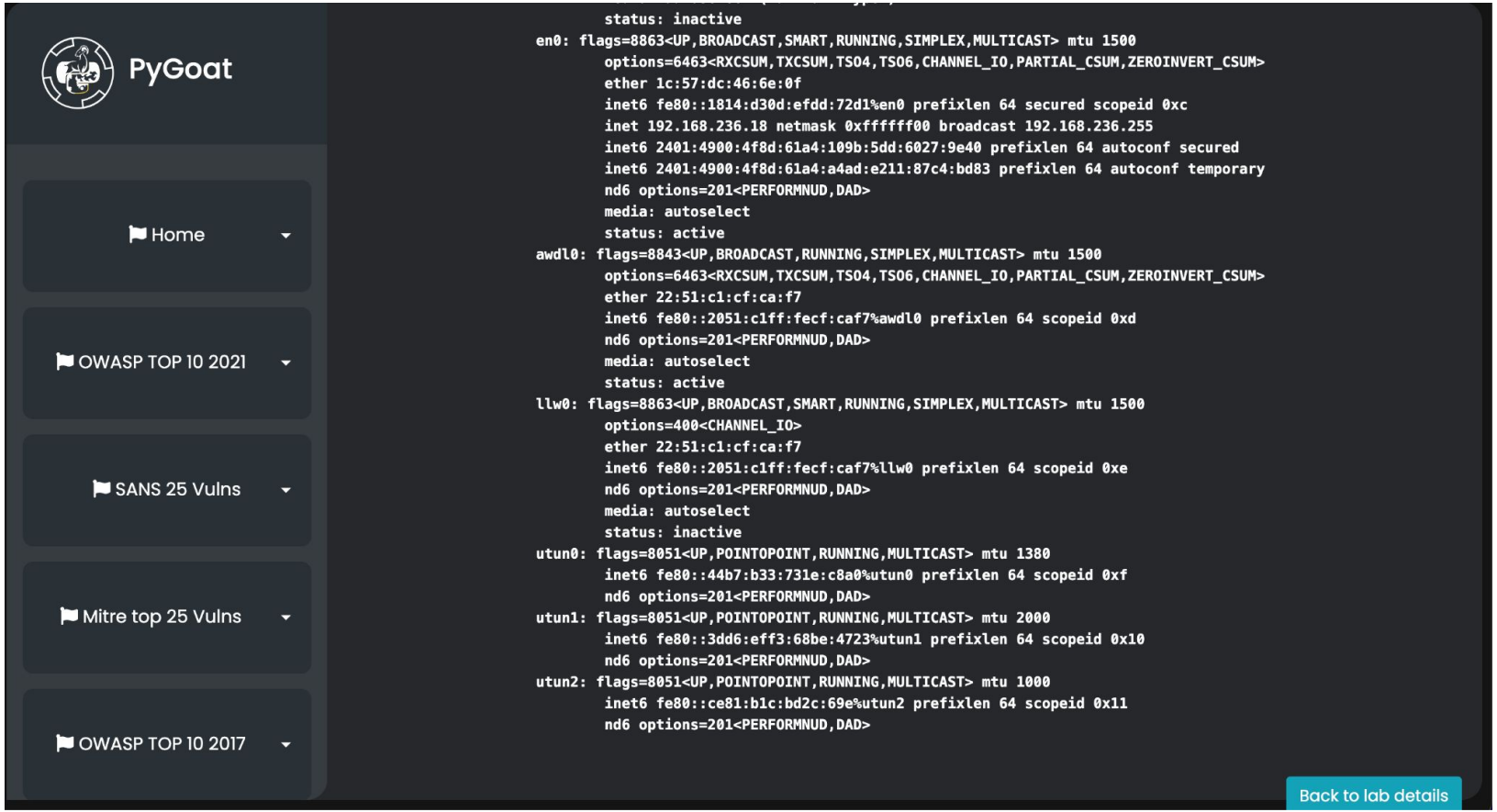
```
if name:
    sql_query = "SELECT * FROM introduction_sql_lab_table WHERE id='"+name+"'AND
password='"+password+"'"

    sql_instance = sql_lab_table(id="admin", password="65079b006e85a7e798abecb99e47c154")
    sql_instance.save()
    sql_instance = sql_lab_table(id="jack", password="jack")
    sql_instance.save()
    sql_instance = sql_lab_table(id="slinky", password="b4f945433ea4c369c12741f62a23ccc0")
    sql_instance.save()
    sql_instance = sql_lab_table(id="bloke", password="f8d1ce191319ea8f4d1d26e65e130dd5")
    sql_instance.save()

    print(sql_query)

    try:
        user = sql_lab_table.objects.raw(sql_query)
        user = user[0].id
        print(user)
```

Pygoat Lab 2

Mission: Run shell commands on the server

Demo


PyGoat
Theme
Logout
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Name Server Lookup
test.com && ifconfig
Linux
Windows
GO
Output
; <<>> DiG 9.10.6 <<>> test.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 53135
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1280
;; QUESTION SECTION:
;test.com. IN A
;; ANSWER SECTION:
test.com. 3087 IN A 67.225.146.248


PyGoat
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
status: inactive
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
options=6463<RXCSUM,TXCSUM,TSO4,TSO6,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
ether 1c:57:dc:46:6e:0f
inet6 fe80::1814:d30d:efdd:72d1%en0 prefixlen 64 secured scopeid 0xc
inet 192.168.236.18 netmask 0xffffff00 broadcast 192.168.236.255
inet6 2401:4900:4f8d:61a4:109b:5dd:6027:9e40 prefixlen 64 autoconf secured
inet6 2401:4900:4f8d:61a4:a4ad:e211:87c4:bd83 prefixlen 64 autoconf temporary
nd6 options=201<PERFORMNUD,DAD>
media: autoselect
status: active
awdl0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
options=6463<RXCSUM,TXCSUM,TSO4,TSO6,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
ether 22:51:c1:cf:ca:f7
inet6 fe80::2051:c1ff:fecf:caf7%awdl0 prefixlen 64 scopeid 0xd
nd6 options=201<PERFORMNUD,DAD>
media: autoselect
status: active
llw0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
options=400<CHANNEL_IO>
ether 22:51:c1:cf:ca:f7
inet6 fe80::2051:c1ff:fecf:caf7%llw0 prefixlen 64 scopeid 0xe
nd6 options=201<PERFORMNUD,DAD>
media: autoselect
status: inactive
utun0: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1380
inet6 fe80::44b7:b33:731e:c8a0%utun0 prefixlen 64 scopeid 0xf
nd6 options=201<PERFORMNUD,DAD>
utun1: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 2000
inet6 fe80::3dd6:eff3:68be:4723%utun1 prefixlen 64 scopeid 0x10
nd6 options=201<PERFORMNUD,DAD>
utun2: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1000
inet6 fe80::ce81:b1c:bd2c:69e%utun2 prefixlen 64 scopeid 0x11
nd6 options=201<PERFORMNUD,DAD>
Back to lab details

```
if (request.method == "POST"):
    domain = request.POST.get('domain')
    domain = domain.replace("https://www.", '')
    os = request.POST.get('os')
    print(os)
    if (os == 'win'):
        command = "nslookup {}".format(domain)
    else:
        command = "dig {}".format(domain)

    try:
        # output=subprocess.check_output(command,shell=True,encoding="UTF-8")
        process = subprocess.Popen(
            command,
            shell=True,
            stdout=subprocess.PIPE,
            stderr=subprocess.PIPE)
        stdout, stderr = process.communicate()
        data = stdout.decode('utf-8')
        stderr = stderr.decode('utf-8')
        # res = json.loads(data)
        # print("Stdout\n" + data)
        output = data + stderr
        print(data + stderr)
    except:
        output = "Something went wrong"
        return render(request, 'Lab/CMD/cmd_lab.html', {"output": output})
    print(output)
    return render(request, 'Lab/CMD/cmd_lab.html', {"output": output})
else:
    return render(request, 'Lab/CMD/cmd_lab.html')
```

Mitigation

- The preferred option is to use **a safe API**, which avoids using the interpreter entirely, provides a parameterized interface, or migrate to Object Relational Mapping Tools (ORMs).
- Use positive **server-side input validation**. This is not a complete defense as many applications require special characters.
- For any residual dynamic queries, **escape** special characters using the specific escape syntax for that interpreter.

Mitigation

Django Specific:

- **Django ORM** usage removes SQLi vulnerability. Raw queries should be avoided if possible.
- **Django templates** has built-in protection against (most) XSS attacks, another injection vulnerability.

Python specific:

- **Do not** allow **command/code execution** (os.system, subprocess or eval) using **user supplied inputs**.

A5: Security Misconfiguration

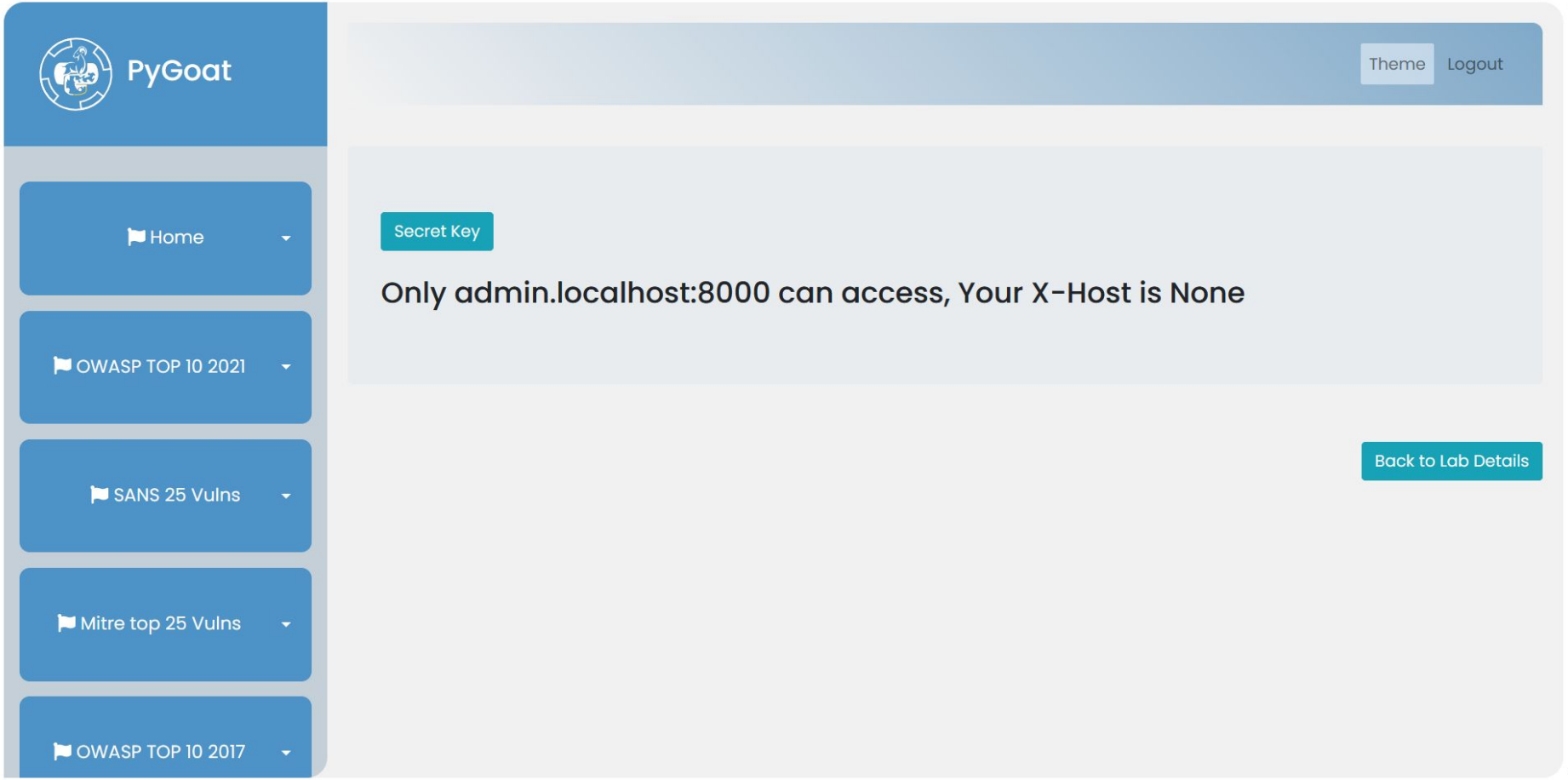
The application might be vulnerable if the application is:

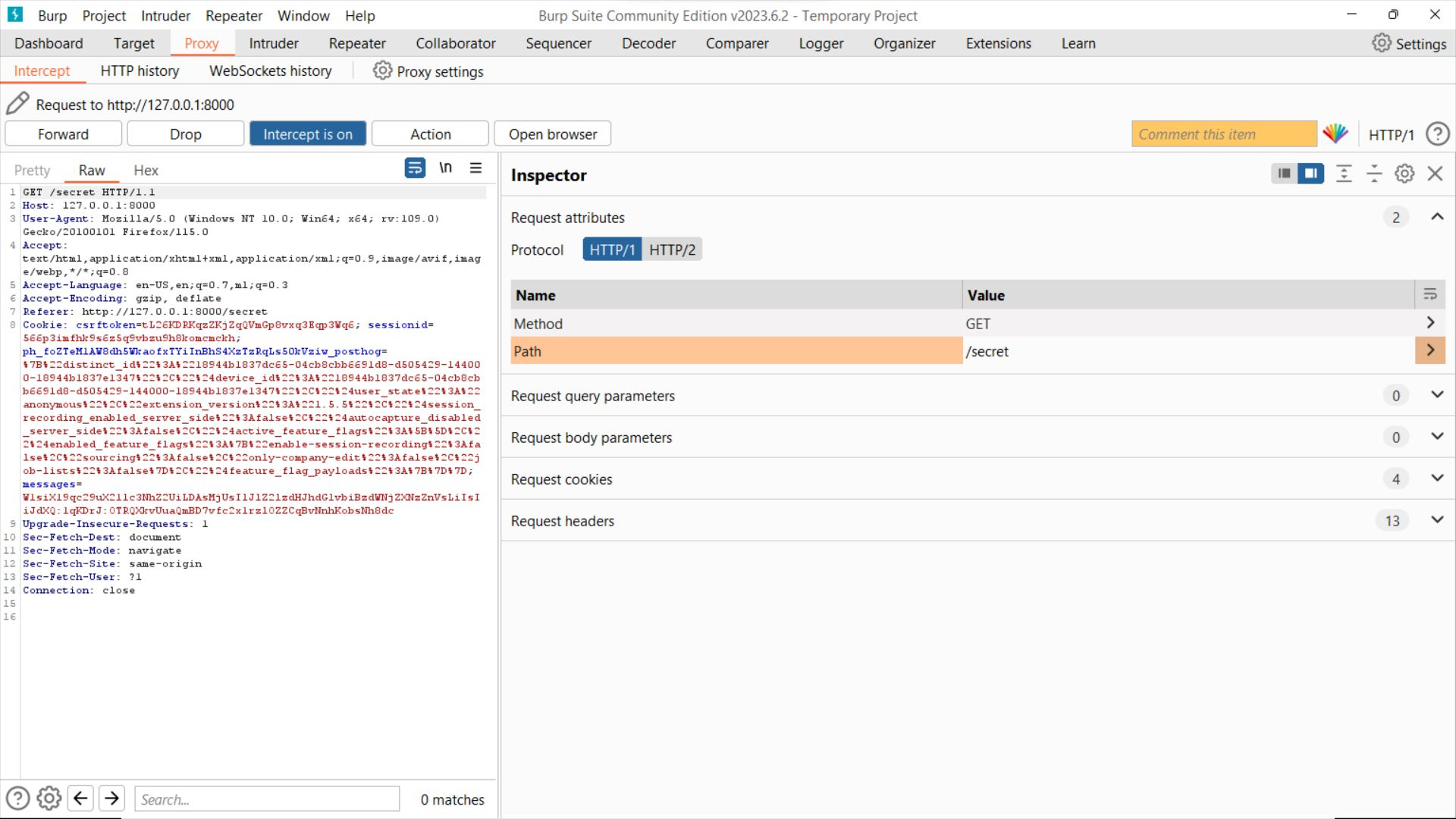
- **Missing** appropriate **security hardening** across any part of the application stack or improperly configured permissions on cloud services.
- **Unnecessary features** are enabled or installed (e.g., unnecessary ports, services, pages, accounts, or privileges).
- **Default accounts** and their passwords are still enabled and unchanged.
- Error handling reveals stack traces or other **overly informative error messages** to users.

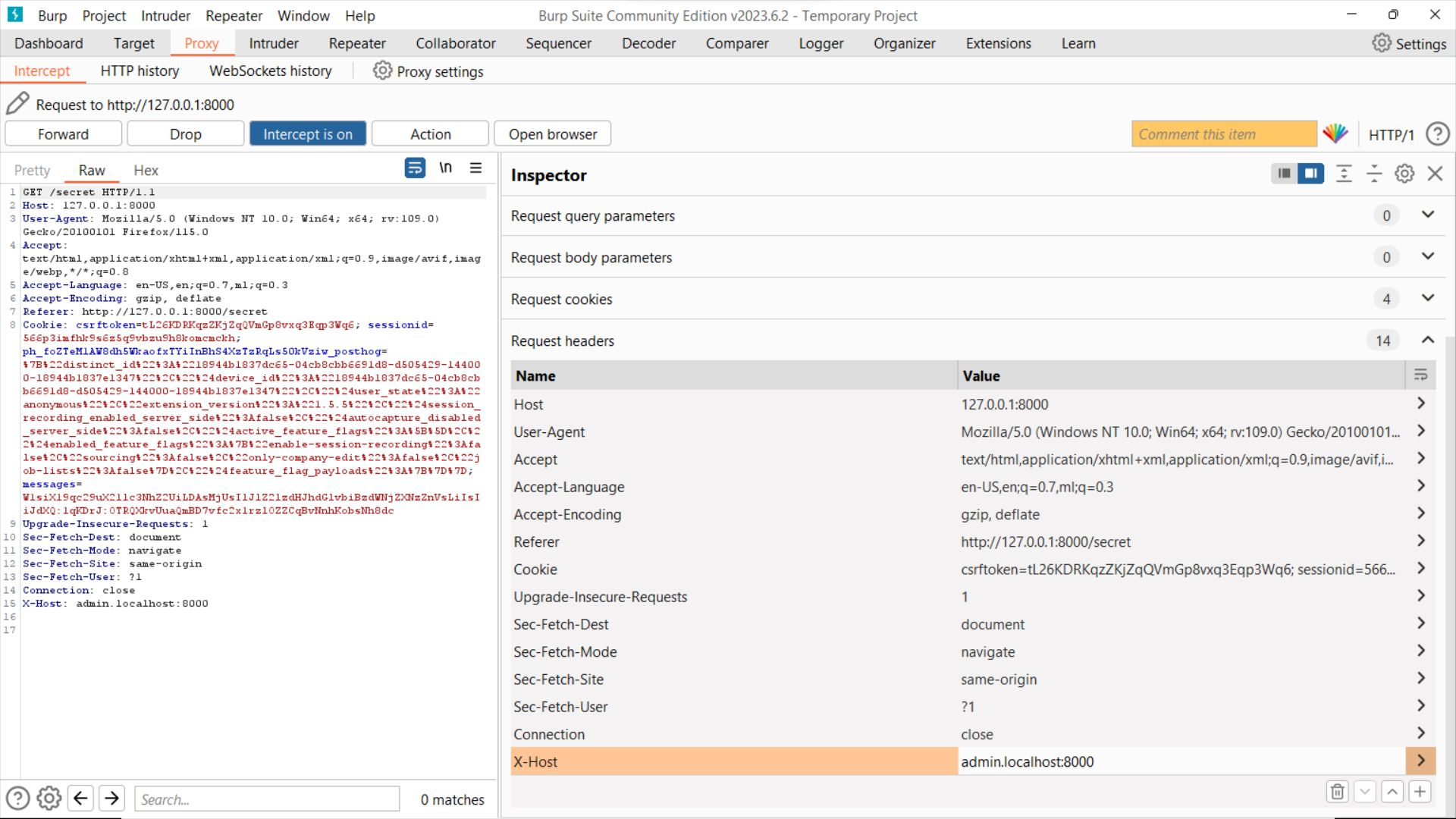
Pygoat Lab 1

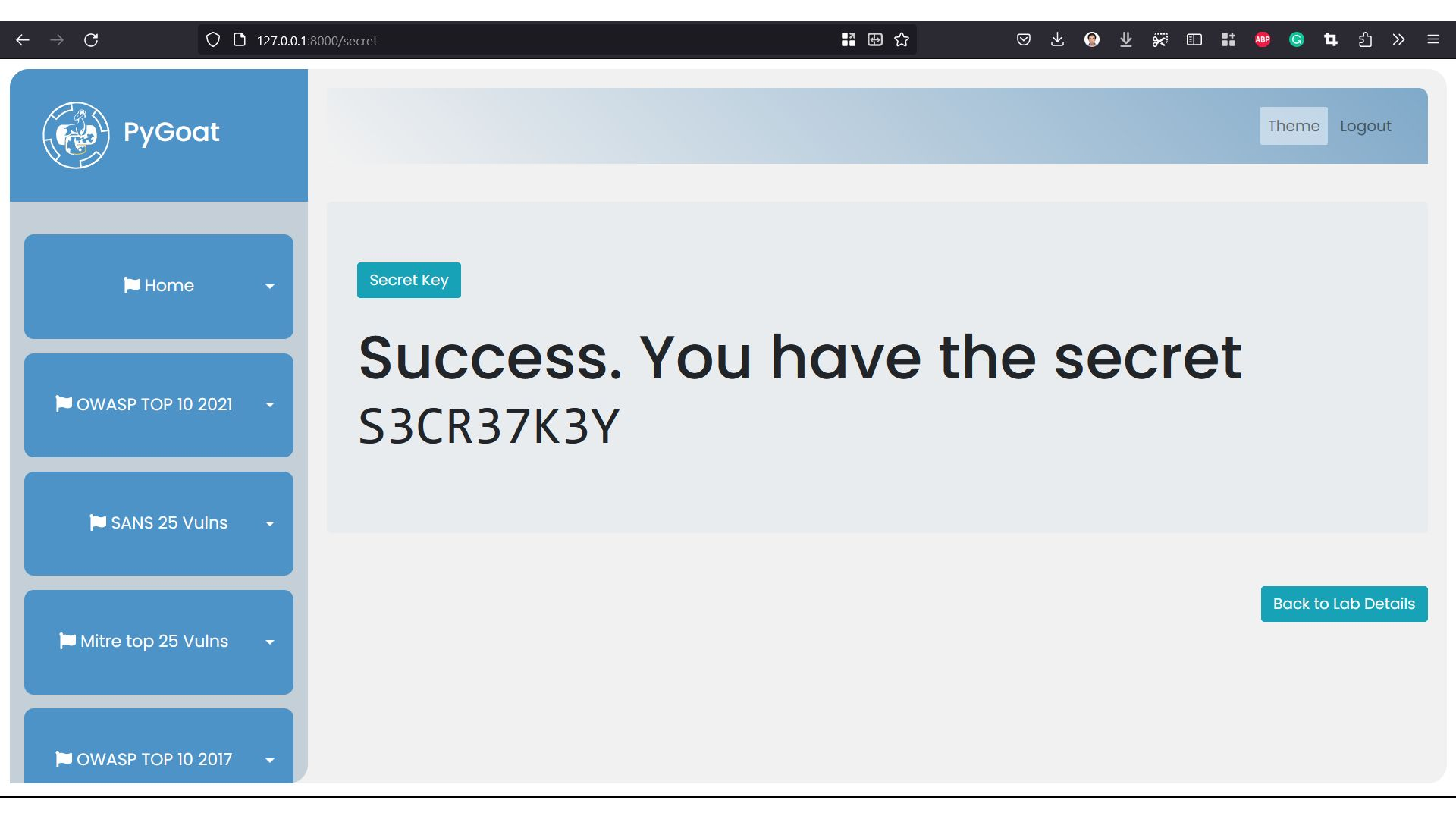
Mission: Get a secret key that is only visible to the admin.

Demo


PyGoat
Theme
Logout
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Secret Key
Only admin.localhost:8000 can access, Your X-Host is None
Back to Lab Details


Burp Project Intruder Repeater Window Help
Burp Suite Community Edition v2023.6.2 - Temporary Project
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Settings
Intercept HTTP history WebSockets history
Proxy settings
Request to http://127.0.0.1:8000
Forward Drop Intercept is on Action Open browser
Comment this item
HTTP/1
Pretty Raw Hex
GET /secret HTTP/1.1
Host: 127.0.0.1:8000
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.7,ml;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:8000/secret
Cookie: csrftoken=tL26KDRKqzZKjZqQVmGp8vxq3Eqp3Wq6; sessionid=566p3imfhk9s6z5q9vbzu9h8romcmckh; ph_foZTeMlAW8dh5WkaofxTYiInBhS4XzTzRqLs50kVziw_posthog=%7B%22distinct_id%22%3A%2218944b1837dc65-04cb8cbb6691d8-d505429-144000-18944b1837e1347%22%2C%22%24device_id%22%3A%2218944b1837dc65-04cb8cbb6691d8-d505429-144000-18944b1837e1347%22%2C%22%24user_state%22%3A%22anonymous%22%2C%22extension_version%22%3A%221.5.5%22%2C%22%24session_recording_enabled_server_side%22%3Afalse%2C%22%24autocapture_disabled_server_side%22%3Afalse%2C%22%24active_feature_flags%22%3A%5B%5D%2C%22%24enabled_feature_flags%22%3A%7B%22enable-session-recording%22%3Afalse%2C%22sourcing%22%3Afalse%2C%22only-company-edit%22%3Afalse%2C%22job-lists%22%3Afalse%7D%2C%22%24feature_flag_payloads%22%3A%7B%7D%7D; messages=WlsiX19qc29uX2llc3NhZ2UiLDAsMjUsI1J1Z2lzdHJhdGlvbiBzdWNjZXNzZnVsLiIsIiJdXQ:1qKDrJ:0TRQXrvUuaQmBD7vfc2x1rz10ZZCqBvNnhKobsNh8dc
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Connection: close
Search...
0 matches
Inspector
Request attributes
2
Protocol HTTP/1 HTTP/2
Name
Value
Method
GET
Path
/secret
Request query parameters
0
Request body parameters
0
Request cookies
4
Request headers
13


Burp Project Intruder Repeater Window Help
Burp Suite Community Edition v2023.6.2 - Temporary Project
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn Settings
Intercept HTTP history WebSockets history Proxy settings
Request to http://127.0.0.1:8000
Forward Drop Intercept is on Action Open browser
Comment this item HTTP/1
Pretty Raw Hex
1 GET /secret HTTP/1.1
2 Host: 127.0.0.1:8000
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.7,ml;q=0.3
6 Accept-Encoding: gzip, deflate
7 Referer: http://127.0.0.1:8000/secret
8 Cookie: csrftoken=tL26KDRKqzZKjZqQVmGp8vxq3Eqp3Wq6; sessionid=566p3imfhkr9s6z5q9vbzu9h8romcmckh; ph_foZTeMlAW8dh5WkaofxTYiInBhS4XzTzRqLs50kVziw_posthog=%7B%22distinct_id%22%3A%2218944b1837dc65-04cb8cbb6691d8-d505429-144000-18944b1837e1347%22%2C%22%24device_id%22%3A%2218944b1837dc65-04cb8cbb6691d8-d505429-144000-18944b1837e1347%22%2C%22%24user_state%22%3A%22anonymous%22%2C%22extension_version%22%3A%221.5.5%22%2C%22%24session_recording_enabled_server_side%22%3Afalse%2C%22%24autocapture_disabled_server_side%22%3Afalse%2C%22%24active_feature_flags%22%3A%5B%5D%2C%22%24enabled_feature_flags%22%3A%7B%22enable-session-recording%22%3Afalse%2C%22sourcing%22%3Afalse%2C%22only-company-edit%22%3Afalse%2C%22job-lists%22%3Afalse%7D%2C%22%24feature_flag_payloads%22%3A%7B%7D%7D; messages=WlsiXl9qc29uX2llc3NhZ2UiLDAsMjUsI1J1Z2lzdHJhdGlvbiBzdWNjZXNzZnVsLiIsIiJdXQ:1qKDrJ:0TRQXrvUuaQmBD7vfc2xlrzl0ZZCqBvNnhKobsNh8dc
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Connection: close
15 X-Host: admin.localhost:8000
16
17
Search... 0 matches
Inspector
Request query parameters 0
Request body parameters 0
Request cookies 4
Request headers 14
Name Value
Host 127.0.0.1:8000
User-Agent Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101...
Accept text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,i...
Accept-Language en-US,en;q=0.7,ml;q=0.3
Accept-Encoding gzip, deflate
Referer http://127.0.0.1:8000/secret
Cookie csrftoken=tL26KDRKqzZKjZqQVmGp8vxq3Eqp3Wq6; sessionid=566...
Upgrade-Insecure-Requests 1
Sec-Fetch-Dest document
Sec-Fetch-Mode navigate
Sec-Fetch-Site same-origin
Sec-Fetch-User ?1
Connection close
X-Host admin.localhost:8000


127.0.0.1:8000/secret
PyGoat
Theme
Logout
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Secret Key
Success. You have the secret
S3CR37K3Y
Back to Lab Details

Mitigation

This case is an example of Security through obscurity.

- Raw inputs from **front end** should **not be trusted**.
- As seen in the lab if access controls are enforced using **predictable, front-end supplied values**, there is a risk of attack. Cookies/token values should be truly random and unpredictable.

In Real-World

Facebook Bug: Delete any Facebook page [source]

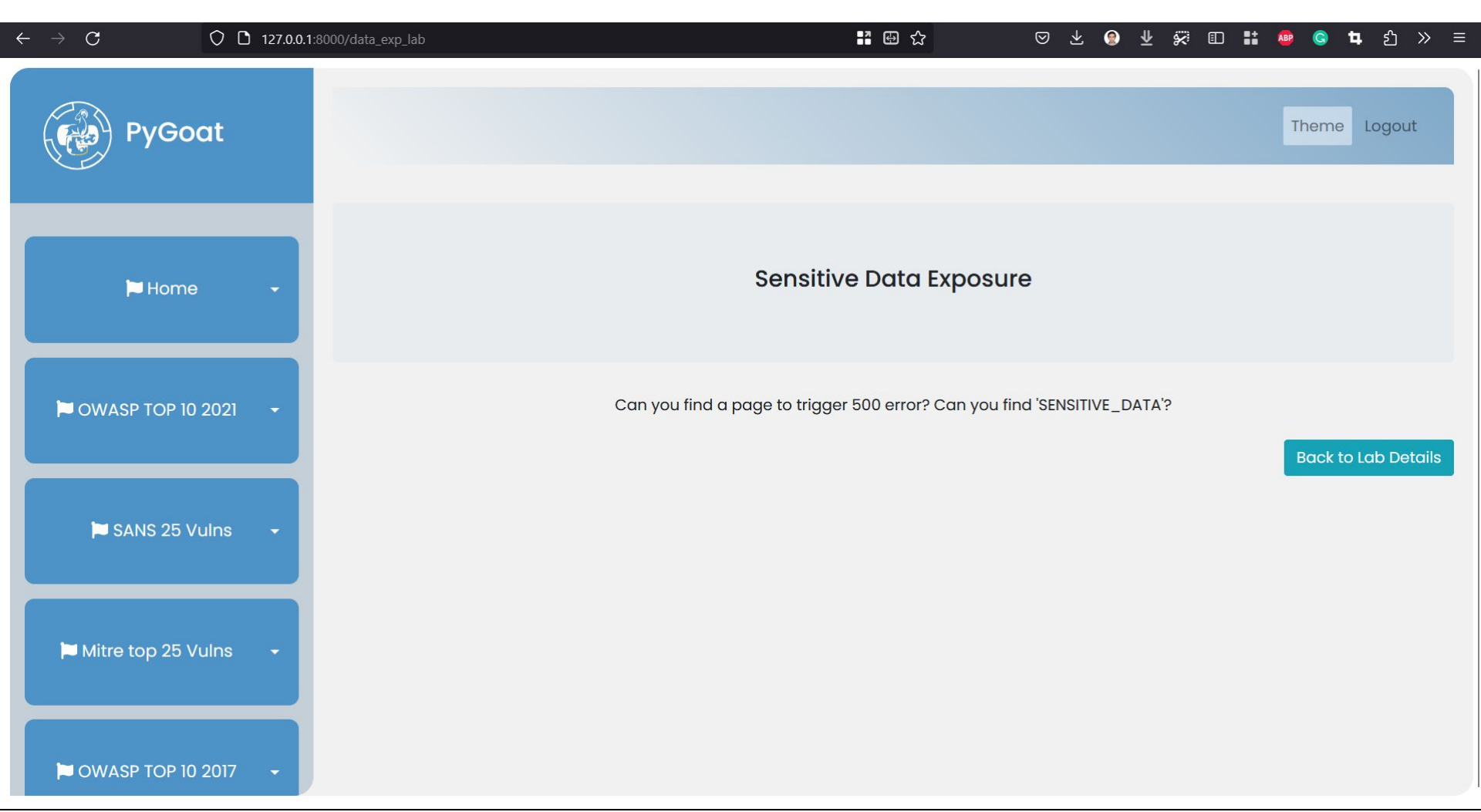
Found by Arun Suresh Kumar (2016)

- The misconfiguration vulnerability could have allowed an attacker to delete any facebook page.
- An attacker could intercept a deletion request (for their own page) sent from the front-end and swap the id to any page id to delete it.
- The users' permission to manage the page was not checked in the backend.

Pygoat Lab 2

Mission: Get a sensitive env variable stored in the application.

Demo


127.0.0.1:8000/data_exp_lab
PyGoat
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Theme
Logout
Sensitive Data Exposure
Can you find a page to trigger 500 error? Can you find 'SENSITIVE_DATA'?
Back to Lab Details

Page not found (404)

Request Method: GET
Request URL: http://127.0.0.1:8000/data_exp_lab/err

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

1. admin/
2. accounts/
3. [name='homepage']
4. xss [name='xss']
5. xssL [name='xss_lab']
6. xssL2 [name='xss_lab2']
7. xssL3 [name='xss_lab3']
8. xssL1 [name='xss_lab']
9. sql [name='sql']
10. sql_lab [name='sql_lab']
11. sql_lab1 [name='sql_lab']
12. insec_des [name='insec_des']
13. insec_des_lab [name='insec_des_lab']
14. xxe [name='xxe']
15. xxe_lab [name='xxe_lab']
16. xxe_see [name='xxe_see']
17. xxe_parse [name='xxe_parse']
18. auth_lab [name='auth_lab']
19. auth_lab/signup [name='auth_lab_signup']
20. auth_lab/login [name='auth_lab_login']
21. auth_lab/logout [name='auth_lab_logout']
22. auth [name='auth_home']
23. ba [name='Broken Access Control']
24. ba_lab [name='Broken Access Control Lab']
25. data_exp [name='data_exp']
26. data_exp_lab [name='data_exp_lab']
27. robots.txt [name='robots.txt']
28. 500error [name='500error']
29. cmd [name='Command Injection']
30. cmd_lab [name='Command Injection Lab']
31. cmd_lab2 [name='Command Injection Lab 2']
32. bau [name='Broken Authe']
33. bau_lab [name='LAB']
34. login_otp [name='OTP Login']
35. otp [name='OTP Verification']
36. sec_mis [name='Security Misconfiguration']
37. sec_mis_lab [name='Security Misconfiguration Lab']
38. sec_mis_lab3 [name='Security Misconfiguration Lab']
39. secret [name='Secret key for A6']
40. a9 [name='A9']
41. a9_lab [name='A9 LAb']
42. a9_lab2 [name='A9 LAb 2']
43. get_version [name='Get Version']
44. a10 [name='A10']
45. a10_lab [name='A10 LAb']

▼ Local vars

Variable	Value
callback	<function error at 0x0000014DA2565160>
name	'The view introduction.views.error'
response	None
self	<django.core.handlers.wsgi.WSGIHandler object at 0x0000014DA1A4B070>

Request information

USER admin

GET No GET data

POST No POST data

FILES No FILES data

COOKIES

Variable	Value
csrftoken	'tL26KDRKqzZKjZqQVmGp8vxq3Eqp3Wq6'
sessionid	'566p3imfhk9s6z5q9vbzu9h8komcmckh'
ph_foZTeM1AW8dh5WkaofxTYiInBhS4XzTzRqLs50kVziw_posthog	'%7B%22distinct_id%22%3A%2218944b1837dc65-04cb8cbb6691d8-d505429-144000-18944b1837e1347%22%2C%22%24device_id%22%3A%2218944b1837dc65-04cb8cbb6691d8-d505429-144000-18944b1837e1347%22%2C%22%24user_state%22%3A%22anonymous%22%2C%22extension_version%22%3A%221.5.5%22%2C%22%24session_recording_enabled_server_side%22%3Afalse%2C%22%24autocapture_disabled_server_side%22%3Afalse%2C%22%24active_feature_flags%22%3A%5B%5D%2C%22%24enabled_feature_flags%22%3A%7B%22enable-session-recording%22%3Afalse%2C%22sourcing%22%3Afalse%2C%22only-company-edit%22%3Afalse%2C%22job-lists%22%3Afalse%7D%2C%22%24feature_flag_payloads%22%3A%7B%7D%7D'
messages	'W1siX19qc29uX211c3NhZ2UiLDAsMjUsIlJlZ2lzdHJhdGlvbiBzdWNjZXNzZnVsLiIsIiJdXQ:1qKDrJ:OTRQXkvUuaQmBD7vfc2x1rzlOZZCqBvNnhKobsNh8dc'
auth_cookie	'eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyIjoibm90X2FkbWluIiwiZXhwIjoxNjg5NjY2MzAwLCJpYXQiOjE2ODk2NjI3MDB9.Ls7y0ISAyQJPeZgt48QR_sZnvq9b0sQt-RfOiH2yeeY'

META

Variable	Value
ALLUSERSPROFILE	'C:\\ProgramData'
ANDROID_SDK_HOME	'C:\\Android'
APPDATA	'C:\\Users\\adars\\AppData\\Roaming'
COMMONPROGRAMFILES	'C:\\Program Files\\Common Files'
COMMONPROGRAMFILES(X86)	'C:\\Program Files (x86)\\Common Files'
COMMONPROGRAMW6432	'C:\\Program Files\\Common Files'
COMPUTERNAME	'WINDOWS'
COMSPEC	'C:\\WINDOWS\\system32\\cmd.exe'
CONTENT_LENGTH	''
CONTENT_TYPE	'text/plain'
CSRF_COOKIE	'tL26KDRKqzZKjZqQVmGp8vxq3Eqp3Wq6'
DJANGO_SETTINGS_MODULE	'pygoat.settings'
DRIVERDATA	'C:\\Windows\\System32\\Drivers\\DriverData'
EFC_5468	'1'
FPS_BROWSER_APP_PROFILE_STRING	'Internet Explorer'

Mitigation

Django debug mode was enabled. This can lead to tracebacks, error messages and env variables being displayed to users.

In a publicly accessible environment, **Debug mode** should be **disabled**.

Security Misconfiguration - Prevention

- **A repeatable hardening process** makes it fast and easy to deploy another environment that is appropriately locked down. This process should be automated to minimize the effort required to set up a new secure environment.
- **Remove or do not install unused features** and frameworks.
- A task to review and update the configurations appropriate to all security notes, updates, and patches as part of the **patch management process**.
- **Review cloud storage permissions** (e.g., S3 bucket permissions).
- An **automated process** to verify the effectiveness of the configurations and settings in all environments.

A6: Vulnerable and Outdated Components

The application might be **vulnerable**:

- If the **software** used **is vulnerable, unsupported, or out of date**. This includes the OS, web/application server, database management system (DBMS), applications, APIs and all components, runtime environments, and libraries.
- If you do not **scan for vulnerabilities** regularly and subscribe to security bulletins related to the components you use.

Pygoat Lab - Objective

Mission: Execute code on the server (RCE) using an outdated dependency.

Demo

Pygoat Lab - Vulnerable Components

This lab helps us to understand why components with known vulnerabilities can be a serious issue.

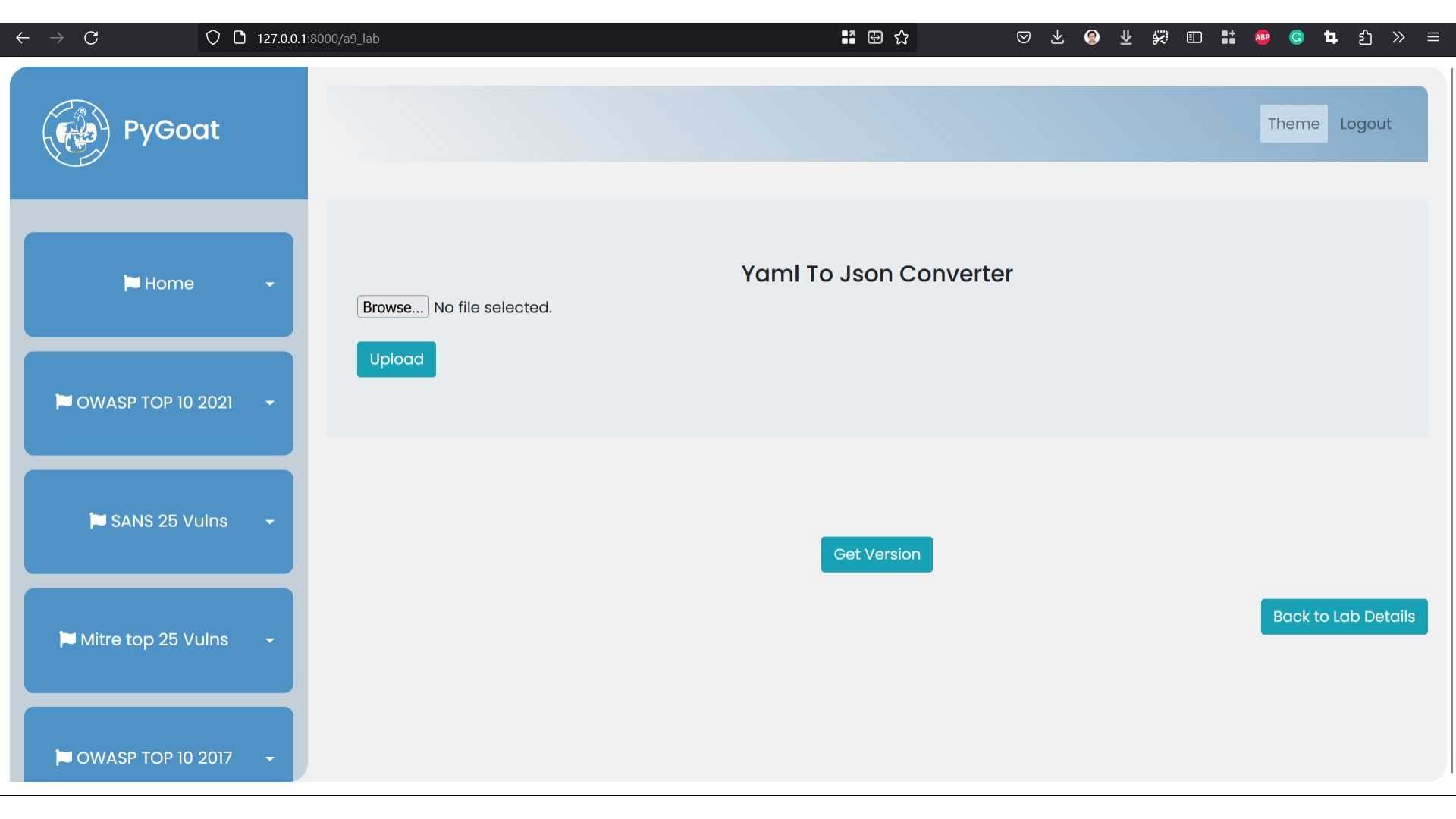
The user on accessing the lab is provided with a feature to convert yaml files into json objects. A yaml file needs to be chosen and uploaded to get the json data.

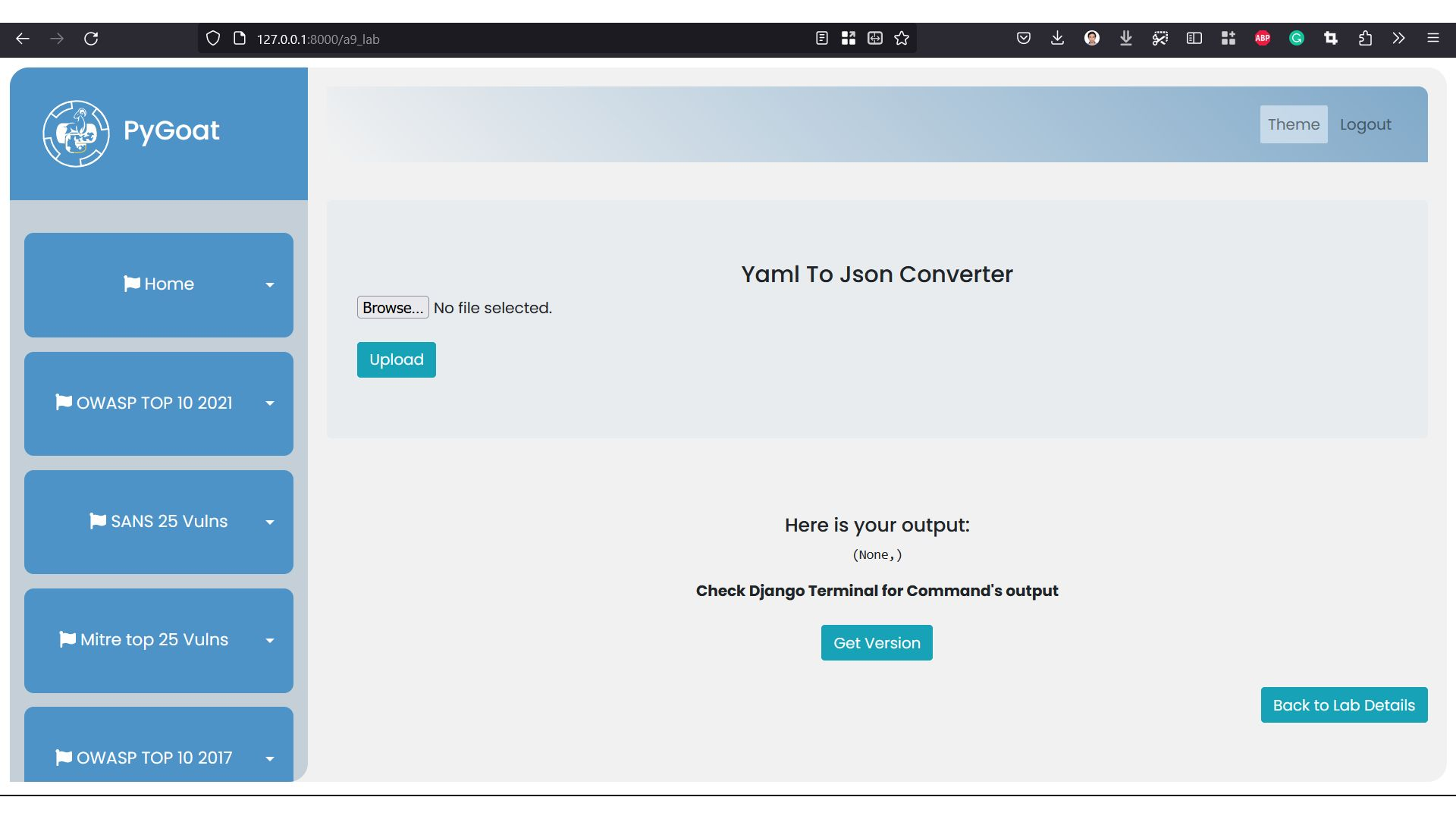
The app uses **pyyaml 5.1** which is vulnerable to code execution.

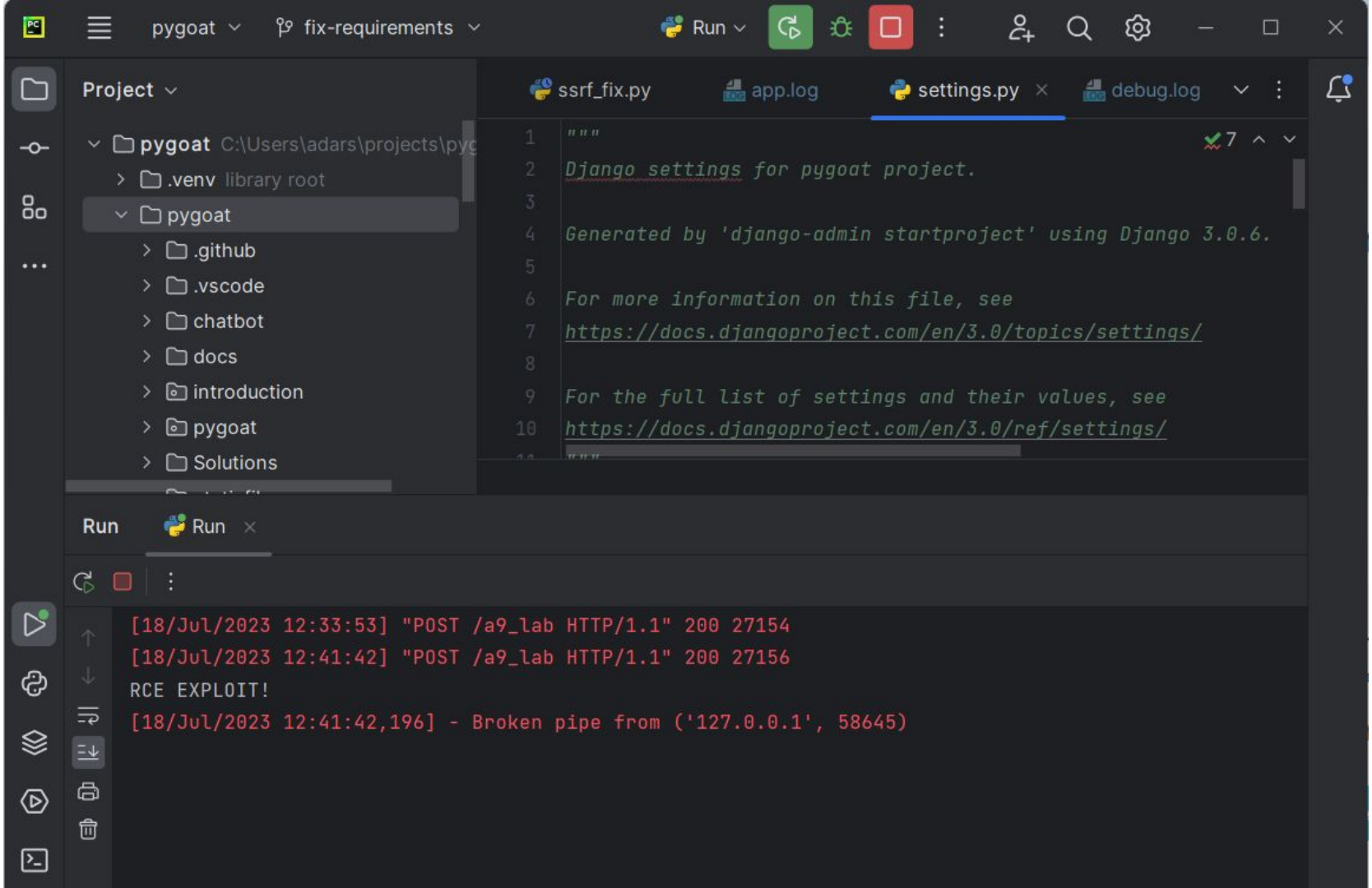
Pygoat Lab

We can craft a yaml as shown below to execute code on the server

```
1
2  !!python/object/new:tuple
3  - !!python/object/new:map
4    - !!python/name:eval
5    - [ print("RCE EXPLOIT!") ]
6
7
```


127.0.0.1:8000/a9_lab
PyGoat
Theme
Logout
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Yaml To Json Converter
Browse...
No file selected.
Upload
Get Version
Back to Lab Details


127.0.0.1:8000/a9_lab
PyGoat
Theme
Logout
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Yaml To Json Converter
Browse...
No file selected.
Upload
Here is your output:
(None,)
Check Django Terminal for Command's output
Get Version
Back to Lab Details


pygoat
fix-requirements
Run
Project
pygoat C:\Users\adars\projects\pyg
.venv library root
pygoat
.github
.vscode
chatbot
docs
introduction
pygoat
Solutions
ssrf_fix.py
app.log
settings.py
debug.log
"""
Django settings for pygoat project.
Generated by 'django-admin startproject' using Django 3.0.6.
For more information on this file, see
https://docs.djangoproject.com/en/3.0/topics/settings/
For the full list of settings and their values, see
https://docs.djangoproject.com/en/3.0/ref/settings/
"""
7
Run
Run
[18/Jul/2023 12:33:53] "POST /a9_lab HTTP/1.1" 200 27154
[18/Jul/2023 12:41:42] "POST /a9_lab HTTP/1.1" 200 27156
RCE EXPLOIT!
[18/Jul/2023 12:41:42,196] - Broken pipe from ('127.0.0.1', 58645)

Mitigation

- **Remove** unused dependencies, unnecessary features, components, files, and documentation.
- Continuously inventory the versions of both client-side and server-side components (e.g., frameworks, libraries) and their dependencies using tools like versions, OWASP Dependency Check, retire.js, etc. **Continuously monitor** sources like Common Vulnerability and Exposures (CVE) and National Vulnerability Database (NVD) for vulnerabilities in the components.
- Only obtain components from **official sources over secure links**. Prefer signed packages to reduce the chance of including a modified, malicious component (See A08:2021-Software and Data Integrity Failures).

A7: Identification and Authentication Failures

There are times when the way applications **handle passwords and user logins** is done **incorrectly**.

This can let attackers get access to passwords, keys, or tokens, or take advantage of other mistakes in how the app works to pretend to be someone else, either for a little while or forever.

Pygoat Lab - Objective

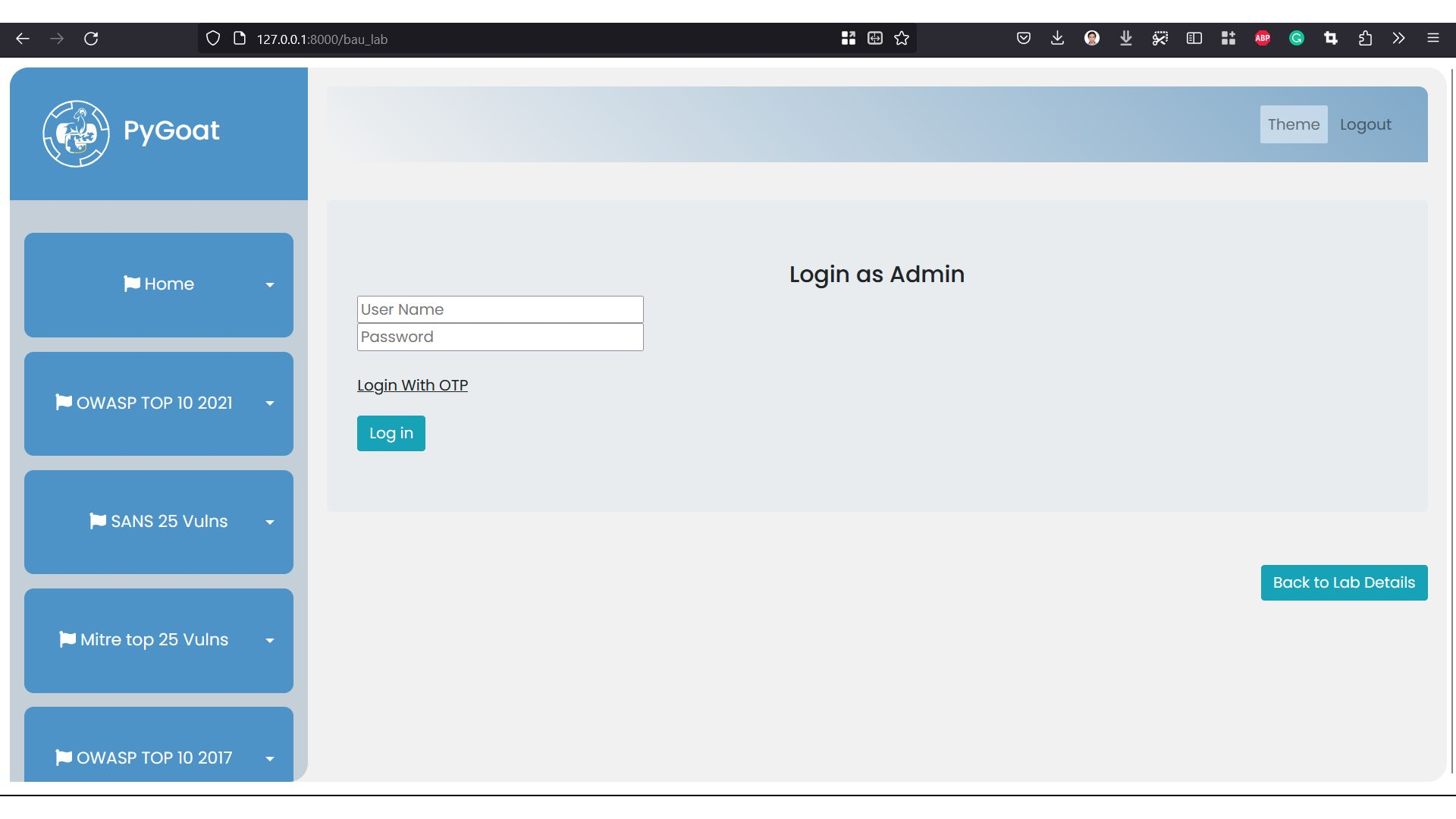
Mission: The aim of this lab is to login as admin.

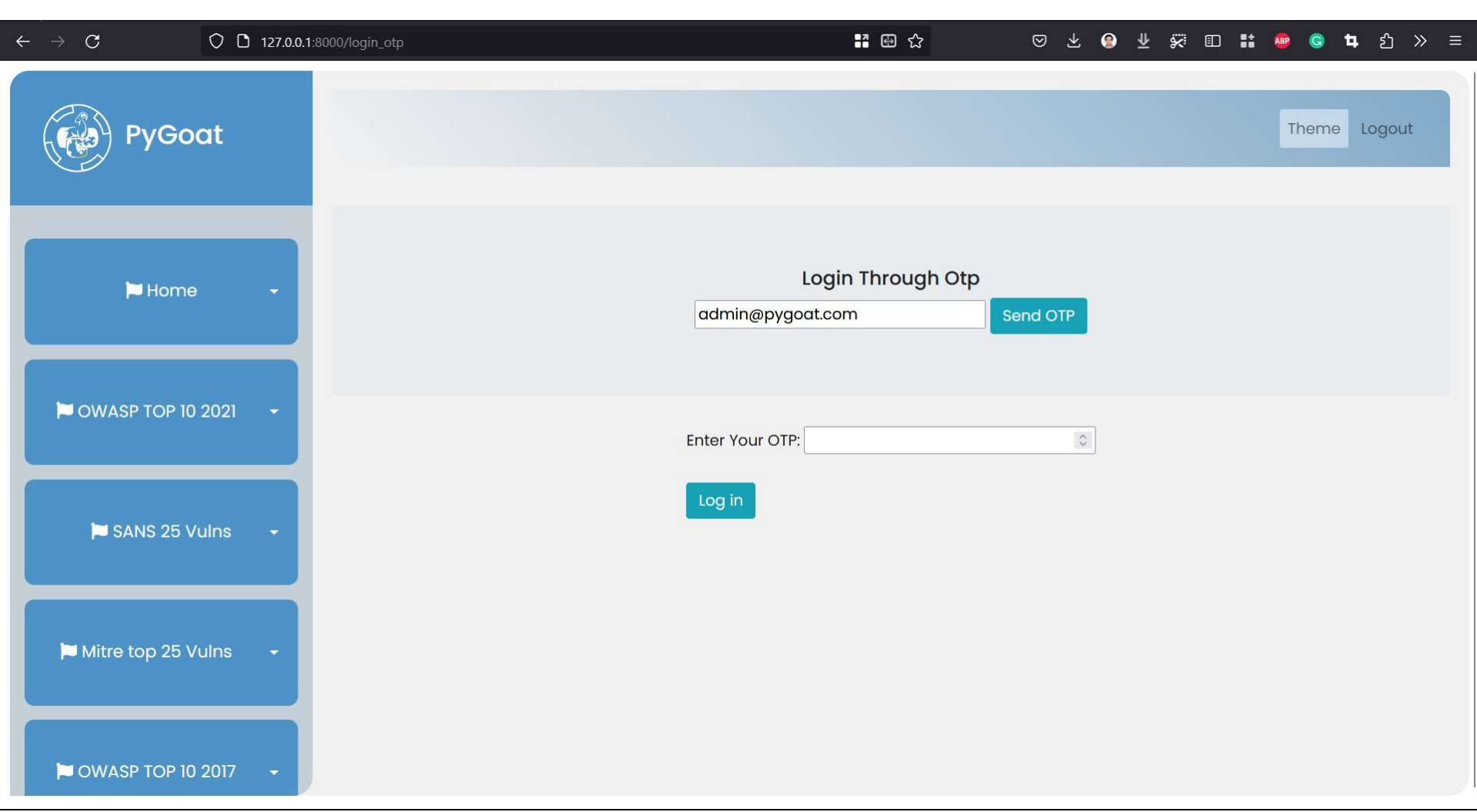
Demo

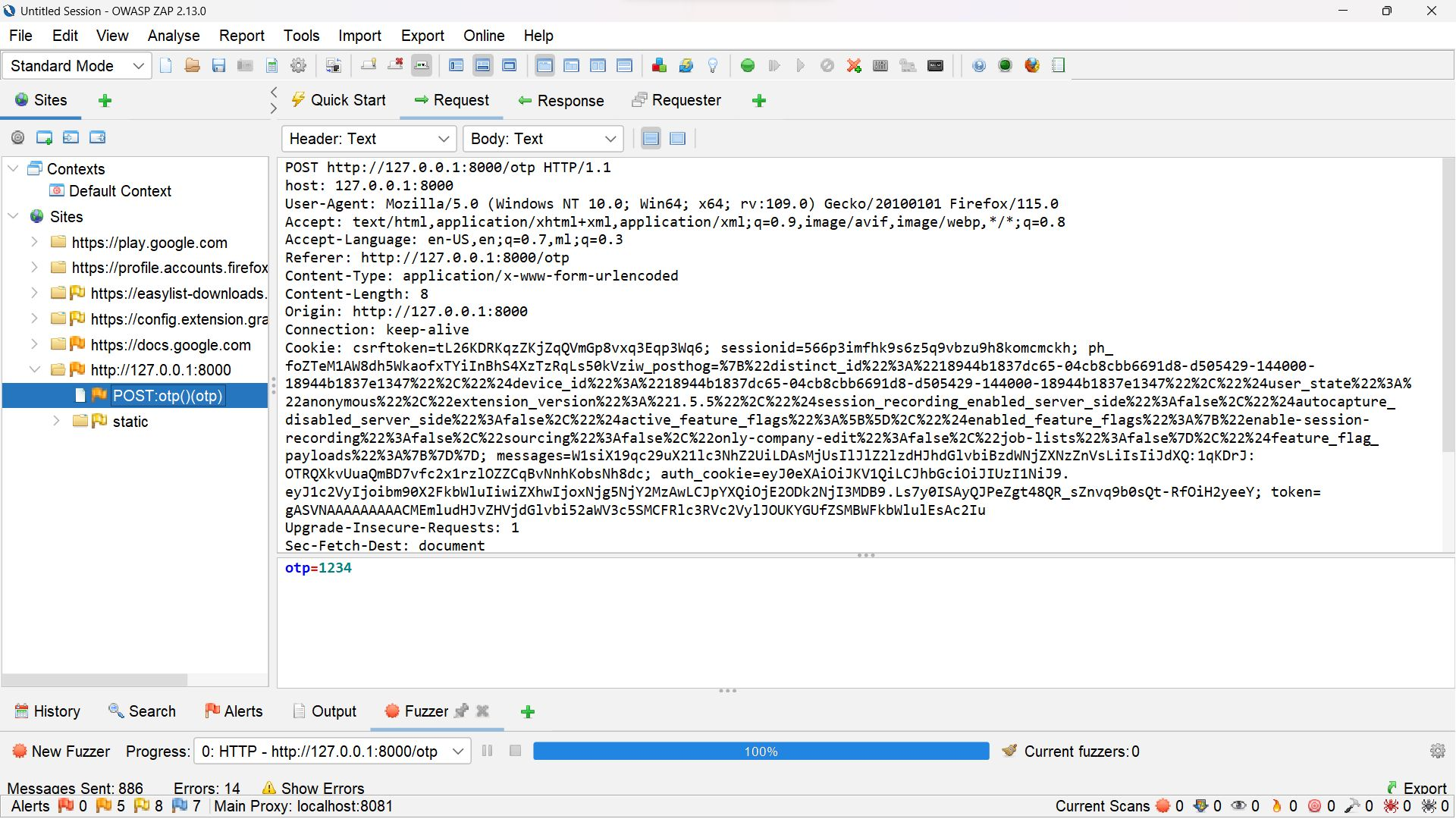
Pygoat Lab

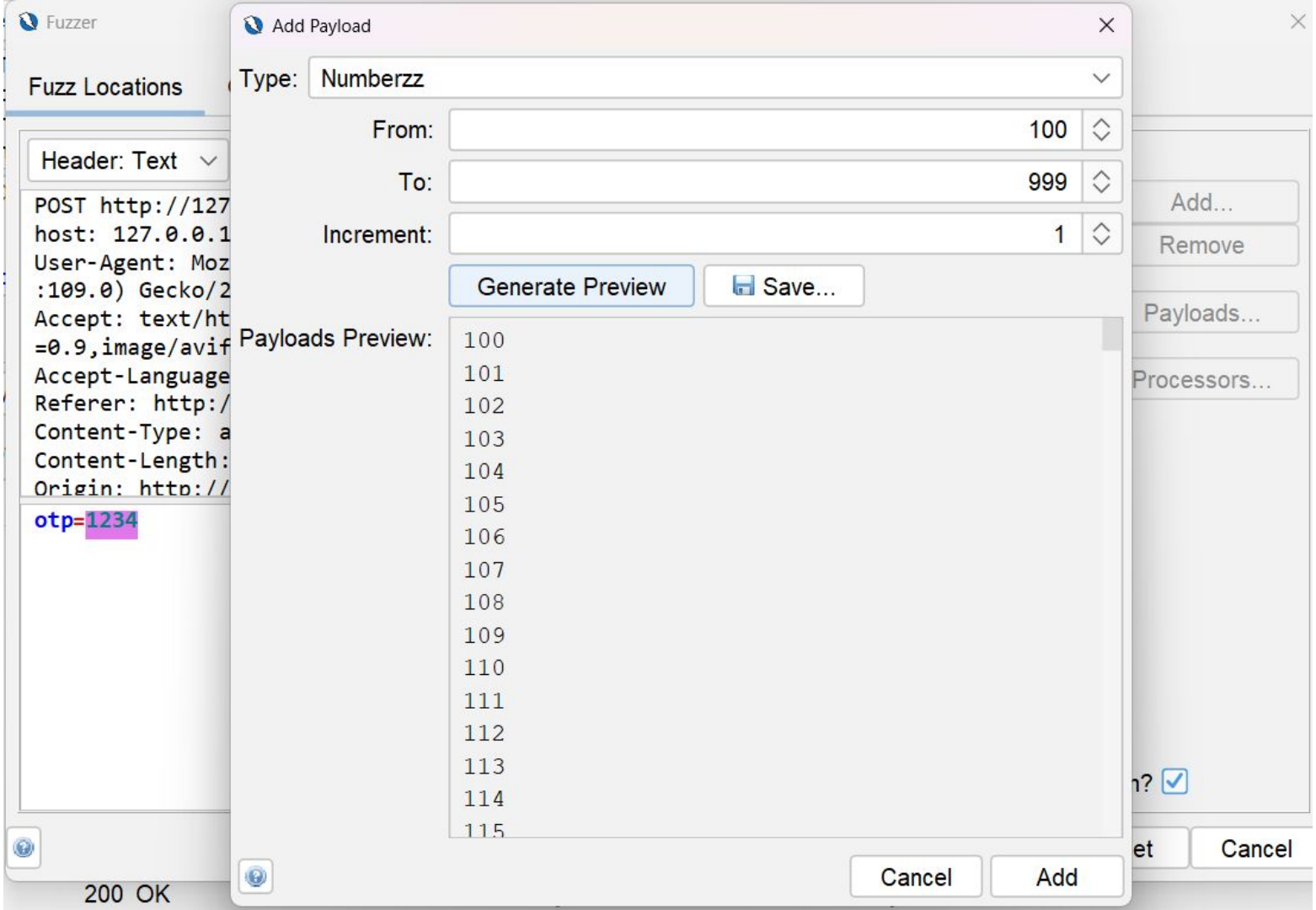
We need to exploit the lack of rate limiting feature in the otp verification flow.

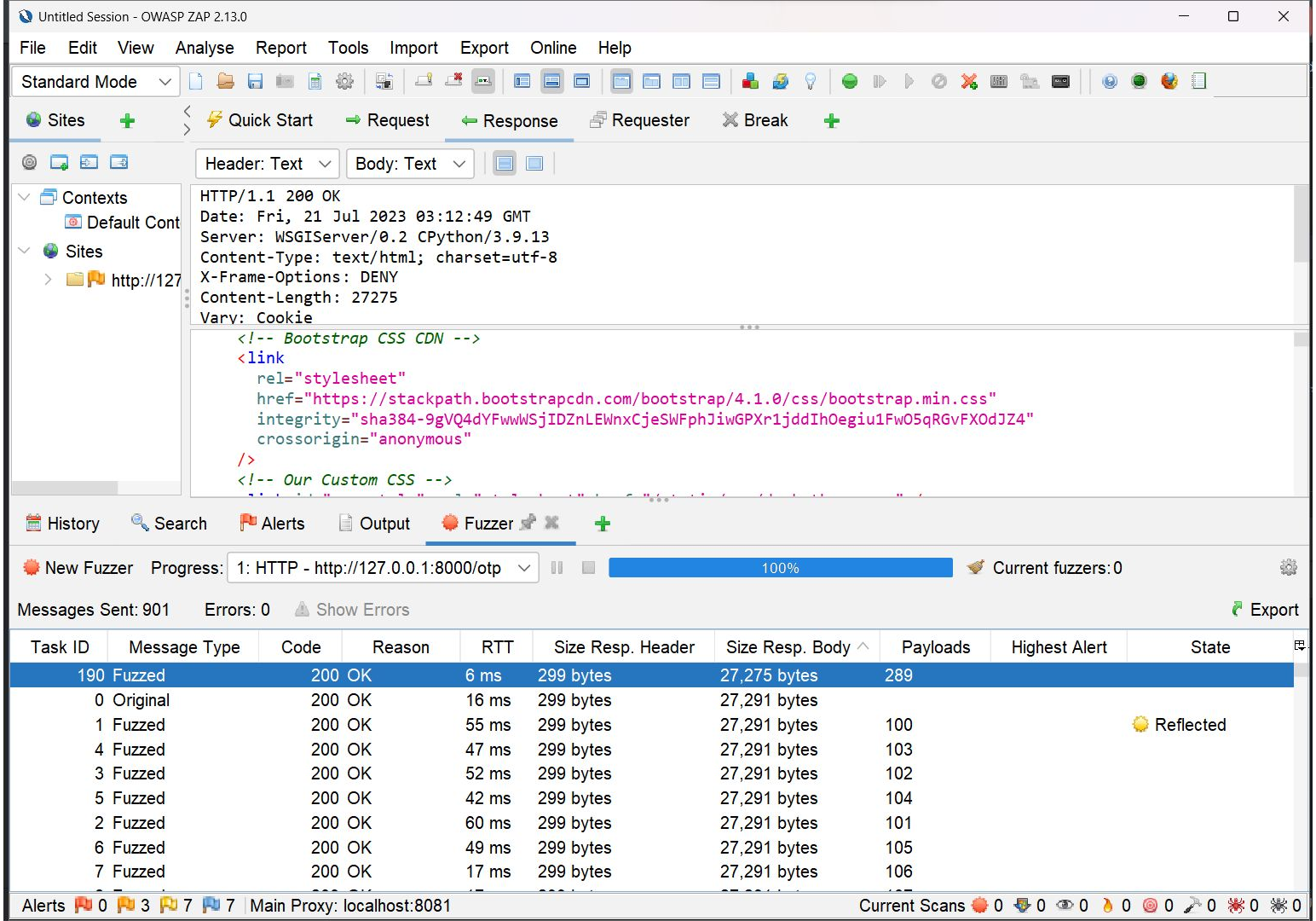
You can see that the otp is only of 3 digit and the application doesn't have any captcha (To disallow any automated scripts or bots) or any restrictions on the number of tries for the otp.

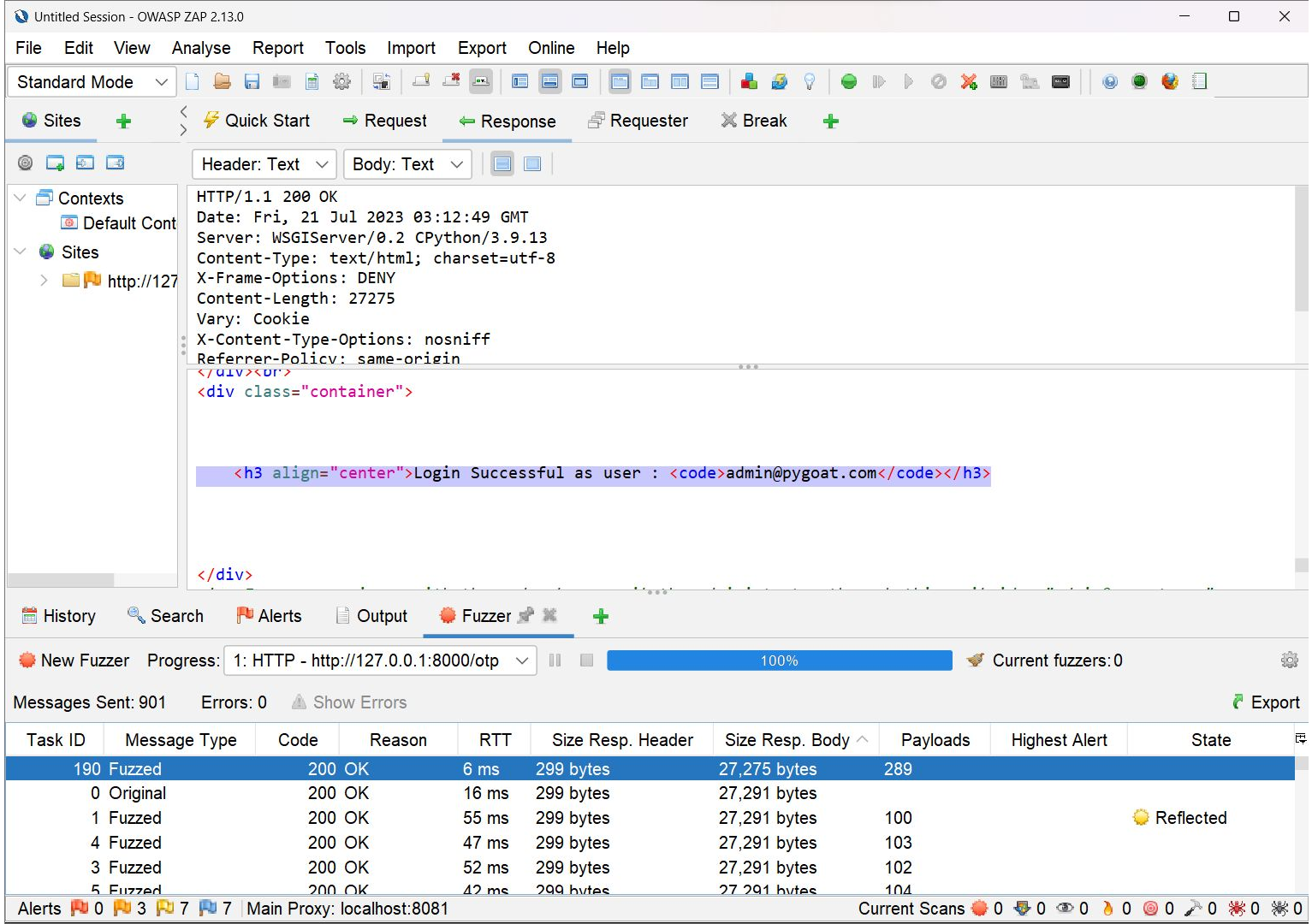

127.0.0.1:8000/bau_lab
PyGoat
Theme
Logout
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Login as Admin
User Name
Password
Login With OTP
Log in
Back to Lab Details


127.0.0.1:8000/login_otp
PyGoat
Theme
Logout
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Login Through Otp
admin@pygoat.com
Send OTP
Enter Your OTP:
Log in


Untitled Session - OWASP ZAP 2.13.0
File Edit View Analyse Report Tools Import Export Online Help
Standard Mode
Sites
Quick Start
Request
Response
Requester
Header: Text
Body: Text
Contexts
Default Context
Sites
https://play.google.com
https://profile.accounts.firefox
https://easylist-downloads.
https://config.extension.gra
https://docs.google.com
http://127.0.0.1:8000
POST:otp()(otp)
static
POST http://127.0.0.1:8000/otp HTTP/1.1
host: 127.0.0.1:8000
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.7,ml;q=0.3
Referer: http://127.0.0.1:8000/otp
Content-Type: application/x-www-form-urlencoded
Content-Length: 8
Origin: http://127.0.0.1:8000
Connection: keep-alive
Cookie: csrftoken=tL26KDRKqzZKjZqQVmGp8vxq3Eqp3Wq6; sessionid=566p3imfhk9s6z5q9vbzu9h8komcmckh; ph_
foZTeM1AW8dh5WkaofxTYiInBhS4XzTzRqLs50kVziw_posthog=%7B%22distinct_id%22%3A%2218944b1837dc65-04cb8cbb6691d8-d505429-144000-
18944b1837e1347%22%2C%22%24device_id%22%3A%2218944b1837dc65-04cb8cbb6691d8-d505429-144000-18944b1837e1347%22%2C%22%24user_state%22%3A%
22anonymous%22%2C%22extension_version%22%3A%221.5.5%22%2C%22%24session_recording_enabled_server_side%22%3Afalse%2C%22%24autocapture_
disabled_server_side%22%3Afalse%2C%22%24active_feature_flags%22%3A%5B%5D%2C%22%24enabled_feature_flags%22%3A%7B%22enable-session-
recording%22%3Afalse%2C%22sourcing%22%3Afalse%2C%22only-company-edit%22%3Afalse%2C%22job-lists%22%3Afalse%7D%2C%22%24feature_flag_
payloads%22%3A%7B%7D%7D; messages=W1siX19qc29uX21lc3NhZ2UiLDAsMjUsIlJlZ2lzdHJhdGlvbiBzdWNjZXNzZnVsLiIsIiJdXQ:1qKDrJ:
OTRQXkvUuaQmBD7vfc2x1rzlOZZCqBvNnhKobsNh8dc; auth_cookie=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.
eyJ1c2VyIjoibm90X2FkbWluIiwiZXhwIjoxNjg5NjY2MzAwLCJpYXQiOjE2ODk2NjI3MDB9.Ls7y0ISAyQJPeZgt48QR_sZnvq9b0sQt-RfOiH2yeeY; token=
gASVNAAAAAAAAACMEmludHJvZHVjdGlvbi52aWV3c5SMCFRlc3RVc2VylJOUKYGUfZSMBWFkbWlulEsAc2Iu
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
otp=1234
History
Search
Alerts
Output
Fuzzer
New Fuzzer
Progress:
0: HTTP - http://127.0.0.1:8000/otp
100%
Current fuzzers: 0
Messages Sent: 886
Errors: 14
Show Errors
Export
Alerts 0 5 8 7
Main Proxy: localhost:8081
Current Scans 0 0 0 0 0 0 0 0


Fuzzer
Fuzz Locations
Header: Text
POST http://127
host: 127.0.0.1
User-Agent: Moz
:109.0) Gecko/2
Accept: text/ht
=0.9,image/avif
Accept-Language
Referer: http:/
Content-Type: a
Content-Length:
Origin: http://
otp=1234
200 OK
Add Payload
Type: Numberzz
From: 100
To: 999
Increment: 1
Generate Preview
Save...
Payloads Preview:
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
Cancel
Add
Add...
Remove
Payloads...
Processors...
Cancel


Untitled Session - OWASP ZAP 2.13.0
File Edit View Analyse Report Tools Import Export Online Help
Standard Mode
Sites
Quick Start
Request
Response
Requester
Break
Header: Text
Body: Text
Contexts
Default Cont
Sites
http://127
HTTP/1.1 200 OK
Date: Fri, 21 Jul 2023 03:12:49 GMT
Server: WSGIServer/0.2 CPython/3.9.13
Content-Type: text/html; charset=utf-8
X-Frame-Options: DENY
Content-Length: 27275
Vary: Cookie
<!-- Bootstrap CSS CDN -->
<link
rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css"
integrity="sha384-9gVQ4dYFwwWSjIDZnLEWnxCjeSWFphJiwGPXr1jddIhOegiu1FwO5qRGvFXOdJZ4"
crossorigin="anonymous"
/>
<!-- Our Custom CSS -->
History
Search
Alerts
Output
Fuzzer
New Fuzzer
Progress:
1: HTTP - http://127.0.0.1:8000/otp
100%
Current fuzzers: 0
Messages Sent: 901
Errors: 0
Show Errors
Export
Task ID
Message Type
Code
Reason
RTT
Size Resp. Header
Size Resp. Body
Payloads
Highest Alert
State
190 Fuzzed 200 OK 6 ms 299 bytes 27,275 bytes 289
0 Original 200 OK 16 ms 299 bytes 27,291 bytes
1 Fuzzed 200 OK 55 ms 299 bytes 27,291 bytes 100 Reflected
4 Fuzzed 200 OK 47 ms 299 bytes 27,291 bytes 103
3 Fuzzed 200 OK 52 ms 299 bytes 27,291 bytes 102
5 Fuzzed 200 OK 42 ms 299 bytes 27,291 bytes 104
2 Fuzzed 200 OK 60 ms 299 bytes 27,291 bytes 101
6 Fuzzed 200 OK 49 ms 299 bytes 27,291 bytes 105
7 Fuzzed 200 OK 17 ms 299 bytes 27,291 bytes 106
Alerts 0 3 7 7
Main Proxy: localhost:8081
Current Scans 0 0 0 0 0 0 0 0


Untitled Session - OWASP ZAP 2.13.0
File Edit View Analyse Report Tools Import Export Online Help
Standard Mode
Sites
Quick Start
Request
Response
Requester
Break
Header: Text
Body: Text
Contexts
Default Cont
Sites
http://127
HTTP/1.1 200 OK
Date: Fri, 21 Jul 2023 03:12:49 GMT
Server: WSGIServer/0.2 CPython/3.9.13
Content-Type: text/html; charset=utf-8
X-Frame-Options: DENY
Content-Length: 27275
Vary: Cookie
X-Content-Type-Options: nosniff
Referrer-Policy: same-origin

<h3 align="center">Login Successful as user : <code>admin@pygoat.com</code></h3>

History
Search
Alerts
Output
Fuzzer
New Fuzzer
Progress:
1: HTTP - http://127.0.0.1:8000/otp
100%
Current fuzzers: 0
Messages Sent: 901
Errors: 0
Show Errors
Export
Task ID
Message Type
Code
Reason
RTT
Size Resp. Header
Size Resp. Body
Payloads
Highest Alert
State
190 Fuzzed 200 OK 6 ms 299 bytes 27,275 bytes 289
0 Original 200 OK 16 ms 299 bytes 27,291 bytes
1 Fuzzed 200 OK 55 ms 299 bytes 27,291 bytes 100 Reflected
4 Fuzzed 200 OK 47 ms 299 bytes 27,291 bytes 103
3 Fuzzed 200 OK 52 ms 299 bytes 27,291 bytes 102
5 Fuzzed 200 OK 42 ms 299 bytes 27,291 bytes 104
Alerts 0 3 7 7
Main Proxy: localhost:8081
Current Scans 0 0 0 0 0 0 0 0

Mitigation

- Where possible, implement **multi-factor authentication** to prevent automated credential stuffing, brute force, and stolen credential reuse attacks.
- Do not ship or deploy with any **default credentials,** particularly for admin users.
- Implement **weak password checks**, such as testing new or changed passwords against the top 10,000 worst passwords list.
- Limit or increasingly **delay failed login attempts**, but be careful not to create a denial of service scenario. Log all failures and alert administrators when credential stuffing, brute force, or other attacks are detected.

In Real World

Facebook Password Reset (2016)

Source: Getting $15,000 bounty for a bug that let me hack into any of Facebook's 2 billion accounts - Anand Prakash

- Facebook used 6 digits password recovery codes
- Brute force was blocked on Facebook.com root domain after a few attempts
- But there was no limits on the number of attempts on beta.facebook.com domain.

References

OWASP Top 10: https://owasp.org/Top10/

Django Security Doc:

https://docs.djangoproject.com/en/5.1/topics/security/

Pygoat Project and Demos:

https://github.com/adeyosemanputra/pygoat

https://github.com/adarshdigievo/pygoat [Fork]

https://www.youtube.com/watch?v=k_C5bNF1VGs

Learning resources

Hacksplaining: https://www.hacksplaining.com/owasp

Web Security Academy: https://portswigger.net/web-security

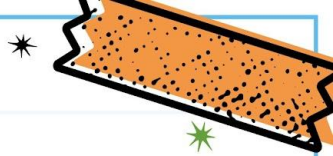
Owasp Cheatsheets: https://cheatsheetseries.owasp.org/IndexTopTen.html

2024 djangocon.us DURHAM

Thank You

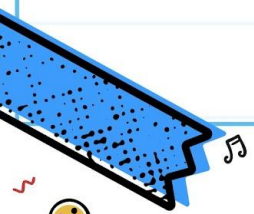
Get the talk materials & connect with me

linkhq.co/adarsh



Bonus Slides

Remaining OWASP top 10 vulnerabilities

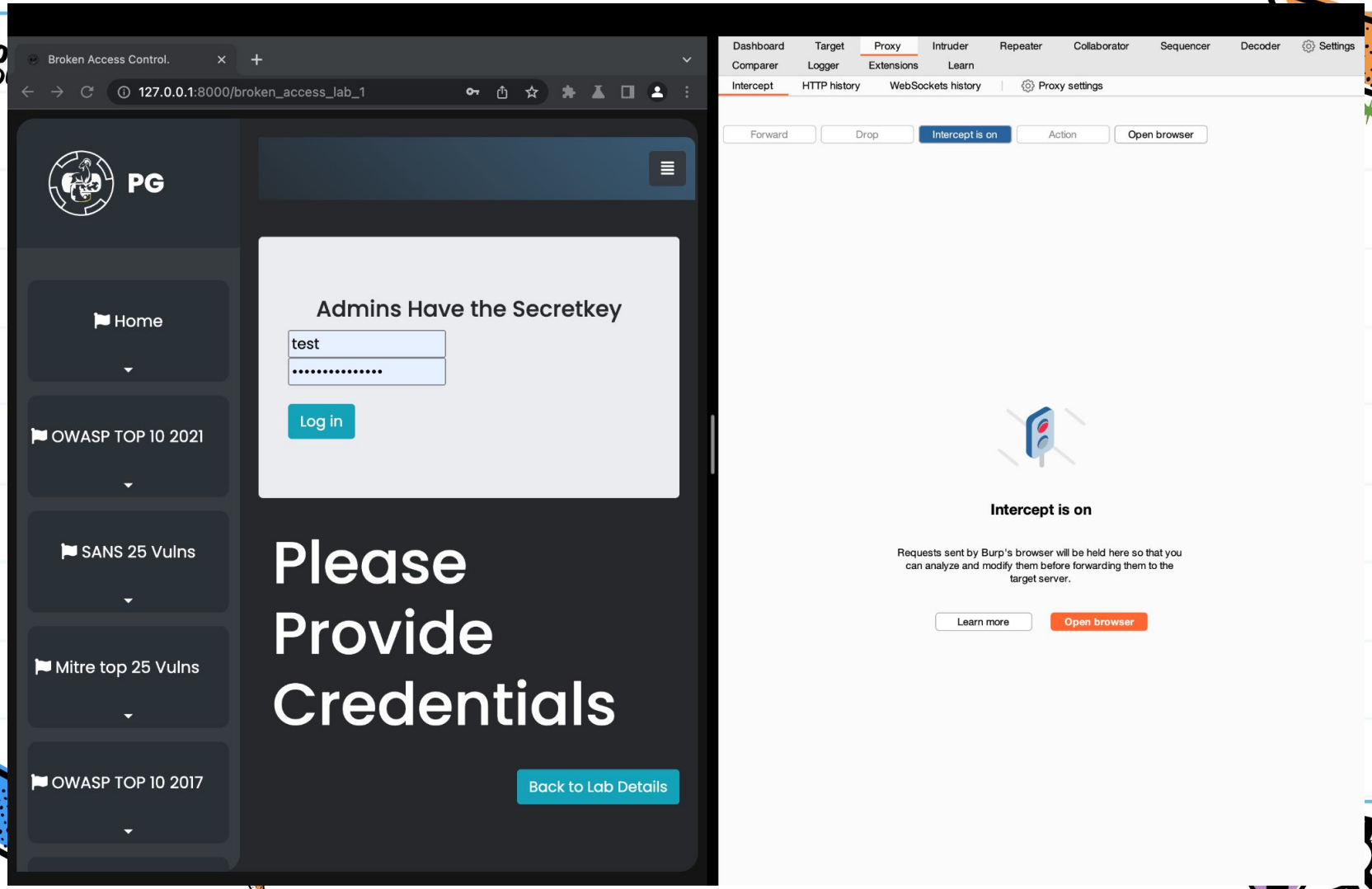


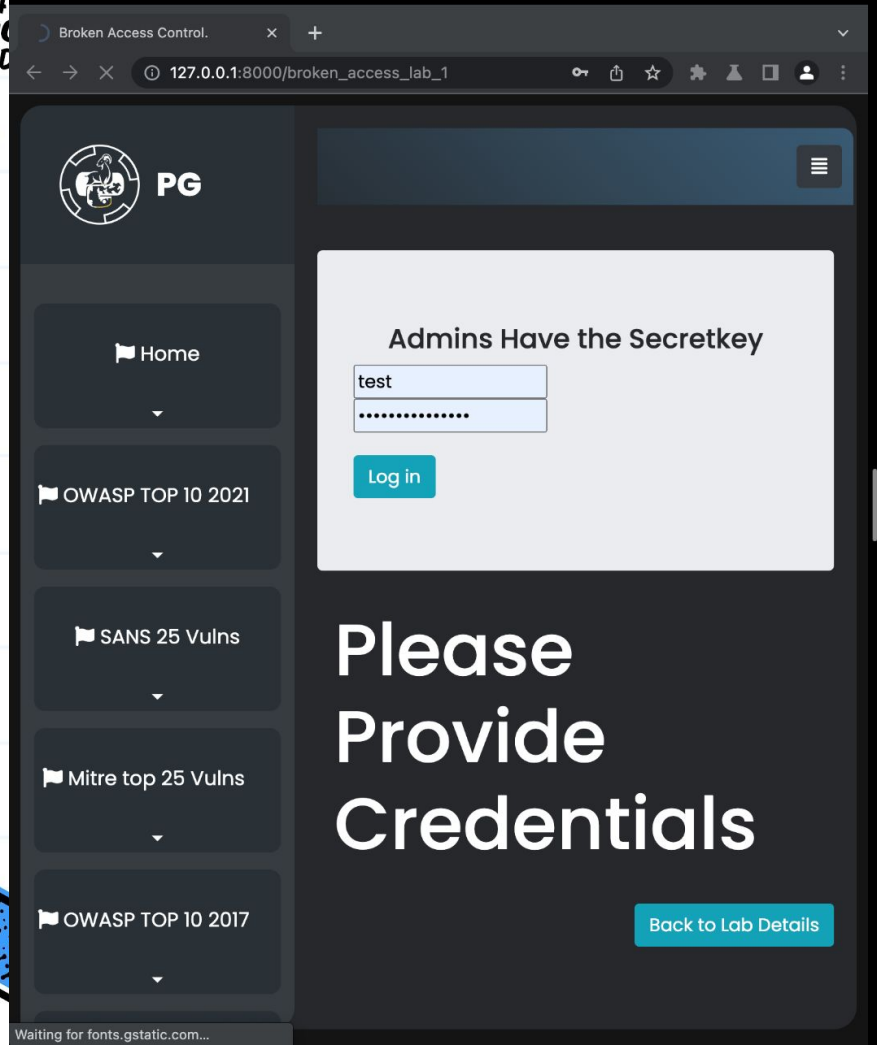
A1 Broken Access Control

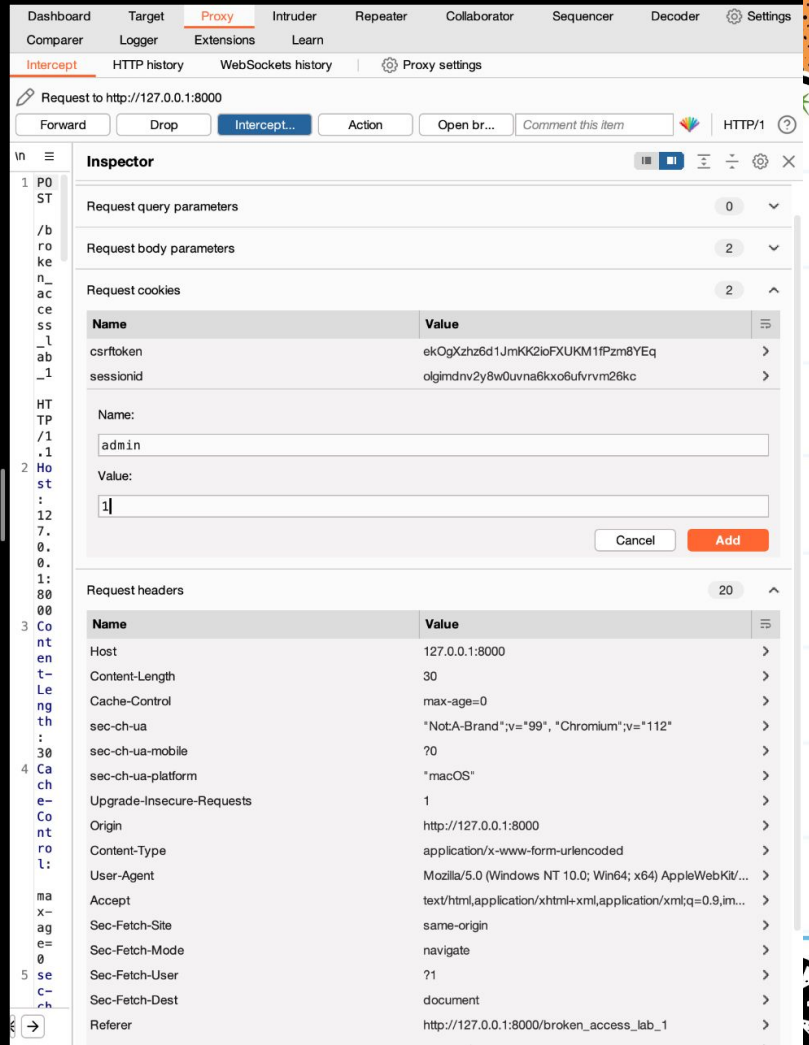
Access control, sometimes called authorization, is how a web application grants **access** to content and functions to some users and not others.

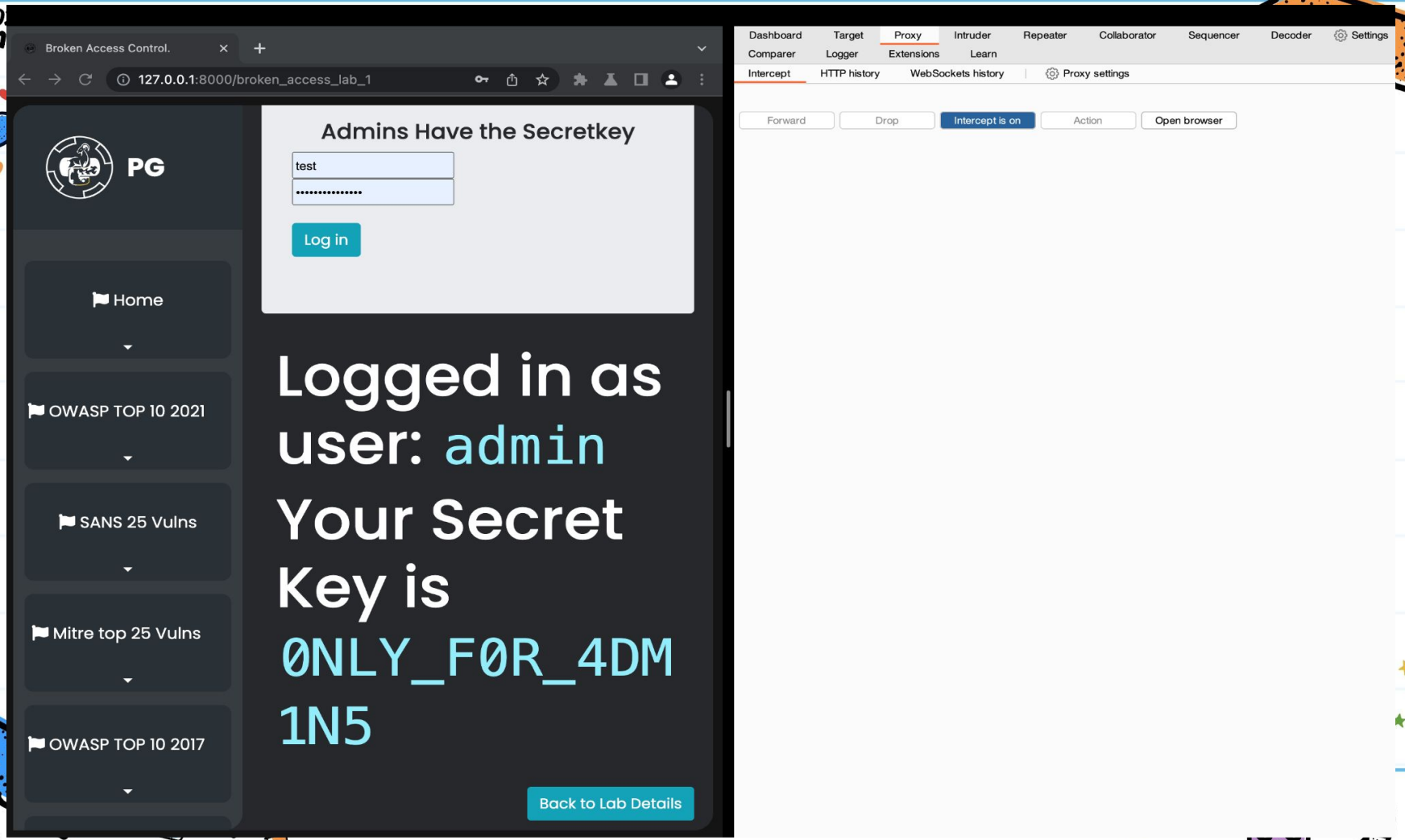
These checks are performed after authentication, and govern what **‘authorized’** users are **allowed to do**.

PyGoat Lab 1


Broken Access Control.
127.0.0.1:8000/broken_access_lab_1
PG
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Admins Have the Secretkey
test
Log in
Please Provide Credentials
Back to Lab Details
Dashboard
Target
Proxy
Intruder
Repeater
Collaborator
Sequencer
Decoder
Settings
Comparer
Logger
Extensions
Learn
Intercept
HTTP history
WebSockets history
Proxy settings
Forward
Drop
Intercept is on
Action
Open browser
Intercept is on
Requests sent by Burp's browser will be held here so that you can analyze and modify them before forwarding them to the target server.
Learn more
Open browser


Broken Access Control.
127.0.0.1:8000/broken_access_lab_1
PG
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Admins Have the Secretkey
test
Log in
Please Provide Credentials
Back to Lab Details
Waiting for fonts.gstatic.com...


Dashboard
Target
Proxy
Intruder
Repeater
Collaborator
Sequencer
Decoder
Settings
Comparer
Logger
Extensions
Learn
Intercept
HTTP history
WebSockets history
Proxy settings
Request to http://127.0.0.1:8000
Forward
Drop
Intercept...
Action
Open br...
Comment this item
HTTP/1
Inspector
Request query parameters 0
Request body parameters 2
Request cookies 2
Name
Value
csrftoken
ekOgXzhz6d1JmKK2ioFXUKM1fPzm8YEq
sessionid
olgimdnv2y8w0uvna6kxo6ufvrvm26kc
Name:
admin
Value:
1
Cancel
Add
Request headers 20
Host
127.0.0.1:8000
Content-Length
30
Cache-Control
max-age=0
sec-ch-ua
"Not:A-Brand";v="99", "Chromium";v="112"
sec-ch-ua-mobile
?0
sec-ch-ua-platform
"macOS"
Upgrade-Insecure-Requests
1
Origin
http://127.0.0.1:8000
Content-Type
application/x-www-form-urlencoded
User-Agent
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/...
Accept
text/html,application/xhtml+xml,application/xml;q=0.9,im...
Sec-Fetch-Site
same-origin
Sec-Fetch-Mode
navigate
Sec-Fetch-User
?1
Sec-Fetch-Dest
document
Referer
http://127.0.0.1:8000/broken_access_lab_1


Broken Access Control.
127.0.0.1:8000/broken_access_lab_1
PG
Admins Have the Secretkey
test
Log in
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Logged in as user: admin
Your Secret Key is
0NLY_F0R_4DM1N5
Back to Lab Details
Dashboard
Target
Proxy
Intruder
Repeater
Collaborator
Sequencer
Decoder
Settings
Comparer
Logger
Extensions
Learn
Intercept
HTTP history
WebSockets history
Proxy settings
Forward
Drop
Intercept is on
Action
Open browser

Vulnerable Code

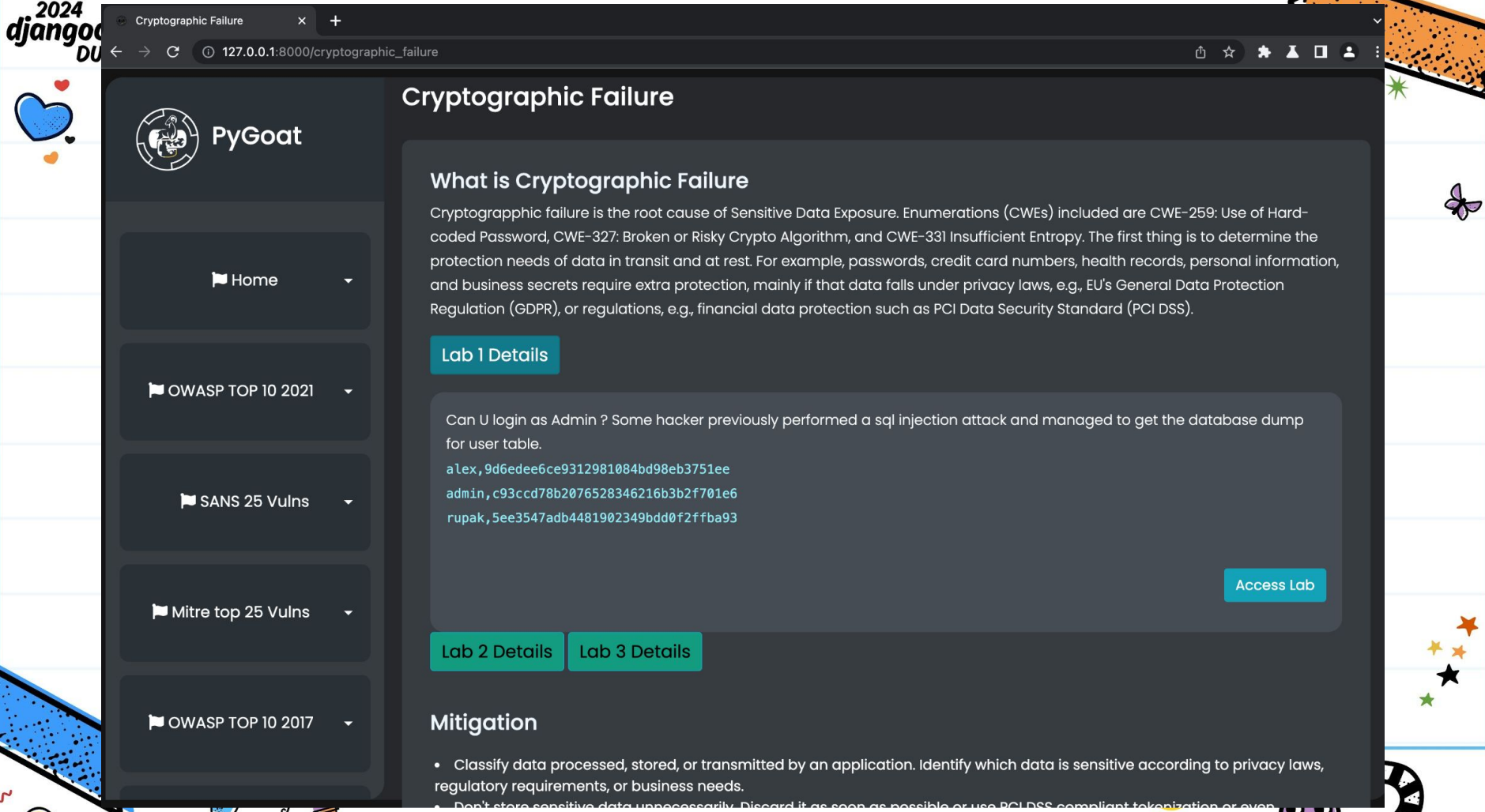
```
if request.COOKIES.get('admin') == "1":
    return render(
        request,
        'Lab_2021/A1_BrokenAccessControl/broken_access_lab_1.html',
        {
            "data":"0NLY_F0R_4DM1N5",
            "username": "admin"
        })
elif (name=='jack' and password=='jacktheripper'): # Will implement hashing here
    html = render(
    request,
    'Lab_2021/A1_BrokenAccessControl/broken_access_lab_1.html',
    {
        "not_admin":"No Secret key for this User",
        "username": name
    })
    html.set_cookie("admin", "0",max_age=200)
    return html
```

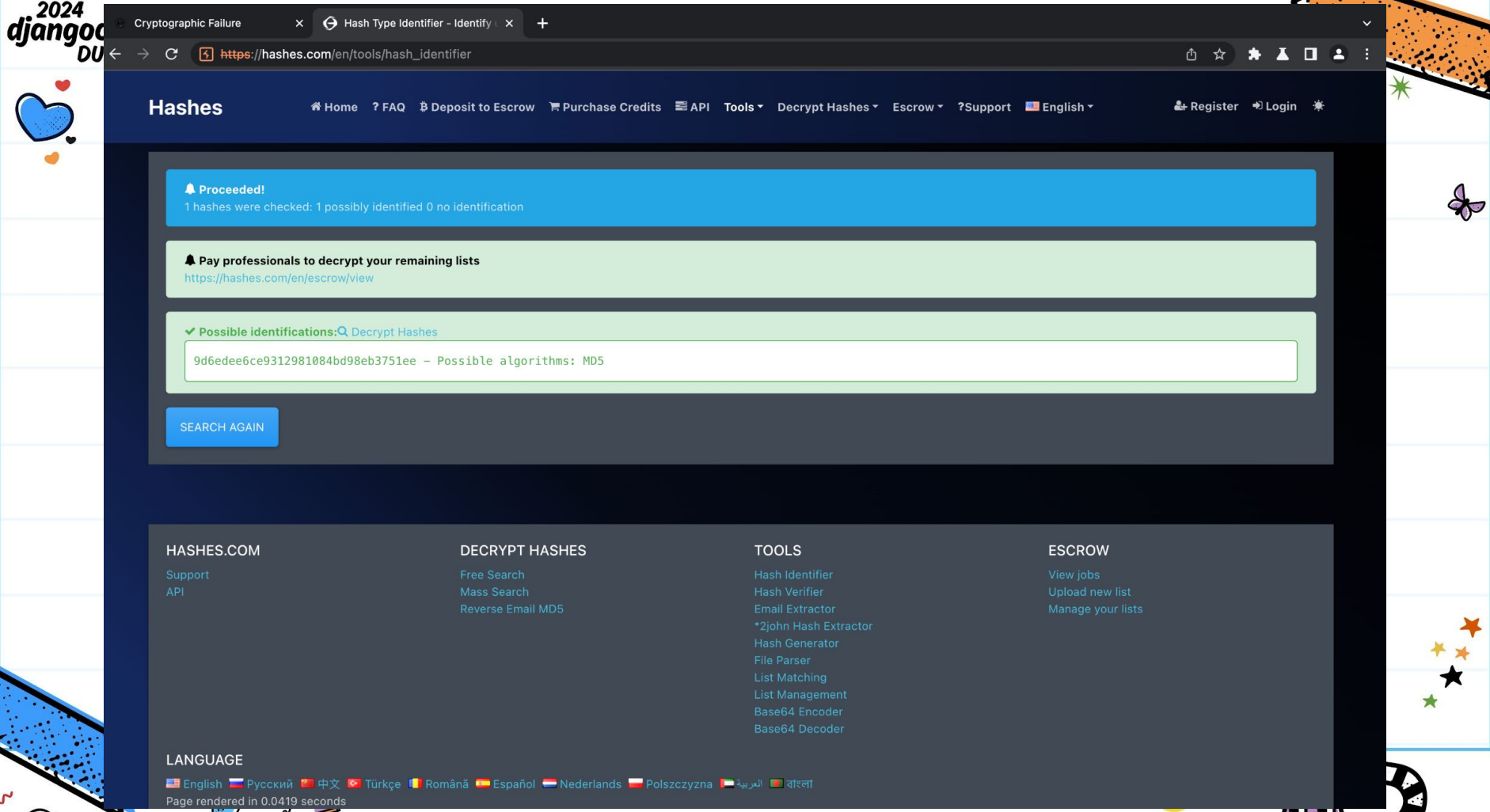
Mitigation

- Except for public resources, **deny by default**.
- Implement access control mechanisms once and **re-use** them throughout the application.
- **Rate limit** API and controller access to minimize the harm from automated attack tooling.
- Stateful **session identifiers** should be invalidated on the server after logout. Stateless **JWT tokens** should rather be short-lived so that the window of opportunity for an attacker is minimized. For longer lived JWTs it's highly recommended to follow the OAuth standards to revoke access.

A2 Cryptographic Failures

Cryptographic failure is the root cause of Sensitive Data Exposure. Enumerations (CWEs) included are CWE-259: Use of Hard-coded Password, CWE-327: Broken or Risky Crypto Algorithm, and CWE-331 Insufficient Entropy.


Cryptographic Failure
127.0.0.1:8000/cryptographic_failure
PyGoat
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Cryptographic Failure
What is Cryptographic Failure
Cryptograpphic failure is the root cause of Sensitive Data Exposure. Enumerations (CWEs) included are CWE-259: Use of Hard-coded Password, CWE-327: Broken or Risky Crypto Algorithm, and CWE-331 Insufficient Entropy. The first thing is to determine the protection needs of data in transit and at rest. For example, passwords, credit card numbers, health records, personal information, and business secrets require extra protection, mainly if that data falls under privacy laws, e.g., EU's General Data Protection Regulation (GDPR), or regulations, e.g., financial data protection such as PCI Data Security Standard (PCI DSS).
Lab 1 Details
Can U login as Admin ? Some hacker previously performed a sql injection attack and managed to get the database dump for user table.
alex,9d6edee6ce9312981084bd98eb3751ee
admin,c93ccd78b2076528346216b3b2f701e6
rupak,5ee3547adb4481902349bdd0f2ffba93
Access Lab
Lab 2 Details
Lab 3 Details
Mitigation
• Classify data processed, stored, or transmitted by an application. Identify which data is sensitive according to privacy laws, regulatory requirements, or business needs.


2024
djangoc
DU
Cryptographic Failure
Hash Type Identifier - Identify
https://hashes.com/en/tools/hash_identifier
Hashes
Home
FAQ
Deposit to Escrow
Purchase Credits
API
Tools
Decrypt Hashes
Escrow
Support
English
Register
Login
Proceeded!
1 hashes were checked: 1 possibly identified 0 no identification
Pay professionals to decrypt your remaining lists
https://hashes.com/en/escrow/view
Possible identifications:
Decrypt Hashes
9d6edee6ce9312981084bd98eb3751ee - Possible algorithms: MD5
SEARCH AGAIN
HASHES.COM
Support
API
DECRYPT HASHES
Free Search
Mass Search
Reverse Email MD5
TOOLS
Hash Identifier
Hash Verifier
Email Extractor
*2john Hash Extractor
Hash Generator
File Parser
List Matching
List Management
Base64 Encoder
Base64 Decoder
ESCROW
View jobs
Upload new list
Manage your lists
LANGUAGE
English
Русский
中文
Türkçe
Română
Español
Nederlands
Polszczyzna
العربية
বাংলা
Page rendered in 0.0419 seconds

Home | Encrypt / Decrypt | Conversion tools | Ciphers | API | Contact | FR | EN

Md5 Decrypt & Encrypt

c93ccd78b2076528346216b3b2f701e6

Encrypt | Decrypt

c93ccd78b2076528346216b3b2f701e6 : **admin1234**

Mitigation

- Classify data processed, stored, or transmitted by an application. Identify which data is sensitive according to privacy laws, regulatory requirements, or business needs.
- Don't store sensitive data unnecessarily. Discard it as soon as possible or use PCI DSS compliant tokenization or even truncation. Data that is not retained cannot be stolen.
- Make sure to encrypt all sensitive data at rest.
- Ensure up-to-date and strong standard algorithms, protocols, and keys are in place; use proper key management.
- Encrypt all data in transit with secure protocols such as TLS with forward secrecy (FS) ciphers, cipher prioritization by the server, and secure parameters. Enforce encryption using directives like HTTP Strict Transport Security (HSTS).
- Disable caching for response that contain sensitive data.

A4: Insecure Design

Insecure design is a type of flaw that can sit in the background of everything you do. This vulnerability relates to how you as a developer design your programs, architect solutions, and employ security practices such as threat modeling

Pygoat lab objective

This lab helps you to get an idea of how Insecure Design can result in major Security flaw. In the next page,user can get 5 free tickets for a Movie. But he/she have to wait untill all the tickets are sold out. For this particular situation, we can get advantage of the Insecure Design and somehow get all the tickets for the movie.

`Hint`

Logout and then think.

Access Lab

This website is giving everyone free tickets (upto 5 per person). And the movie will be public when all the tickets will be sold.

The ticket generating system is inherently secure, limiting users to obtain a maximum of 5 free tickets. However, a significant design flaw exists – multiple accounts can be created to acquire an unlimited number of tickets. For instance, in this specific scenario where 60 tickets are required, one could easily exploit the system by creating just 12 accounts, claiming 5 tickets from each.

Wait until all tickets are sold (55 tickets left)

Claim Upto 5 Free Tickits

5

Claim

My Tickets

QwLmUHyVoO
qFHNJBKSGc
FfxkTjRDMj
YGaiuFkeku
qzauAhPmrw

Watch Movie

Tickit

Watch

Mitigation

- Establish and use a secure development lifecycle with AppSec professionals to help evaluate and design security and privacy-related controls
- Establish and use a library of secure design patterns or paved road ready to use components
- Use threat modeling for critical authentication, access control, business logic, and key flows
- Integrate security language and controls into user stories
- Integrate plausibility checks at each tier of your application (from frontend to backend)

Mitigation

- Write unit and integration tests to validate that all critical flows are resistant to the threat model. Compile use-cases *and* misuse-cases for each tier of your application.
- Segregate tier layers on the system and network layers depending on the exposure and protection needs
- Segregate tenants robustly by design throughout all tiers
- Limit resource consumption by user or service

A8 Software and Data Integrity Failures

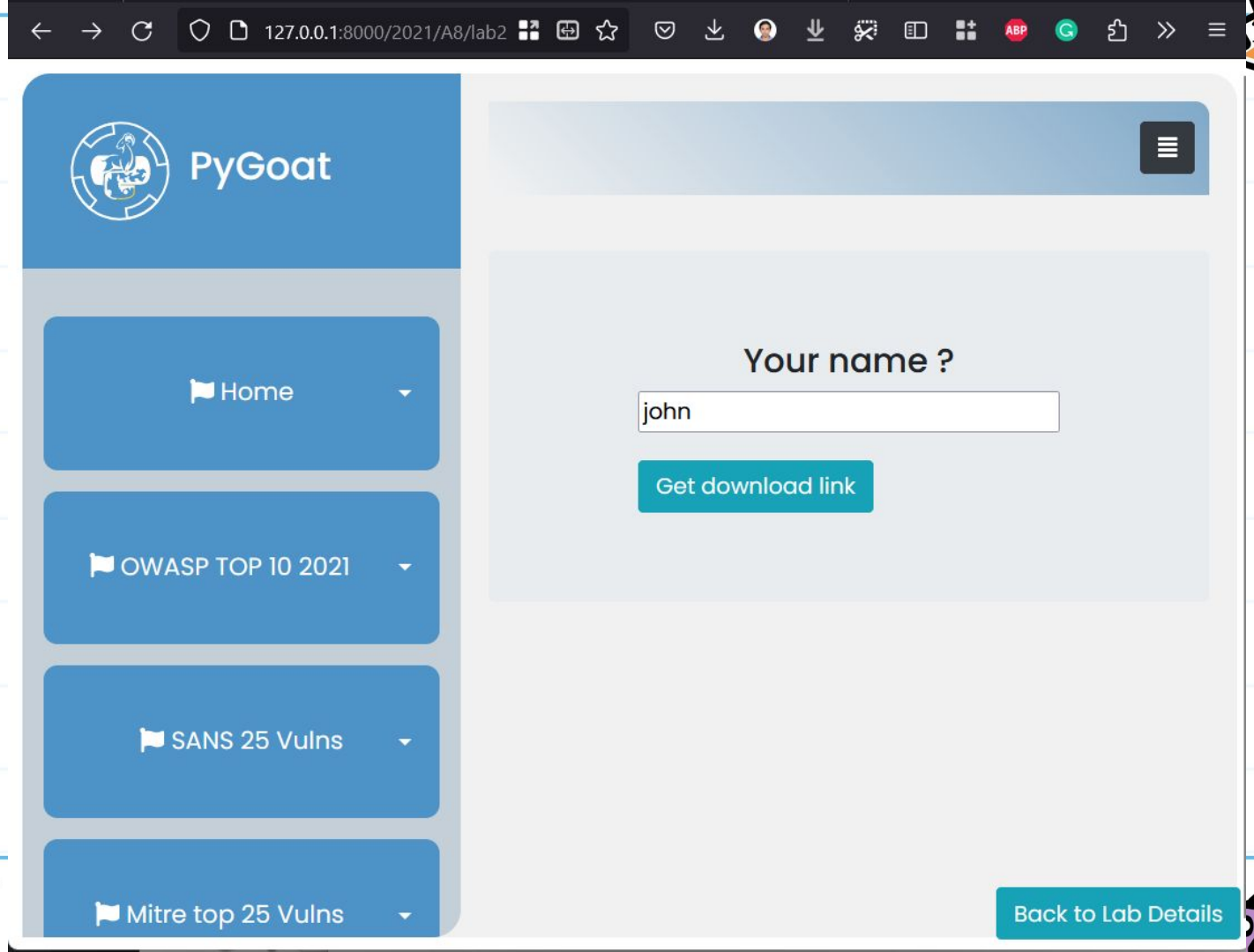
Software and data integrity failures relate to code and infrastructure that does not protect against integrity violations. An example of this is where an application relies upon plugins, libraries, or modules from untrusted sources, repositories, and content delivery networks (CDNs).

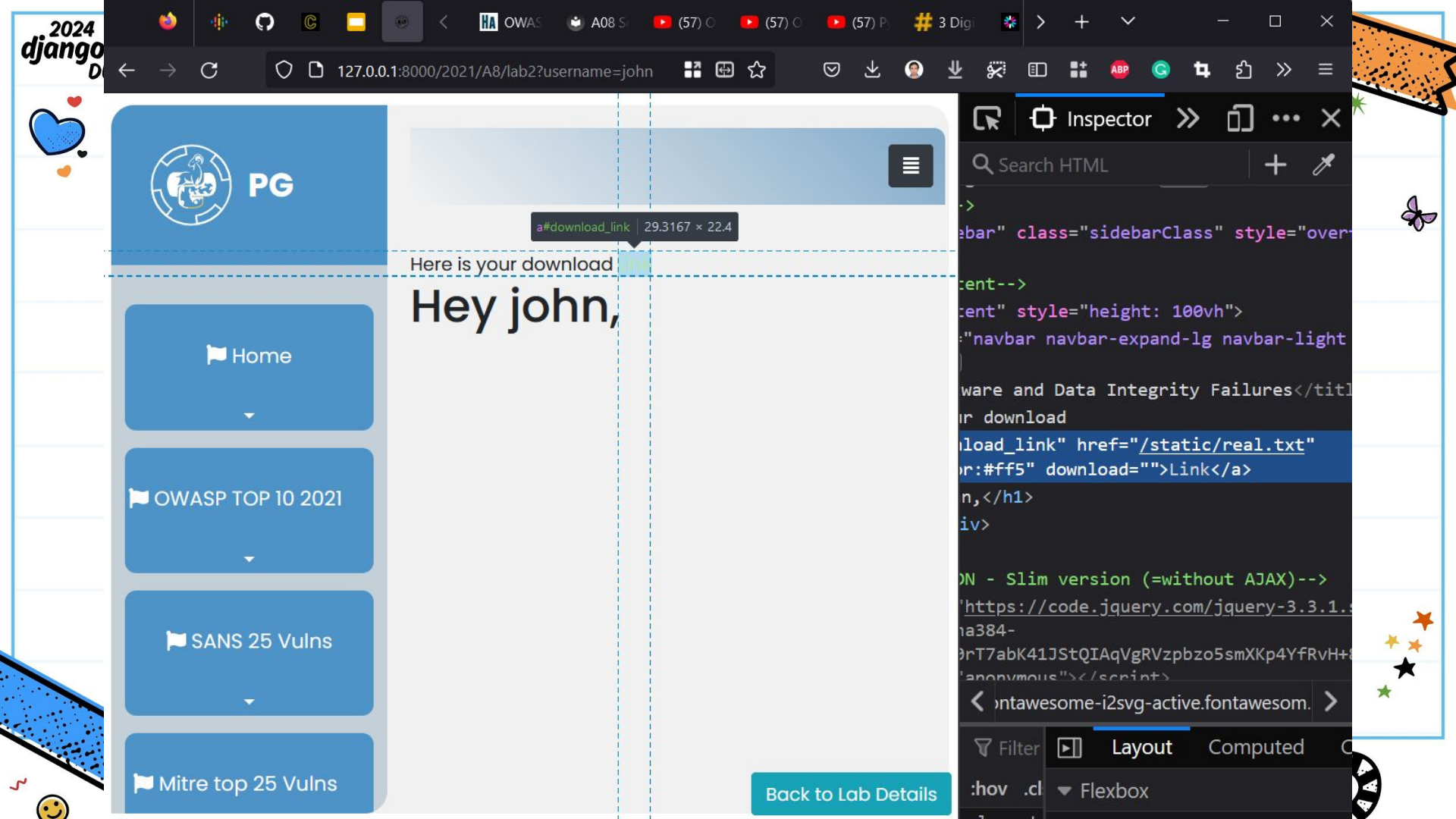
Eg: Insecure Deserialization

Applications and APIs will be vulnerable if they deserialize hostile or tampered objects supplied by an attacker. This can result in two primary types of attacks:

- Object and data structure related attacks where the attacker modifies application logic or achieves arbitrary remote code execution if there are classes available to the application that can change behavior during or after deserialization.
- Typical data tampering attacks such as access-control-related attacks where existing data structures are used but the content is changed.

Pygoat Lab


127.0.0.1:8000/2021/A8/lab2
PyGoat
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
Your name ?
john
Get download link
Back to Lab Details


127.0.0.1:8000/2021/A8/lab2?username=john
PG
a#download_link | 29.3167 × 22.4
Here is your download
Hey john,
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
Back to Lab Details
Inspector
Search HTML
Layout
Computed
Flexbox

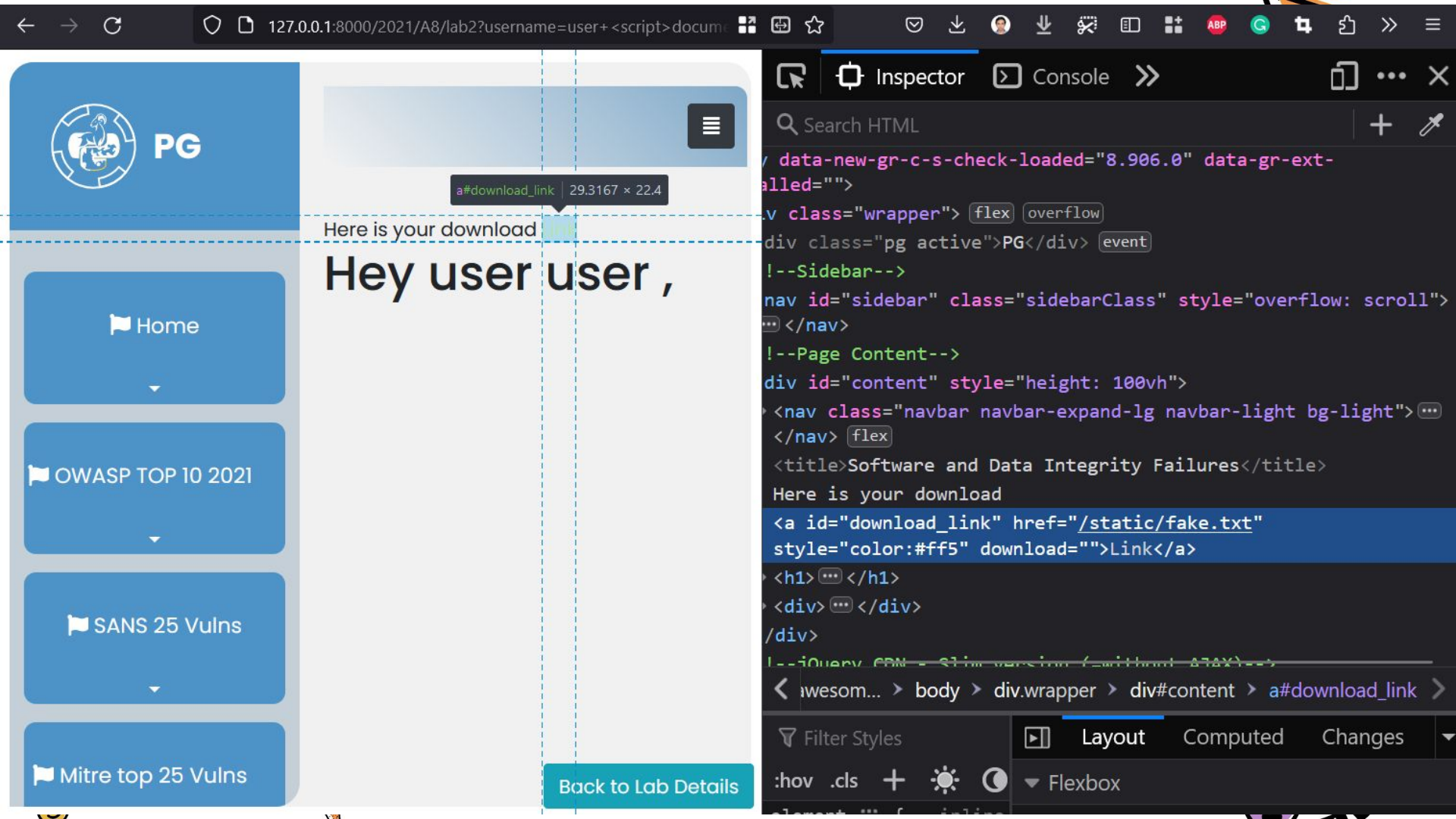
Vulnerability

Page is vulnerable to XSS.

Name is directly printed from the url param.

If we specify name value as:
user+<script>document.getElementById("download_link").setAttribute("href"%2C"%2Fstatic%2Ffake.txt")%3B<%2Fscript>user+<script>document.getElementById("download_link").setAttribute("href"%2C"%2Fstatic%2Ffake.txt")%3B<%2Fscript>

The download url is modified and user may download an arbitrary file by trusting the domain


127.0.0.1:8000/2021/A8/lab2?username=user+<script>docum
PG
a#download_link 29.3167 × 22.4
Here is your download Link
Hey user user ,
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
Back to Lab Details
Inspector
Console
Search HTML
data-new-gr-c-s-check-loaded="8.906.0" data-gr-ext-
alled="">
v class="wrapper"> flex overflow
div class="pg active">PG event
!--Sidebar-->
nav id="sidebar" class="sidebarClass" style="overflow: scroll">
</nav>
!--Page Content-->
div id="content" style="height: 100vh">
<nav class="navbar navbar-expand-lg navbar-light bg-light">
</nav> flex
<title>Software and Data Integrity Failures</title>
Here is your download
<a id="download_link" href="/static/fake.txt"
style="color:#ff5" download="">Link
<h1>
</h1>

/div>
awesom... > body > div.wrapper > div#content > a#download_link
Filter Styles
:hov .cls
Layout
Computed
Changes
Flexbox

Mitigation

For the demonstrated vulnerability, XSS is used to redirect user to a fake file.

As a user we should always cross-check signatures for verification of Data Integrity. Checksums should be provided for downloads so that it can be cross checked from user end.

Mitigation

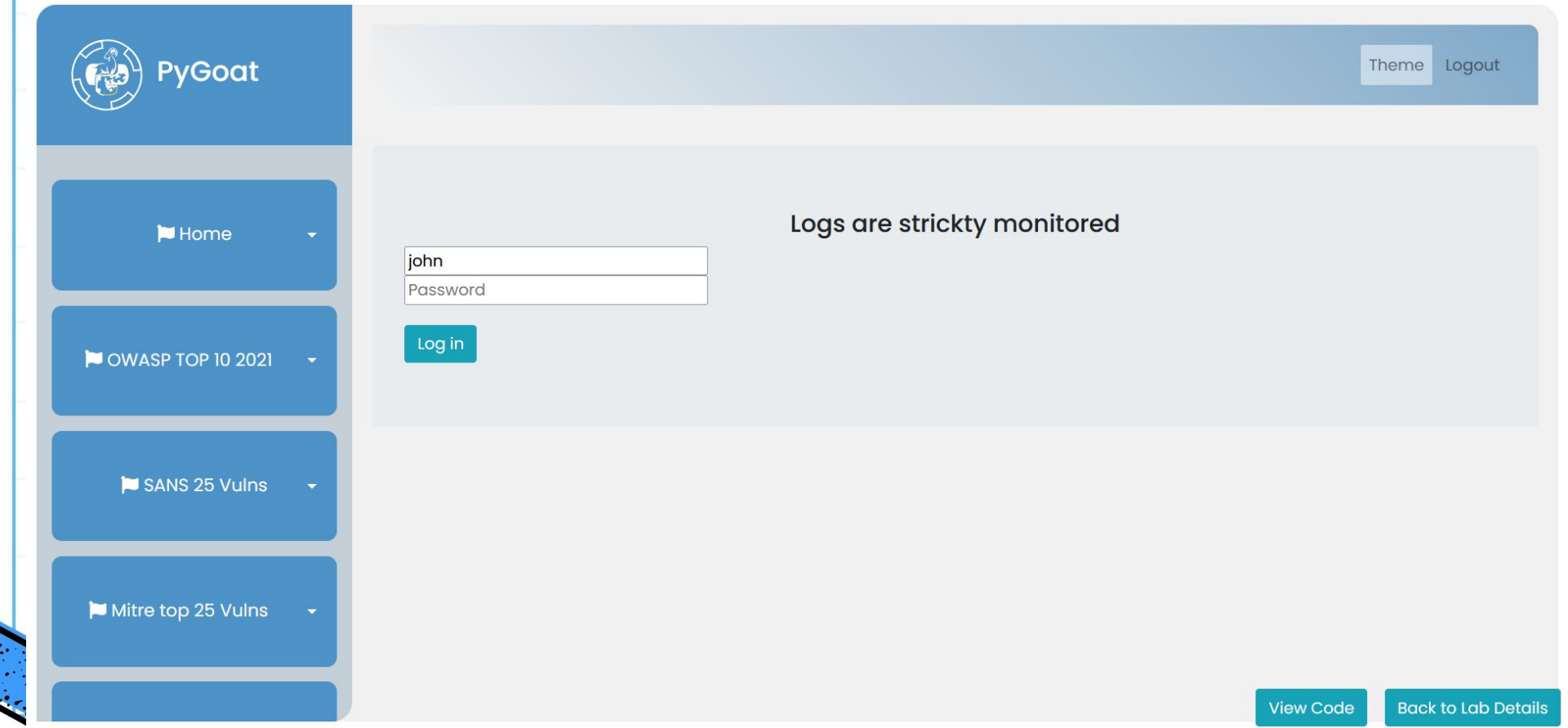
- Use digital signatures or similar mechanisms to verify the software or data is from the expected source and has not been altered.
- Ensure libraries and dependencies, are consuming trusted repositories. If you have a higher risk profile, consider hosting an internal known-good repository that's vetted.
- Ensure that a software supply chain security tool, such as OWASP Dependency Check or OWASP CycloneDX, is used to verify that components do not contain known vulnerabilities
- Ensure that your CI/CD pipeline has proper segregation, configuration, and access control to ensure the integrity of the code flowing through the build and deploy processes.
- Ensure that unsigned or unencrypted serialized data is not sent to untrusted clients without some form of integrity check or digital signature to detect tampering or replay of the serialized data

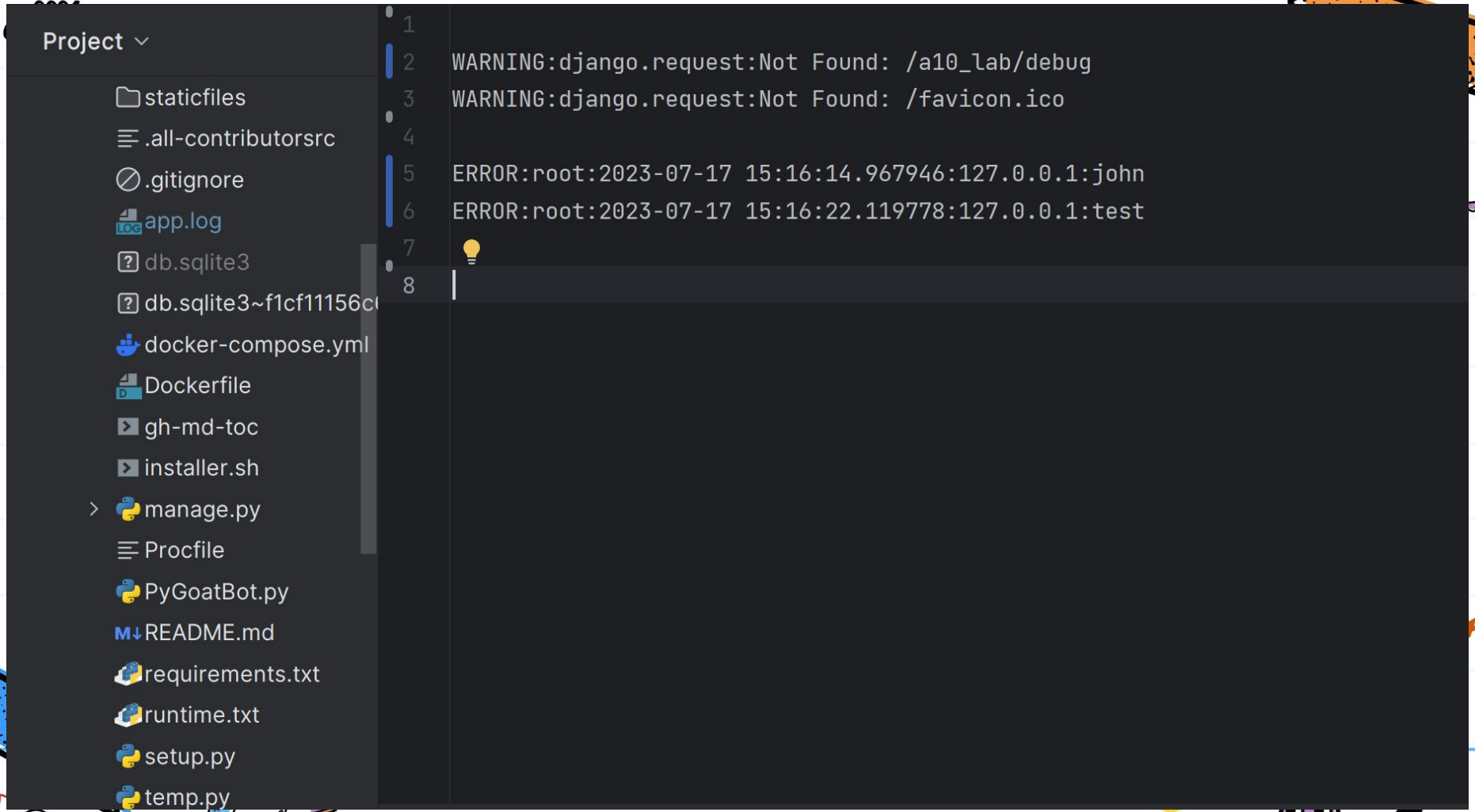
A09: **Security Logging and Monitoring Failures**

This category is to help detect, escalate, and respond to active breaches. Without logging and monitoring, breaches cannot be detected. Insufficient logging, detection, monitoring, and active response occurs any time:

- Auditable events, such as logins, failed logins, and high-value transactions, are not logged.
- Warnings and errors generate no, inadequate, or unclear log messages.
- Logs of applications and APIs are not monitored for suspicious activity.
- Appropriate alerting thresholds and response escalation processes are not in place or effective.
- Penetration testing and scans by dynamic application security testing (DAST) tools (such as OWASP ZAP) do not trigger alerts.
- The application cannot detect, escalate, or alert for active attacks in real-time or near real-time.

Pygoat Lab


PyGoat
Theme
Logout
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
Logs are strickty monitored
john
Password
Log in
View Code
Back to Lab Details


Project
staticfiles
.all-contributorsrc
.gitignore
app.log
db.sqlite3
db.sqlite3~f1cf11156c
docker-compose.yml
Dockerfile
gh-md-toc
installer.sh
manage.py
Procfile
PyGoatBot.py
README.md
requirements.txt
runtime.txt
setup.py
temp.py
1
2 WARNING:django.request:Not Found: /a10_lab/debug
3 WARNING:django.request:Not Found: /favicon.ico
4
5 ERROR:root:2023-07-17 15:16:14.967946:127.0.0.1:john
6 ERROR:root:2023-07-17 15:16:22.119778:127.0.0.1:test
7
8

```
WARNING:django.request:Not Found: /a10_lab/debug
WARNING:django.request:Not Found: /favicon.ico

ERROR:root:2023-07-17 15:16:14.967946:127.0.0.1:john
ERROR:root:2023-07-17 15:16:22.119778:127.0.0.1:test

ERROR:root:2023-07-17 15:30:31.781079:127.0.0.1:Python version is outdated. Update from evil.com/py3
ERROR:root:2023-07-17 15:20:04.464767:127.0.0.1: dolor sit amet, consectetuer adipiscing elit. Aenean comm
```

Mitigation

Ensure all login, access control, and server-side input validation failures can be logged with sufficient user context to identify suspicious or malicious accounts and held for enough time to allow delayed forensic analysis.

Ensure that logs are generated in a format that log management solutions can easily consume.

Ensure log data is encoded correctly to prevent injections or attacks on the logging or monitoring systems.

Ensure high-value transactions have an audit trail with integrity controls to prevent tampering or deletion, such as append-only database tables or similar.

DevSecOps teams should establish effective monitoring and alerting such that suspicious activities are detected and responded to quickly.

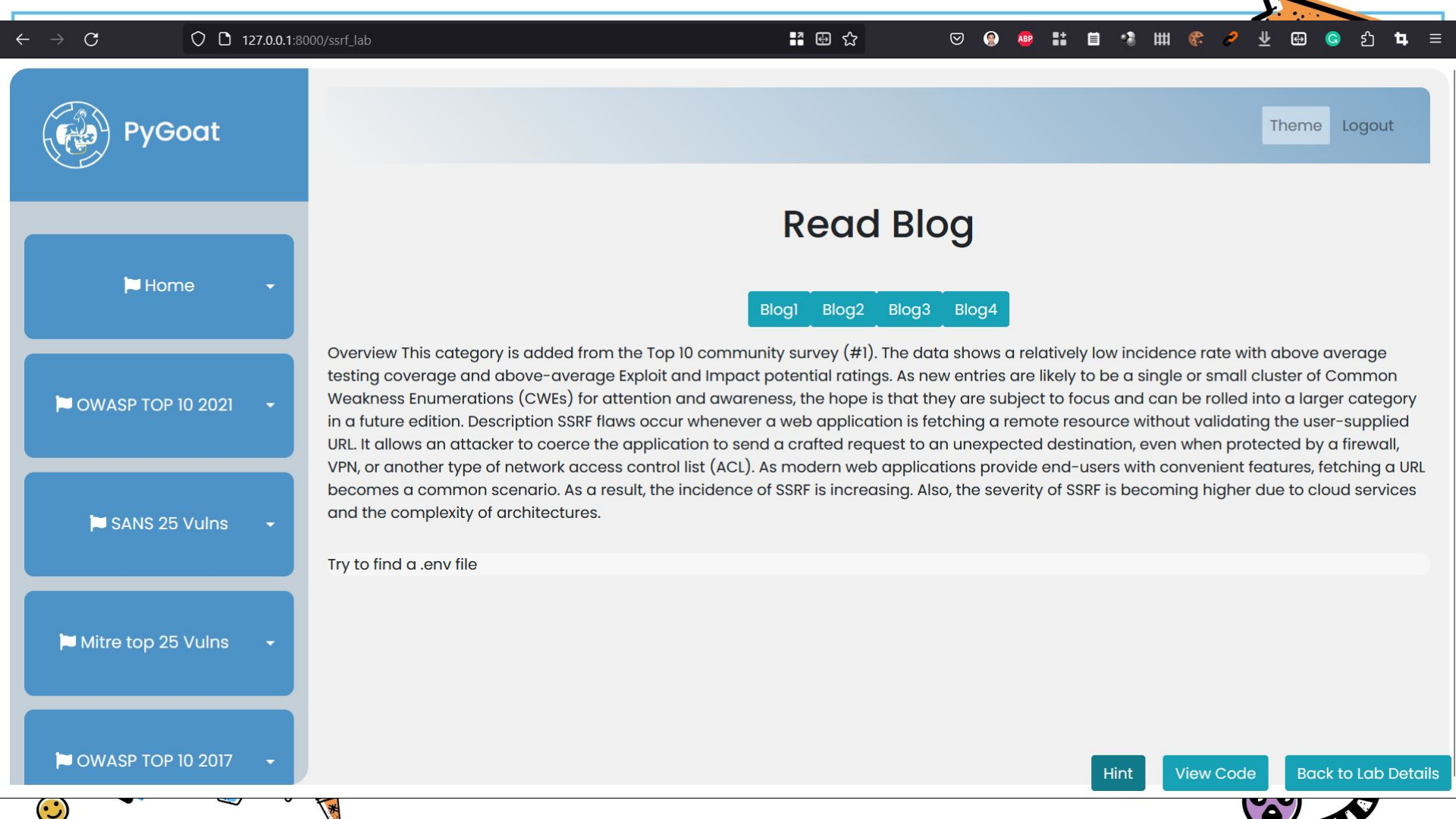
A10: Server Side Request Forgery (SSRF)

- SSRF flaws occur whenever a web application is fetching a remote resource without validating the user-supplied URL. It allows an attacker to coerce the application to send a crafted request to an unexpected destination.

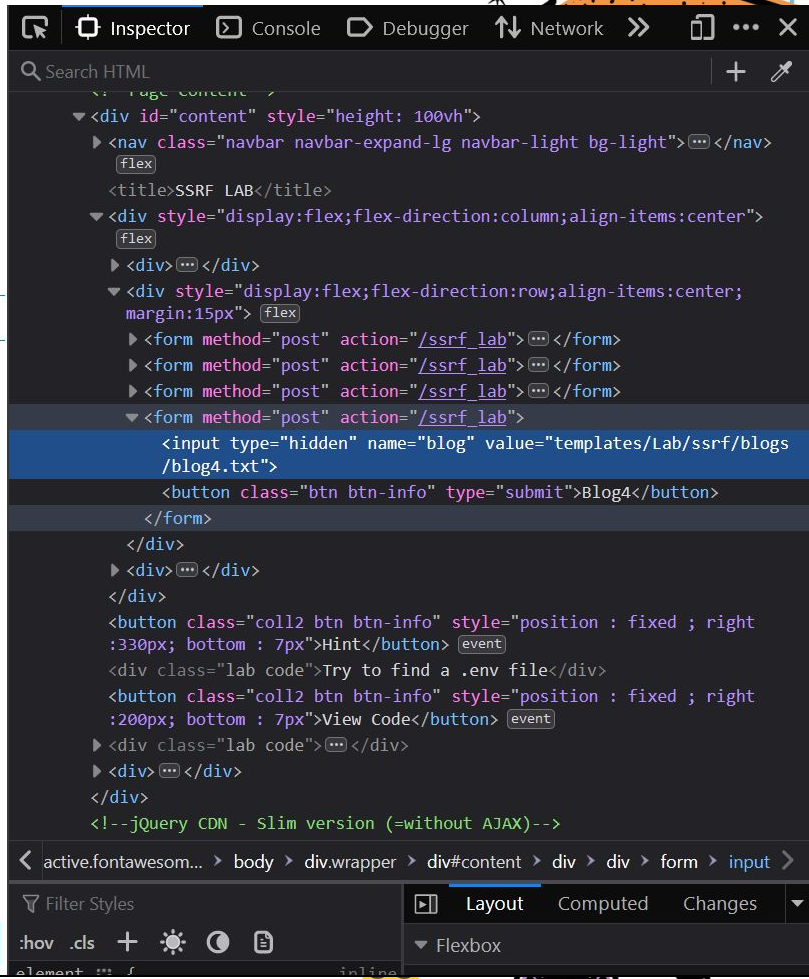
Sample usages for remote url fetching: DNS Checkers, URL previews on social media

- Common attacks: Attacks on internal services/files, Attack on external URLs (DoS)

Pygoat - SSRF Lab

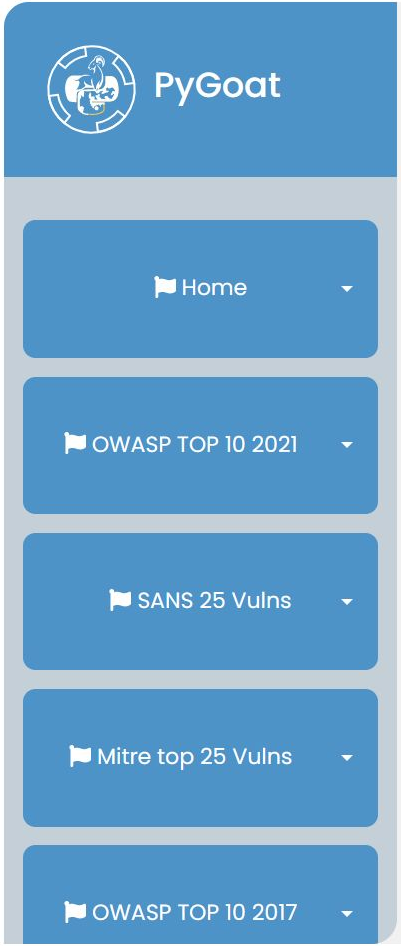

127.0.0.1:8000/ssrf_lab
PyGoat
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Theme
Logout
Read Blog
Blog1
Blog2
Blog3
Blog4
Overview This category is added from the Top 10 community survey (#1). The data shows a relatively low incidence rate with above average testing coverage and above-average Exploit and Impact potential ratings. As new entries are likely to be a single or small cluster of Common Weakness Enumerations (CWEs) for attention and awareness, the hope is that they are subject to focus and can be rolled into a larger category in a future edition. Description SSRF flaws occur whenever a web application is fetching a remote resource without validating the user-supplied URL. It allows an attacker to coerce the application to send a crafted request to an unexpected destination, even when protected by a firewall, VPN, or another type of network access control list (ACL). As modern web applications provide end-users with convenient features, fetching a URL becomes a common scenario. As a result, the incidence of SSRF is increasing. Also, the severity of SSRF is becoming higher due to cloud services and the complexity of architectures.
Try to find a .env file
Hint
View Code
Back to Lab Details


PyGoat
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Read Blog
form | 70.45 × 37.6 | Flex Item
Blog1
Blog2
Blog3
Blog4
The purpose is to give both developers and testers a platform for learning how to test applications and how to code securely. PyGoat is written in python and used Django web framework as a platform. It has both traditional web application vulnerabilities (i.e. XSS, SQLi) as well. PyGoat also has an area where you can see the source code to determine where the mistake was made that caused the vulnerability and allows you to make changes to secure it.
Hint
View Code
Back to Lab Details


Inspector
Console
Debugger
Network
Search HTML

<nav class="navbar navbar-expand-lg navbar-light bg-light">
<title>SSRF LAB</title>

<form method="post" action="/ssrf_lab">
<form method="post" action="/ssrf_lab">
<form method="post" action="/ssrf_lab">
<form method="post" action="/ssrf_lab">
<input type="hidden" name="blog" value="templates/Lab/ssrf/blogs/blog4.txt">
<button class="btn btn-info" type="submit">Blog4</button>
</form>
<button class="col12 btn btn-info" style="position : fixed ; right :330px; bottom : 7px">Hint</button>
Try to find a .env file
<button class="col12 btn btn-info" style="position : fixed ; right :200px; bottom : 7px">View Code</button>
<!--jQuery CDN - Slim version (=without AJAX)-->
active.fontawesom... > body > div.wrapper > div#content > div > div > form > input
Filter Styles
:hov .cls
Layout
Computed
Changes
Flexbox

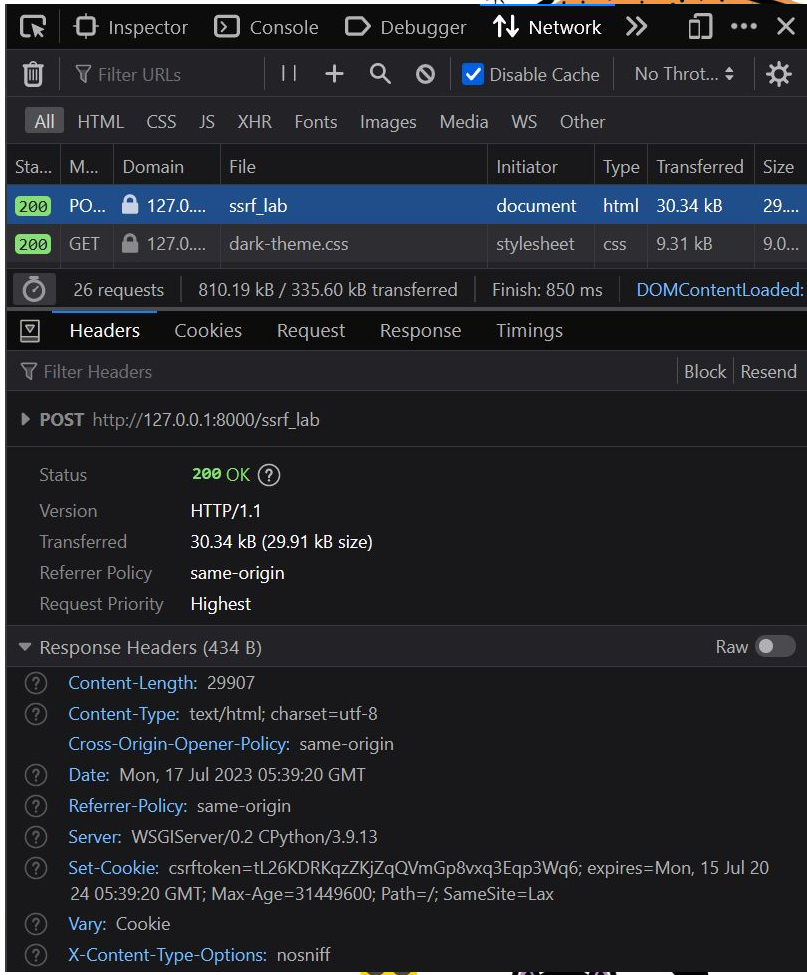


Read Blog

Blog1 | Blog2 | Blog3 | Blog4

The purpose is to give both developers and testers a platform for learning how to test applications and how to code securely. PyGoat is written in python and used Django web framework as a platform. It has both traditional web application vulnerabilities (i.e. XSS, SQLi) as well. PyGoat also has an area where you can see the source code to determine where the mistake was made that caused the vulnerability and allows you to make changes to secure it.

Hint | View Code | Back to Lab Details



PyGoat

- Home
- OWASP TOP 10 2021
- SANS 25 Vulns
- Mitre top 25 Vulns
- OWASP TOP 10 2017

Read Blog

Blog1 | Blog2 | Blog3 | Blog4

The purpose is to give both developers and testers a platform for learning how to test applications and how to code securely. PyGoat is written in python and used Django web framework as a platform. It has both traditional web application vulnerabilities (i.e. XSS, SQLi) as well. PyGoat also has an area where you can see the source code to determine where the mistake was made that caused the vulnerability and allows you to make changes to secure it.

Hint | View Code | Back to Lab Details

Inspector | Console | Debugger | Network

Filter URLs | Disable Cache | No Throt...

All | HTML | CSS | JS | XHR | Fonts | Images | Media | WS | Other

Sta...	M...	Domain	File	Initiator	Type	Transferred	Size
200	PO...	127.0....	ssrf_lab	document	html	30.34 kB	29....
200	GET	127.0....	dark-theme.css	stylesheet	css	9.31 kB	9.0...

26 requests | 810.19 kB / 335.60 kB transferred | Finish: 850 ms | DOMContentLoaded:

Headers | Cookies | Request | Response | Timings

Filter Request Parameters

Form data | Raw

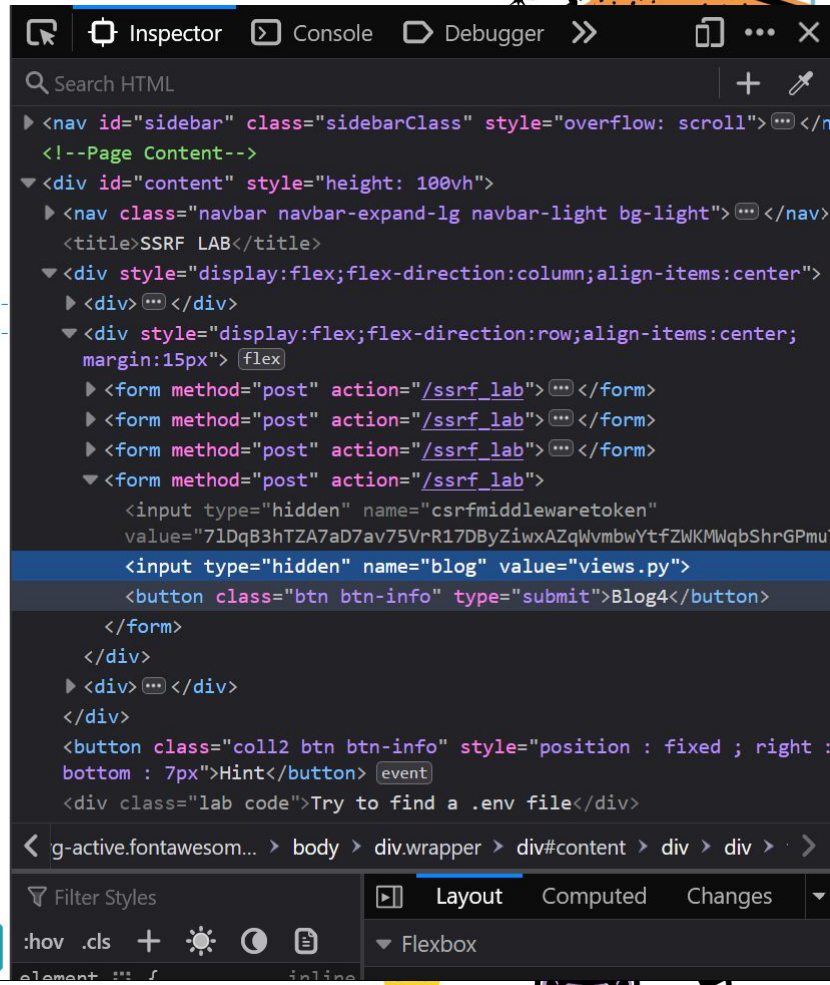
csrfmiddlewaretoken: "8a9OUVKsRlTyUErbkTZ8z9vFUoYLQtibrL1Kuor27Kl83tHR55vnxuSVNSe0Jfy7"

blog: "templates/Lab/ssrf/blogs/blog4.txt"

Vulnerable Code

```python
def ssrf_lab(request):

    if request.user.is_authenticated:
        if request.method=="GET":
            return render(request,"Lab/ssrf/ssrf_lab.html",{"blog":"Read Blog About SSRF"})

        else:
            # vulnerable code
            file=request.POST["blog"]
            try :
                dirname = os.path.dirname(__file__)
                filename = os.path.join(dirname, file)
                file = open(filename,"r")
                data = file.read()
                return render(request,"Lab/ssrf/ssrf_lab.html",{"blog":data})
            except:
                return render(request, "Lab/ssrf/ssrf_lab.html", {"blog": "No blog found"})
    else:
        return redirect('login')
```


PyGoat
Home
OWASP TOP 10 2021
SANS 25 Vulns
Mitre top 25 Vulns
OWASP TOP 10 2017
Read Blog
button.btn.btn-info | 70.45 × 37.6
Blog1
Blog2
Blog3
Blog4
The purpose is to give both developers and testers a platform for learning how to test applications and how to code securely. PyGoat is written in python and used Django web framework as a platform. It has both traditional web application vulnerabilities (i.e. XSS, SQLi) as well. PyGoat also has an area where you can see the source code to determine where the mistake was made that caused the vulnerability and allows you to make changes to secure it.
Hint
View Code
Back to Lab Details


Inspector
Console
Debugger
Search HTML
<nav id="sidebar" class="sidebarClass" style="overflow: scroll">
<!--Page Content-->

<nav class="navbar navbar-expand-lg navbar-light bg-light">
<title>SSRF LAB</title>

flex
<form method="post" action="/ssrf_lab">
<form method="post" action="/ssrf_lab">
<form method="post" action="/ssrf_lab">
<form method="post" action="/ssrf_lab">
<input type="hidden" name="csrfmiddlewaretoken" value="71DqB3hTZA7aD7av75VrR17DByZiwxAZqWvmbwYtfZWKMWqbShrGPmu
<input type="hidden" name="blog" value="views.py">
<button class="btn btn-info" type="submit">Blog4</button>
</form>

<button class="col12 btn btn-info" style="position : fixed ; right : bottom : 7px">Hint</button>
event
Try to find a .env file
g-active.fontawesom...
body
div.wrapper
div#content
div
div
Filter Styles
:hov
.cls
Layout
Computed
Changes
Flexbox

PyGoat

Theme Logout

Home

OWASP TOP 10 2021

SANS 25 Vulns

Mitre top 25 Vulns

OWASP TOP 10 2017

Read Blog

Blog1 Blog2 Blog3 Blog4

import hashlib from django.shortcuts import render,redirect from django.http import HttpResponse, HttpResponseBadRequest, JsonResponse from .models import FAANG, AF_session_id,info,login,comments,authLogin, tickits, sql_lab_table,Blogs,CF_user,AF_admin from django.core import serializers from requests.structures import CaseInsensitiveDict from django.contrib.auth import login,authenticate from django.contrib.auth.forms import UserCreationForm import random import string import os from hashlib import md5 import datetime from .forms import NewUserForm from django.contrib import messages #**********Lab Requirements**********# from .models import FAANG,info,login,comments,otp from random import randint from xml.dom.pulldom import parseString, START_ELEMENT from xml.sax.handler import feature_external_ges from xml.sax import make_parser from django.views.decorators.csrf import csrf_exempt from django.template import loader from django.template.loader import render_to_string import subprocess import pickle import base64 import yaml import json from dataclasses import dataclass import uuid from .utility import filter_blog, customHash import jwt from PIL import Image,ImageMath import base64 from io import BytesIO from argon2 import PasswordHasher import logging import requests import re #**********Login and Registration**********# def register(request): if request.method == "POST": form = NewUserForm(request.POST) if form.is_valid(): user = form.save() login(request, user) messages.success(request, "Registration successful.") return redirect('/') messages.error(request, "Unsuccessful registration. Invalid information.") form = NewUserForm() return render (request=request, template_name="registration/register.html", context={"register_form":form}) # def register(request): # if request.method=="POST": # form = UserCreationForm(request.POST) # if form.is_valid(): # form.save() # return redirect("login") # else: # form=UserCreationForm() # return render(request,"registration/register.html",{"form":form,}) def home(request): if request.user.is_authenticated: return render(request,'introduction/home.html',) else: return redirect('login') ## authentication check decurator function def authentication_decorator(func): def function(*args, **kwargs): if args[0].user.is_authenticated: return func(*args, **kwargs) else: return redirect('login') return function #**********XSS**********

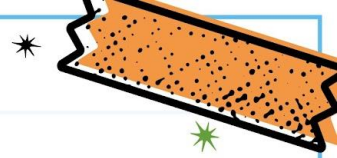
Hint View Code Back to Lab Details

xss(request): if request.user.is_authenticated: return render(request,"Lab/XSS/xss.html") else: return redirect('login') def xss_lab(request): if

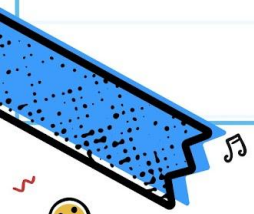
Mitigation

From Application layer:

- Sanitize and validate all client-supplied input data
- Enforce the URL schema, port, and destination with a positive allow list
- Do not send raw responses to clients



Fixed code



```python
def ssrf_lab(request):
    if request.user.is_authenticated:
        if request.method == "GET":
            return render(request, "Lab/ssrf/ssrf_lab.html", {"blog": "Read Blog About SSRF"})

        else:
            # We only take blog slug as the input
            blog_post_slug = request.POST["blog"] # example blog_1, blog_2, ...
            try:
                # should not have slashes. Only allows numbers, letters, underscores and max 50 chars.
                if not re.fullmatch('[a-zA-Z0-9_]{1,50}', blog_post_slug):
                    raise ValueError()

                dirname = os.path.dirname(__file__)
                # only allow txt files in the blog template directory to be loaded
                file_patch = f"templates/Lab/ssrf/blogs/{blog_post_slug}.txt"
                filename = os.path.join(dirname, file_patch)
                file = open(filename, "r")
                data = file.read()
                return render(request, "Lab/ssrf/ssrf_lab.html", {"blog": data})
            except:
                return render(request, "Lab/ssrf/ssrf_lab.html", {"blog": "No blog found"})
    else:
        return redirect('login')
```

2024 djangocon.us DURHAM

Thank You

Get the talk materials & connect with me

linkhq.co/adarsh