

PyGoat

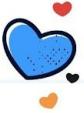
*Learn Django security
the hard way*



Adarsh Divakaran

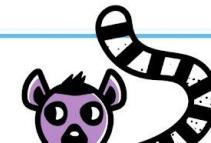
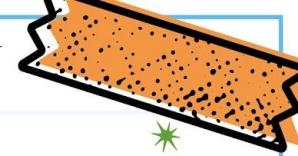
Co-Founder and Lead Consultant
@ Digievo Labs





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**, and are visited by bots approximately 2,306 times a week.
- Malicious bots represent over **60%** of all bot traffic
- **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>

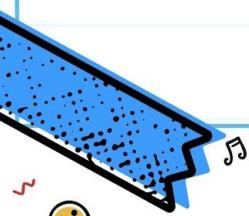
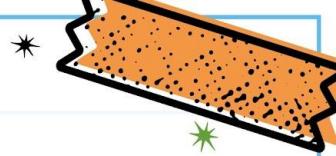




Web Application Security

For the businesses, these attacks lead to:

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



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](#)

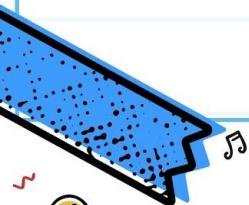
Breaches you were pwned in

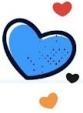
A "breach" is an incident where data has been unintentionally exposed to the public. Using the 1Password password manager helps you ensure all your passwords are strong and unique such that a breach of one service doesn't put your other services at risk.



Ensuring Web Application Security

- Lots of resources are available on building Web applications
- Relatively less guidance and resources for building **Secure Web Applications.**

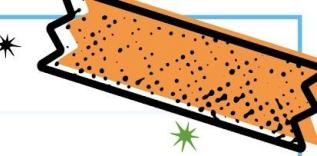
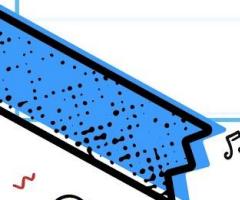




OWASP

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 labs to learn about and exploit the OWASP top 10 vulnerabilities.

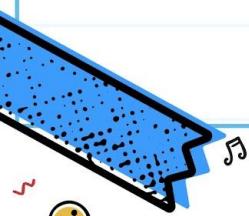
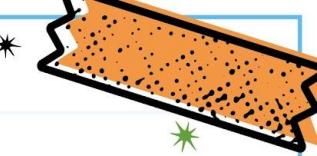




Security Testing Methodology

A security researcher will:

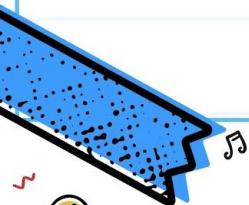
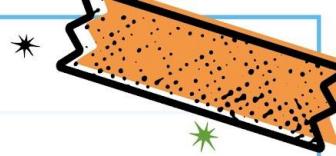
- Perform **enumeration** against the target - Identify the tech stack, various operations and find points of attack.
- They will then proceed with **testing** the web application functionalities (against multiple vulnerabilities) to uncover security bugs - using automated or manual tools.





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 trial and error.

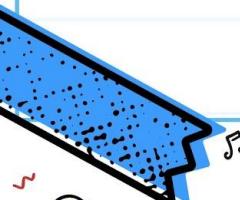
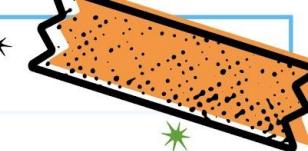




A3: Injection

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**.

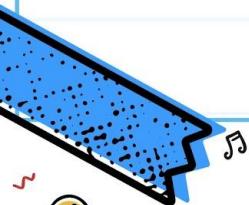
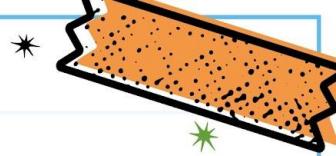
Eg: SQL, NoSQL, OS command, Object Relational Mapping (ORM) injection





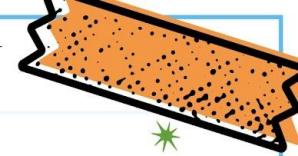
Pygoat Lab 1

Mission: Get unauthorized access as admin





Demo





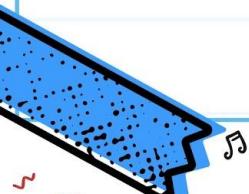
Pygoat Lab 1

```
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 '
```





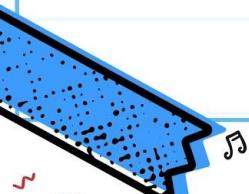
Pygoat Lab 1

```
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'
```





```
<!DOCTYPE html>
<html lang="en" class="fontawesome-i2svg-active fontawesome-i2svg-complete">
  <head> ...
    <body data-new-gr-c-s-check-loaded="14.1115.0" data-gr-ext-installed>
      <div class="wrapper"> ...
        <div class="pg active">PG</div>
        <!-- Sidebar -->
        <nav id="sidebar" style="overflow: scroll" class="sidebarclass"> ...
          <!-- Page Content -->
          <div data-bbox="348 135 500 905" data-new-gr-c-s-check-loaded="14.1115.0" data-gr-ext-installed>
            <div class="content" style="height: 100vh">
              <nav class="navbar navbar-expand-lg navbar-light bg-light"> ...
                <title>SQL LAB</title>
                <div class="jumbotron">
                  <h4 style="text-align:center">Can You Log in as Admin</h4>
                <div class="login">
                  <form method="post" action="/injection_sql_lab">
                    <input type="hidden" name="csrfmiddlewaretoken" value="mbTzEPtNnll1BD917z1kKUjyDRRtw6kq349pxGNksQz684NS61omvwWpb9igzRdx">
                    <input id="input" type="text" name="name" placeholder="User Name">
                    <br>
                    <input id="input" type="password" name="pass" placeholder="Password" value="\$0">
                    <br>
                    <button style="margin-top:20px" class="btn btn-info" type="submit"> Log in </button>
                  </form>
                </div>
              </div>
              <div align="right"> ...
                <p></p>
              </div>
            </div>
            <!-- jQuery CDN - Slim version (=without AJAX) -->
            <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8IYXgZJLZVxXqJv8ZtE5Gq+QFZJdHs0oXqfW9PfD985" crossorigin="anonymous"></script>
            <!-- Pooper.JS -->
          </div>
        </div>
      </div>
    </body>
  </html>
```





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()

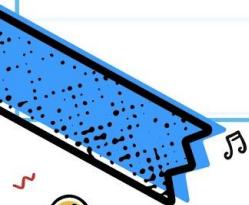
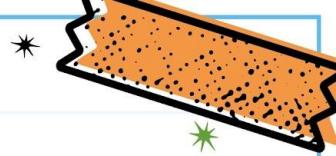
print(sql_query)

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



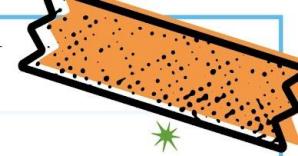
Pygoat Lab 2

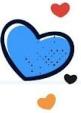
Mission: Run shell commands on the server





Demo





PyGoat

- 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,TS04,TS06,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
ether 1c:57:dc:46:6e:0f
inet6 fe80::1814:d30d:72d1%en0 prefixlen 64 secured scopeid 0xc
inet 192.168.236.18 netmask 0xffffffff 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,TS04,TS06,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
ether 22:51:c1:cf:ca:f7
inet6 fe80::2051:c1ff:fecc: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:fecc:caf7%llw0 prefixlen 64 scopeid 0xe
nd6 options=201<PERFORMNUD,DAD>
media: autoselect
status: inactive
utun0: flags=8851<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1380
inet6 fe80::44b7:b33:731e:c8a0%utun0 prefixlen 64 scopeid 0xf
nd6 options=201<PERFORMNUD,DAD>
utun1: flags=8851<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 2000
inet6 fe80::3dd6:eff3:68be:4723%utun1 prefixlen 64 scopeid 0x10
nd6 options=201<PERFORMNUD,DAD>
utun2: flags=8851<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



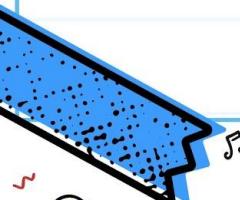
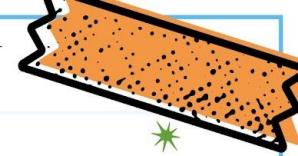
```
if (request.method == "POST"):
    domain = request.POST.get('domain')
    domain = domain.replace("https://www.", '')
    os = request.POST.get('os')
    print(os)
    if (os == 'win'):
        command = "nslookup {}".format(domain)
    else:
        command = "dig {}".format(domain)

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**.

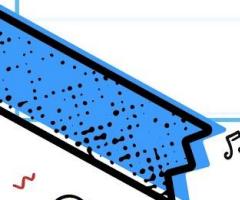
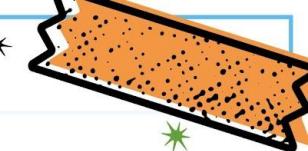




A5: Security Misconfiguration

The application might be vulnerable if the application is:

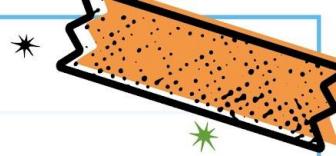
- **Missing** appropriate **security hardening** across any part of the application stack or improperly configured permissions on cloud services.
- **Unnecessary features** are enabled or installed (e.g., unnecessary ports, services, pages, accounts, or privileges).
- **Default accounts** and their passwords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





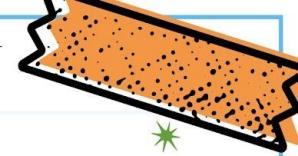
Pygoat Lab 1

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

[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=L26KDRKqzZkjZqQVmGp8vxq3Eqp3Wq6; sessionid=566p3imfh9se6z5g9vhzuShbKomcmchh;
ph_fo2TzM1AWdh5WlaoftTYiInBhS4XzTzRqls50kVziv_posthog=
7B%22distinct_id%2213A2218944b1837dc65-04cb8ccb6691d8-d505429-14400
0-18944b1837e1347%22C%22device_id%2213A2218944b1837dc65-04cb8cb
b6691d8-d505429-144000-18944b1837e1347%22%20%224user_state%22%3A%22
anonymouse%22%22extension_version%22%3A%221.5.%22%20%224session_
recording_enabled_server_side%22%3Afalse%20%22%24autocapture_disabled
_server_side%22%3Afalse%20%22%24active_feature_flags%22%3A%5B%20%22%2
2%24enabled_feature_flags%22%3A%7B%22enable-session-recording%22%3Afa
lse%22sourcing%22%3Afalse%2C%22only-company-edit%22%3Afalse%20%22j
ob-lists%22%3Afalse%7D%20%22%24feature_flag_payloads%22%3A%7B%7D%7D;
messages=
WsiX19gcC9uX21lc3NhZ2UiLDAsMjUsI1J1Z21zdHJhdGlvbibZsdWNjZXNzZnVsLiIsI
iJAX0:lgQDrJ:0TRQ0DrvUuaQmBD7vfc2xlrz10ZZCqBvNnhKobsNh8dc
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
16
```

Inspector

Request attributes

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

 Request to <http://127.0.0.1:8000>

Forward	Drop	Intercept is on	Action
HTTP	Raw	Hex	
ET /secret HTTP/1.1 ost: 127.0.0.1:8000 er-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/115.0 cept: ext/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 cept-Language: en-US,en;q=0.7,ml;q=0.3 cept-Encoding: gzip, deflate erer: http://127.0.0.1:8000/secret ookie: csrftoken=LCEKDPRKqzZKjzqVQmGp8vxq3Eqp3Wq6; sessionid=66p3imvh%9s6z5q9vshkcmckh; h_toFezIWA8Wb5WkaofxTY1nBhS4XzTqLs50hVziw_posthog= 7B#22distinct_id#22+3A#2218944b1837dc65-04cb8ccb6691d8-d505429-14401-18944b1837e1347#22+C124device_id#22+3A#2218944b1837dc65-04cb8ccb6691d8-d505429-14400-anonymous#22+C022extension_version#22+3A#221.5.5#22+C122124session_recording_enabled_server_side#22+3Afalse#20+22+24auto_capture_disabled#24enabled_feature_flags#22+3A#221.5.5#22+C122124active_feature_flag#22+3A#5B#5D#20+22+24sourcing#22+3Afalse#20+22+24company-edit#22+3Afalse#20+22+24feature_flag_payloads#22+3A#7B#7D#7D;essages: iX1L9qC9uK2U1lC3NhZZU1LDAsMjUsIlJ1Z2l2dHJhdGlvb1BzdWNjZXNsZnVsLiLsJ4XQ1qgDxJ0TQRQXrvUuaQmBD7vfCxlrz10Z2CqbVnhNkobsNhs8dc pgrade-Insecure-Requests: 1 ec-Fetch-Dest: document ec-Fetch-Mode: navigate ec-Fetch-Site: same-origin ec-Fetch-User: ?1 onnection: close -Host: admin.localhost:8000			

Open browser

Comment this item

HTTP/1

Inspector

Request query parameter

0

Request body parameter

0 ✓

Request cookie:

4 ✓

Request headers

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	>

A screenshot of a web browser window displaying the PyGoat challenge interface. The URL bar shows the address as 127.0.0.1:8000/secret. The browser's toolbar includes icons for back, forward, search, and other common functions.

The main content area features a blue header bar with the PyGoat logo and navigation links: "Theme" and "Logout".

The central message is "Success. You have the secret" followed by the secret key "S3CR37K3Y". Above this message is a teal button labeled "Secret Key".

The left sidebar contains five dropdown menu items:

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

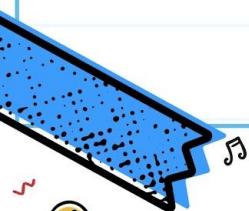
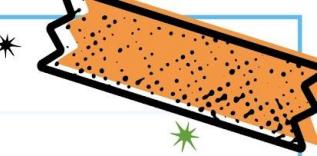
In the bottom right corner of the main content area is a teal button labeled "Back to Lab Details".



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.



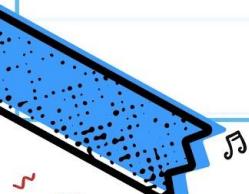
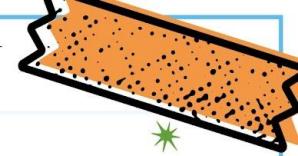


In Real-World

Facebook Bug: Delete any Facebook page [[source](#)]

Found by Arun Suresh Kumar (2016)

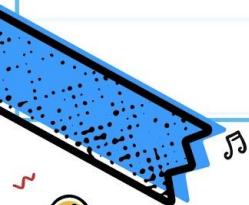
- The misconfiguration vulnerability could have allowed an attacker to delete any facebook page.
- An attacker could intercept a deletion request (for their own page) sent from the front-end and swap the id to any page id to delete it.
- The users' permission to manage the page was not checked in the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





Pygoat Lab 2

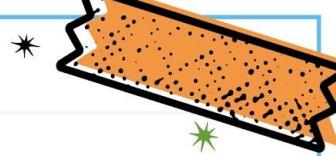
Mission: Get a sensitive env variable stored in the application.



♪

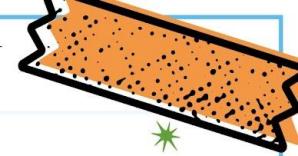


♪





Demo

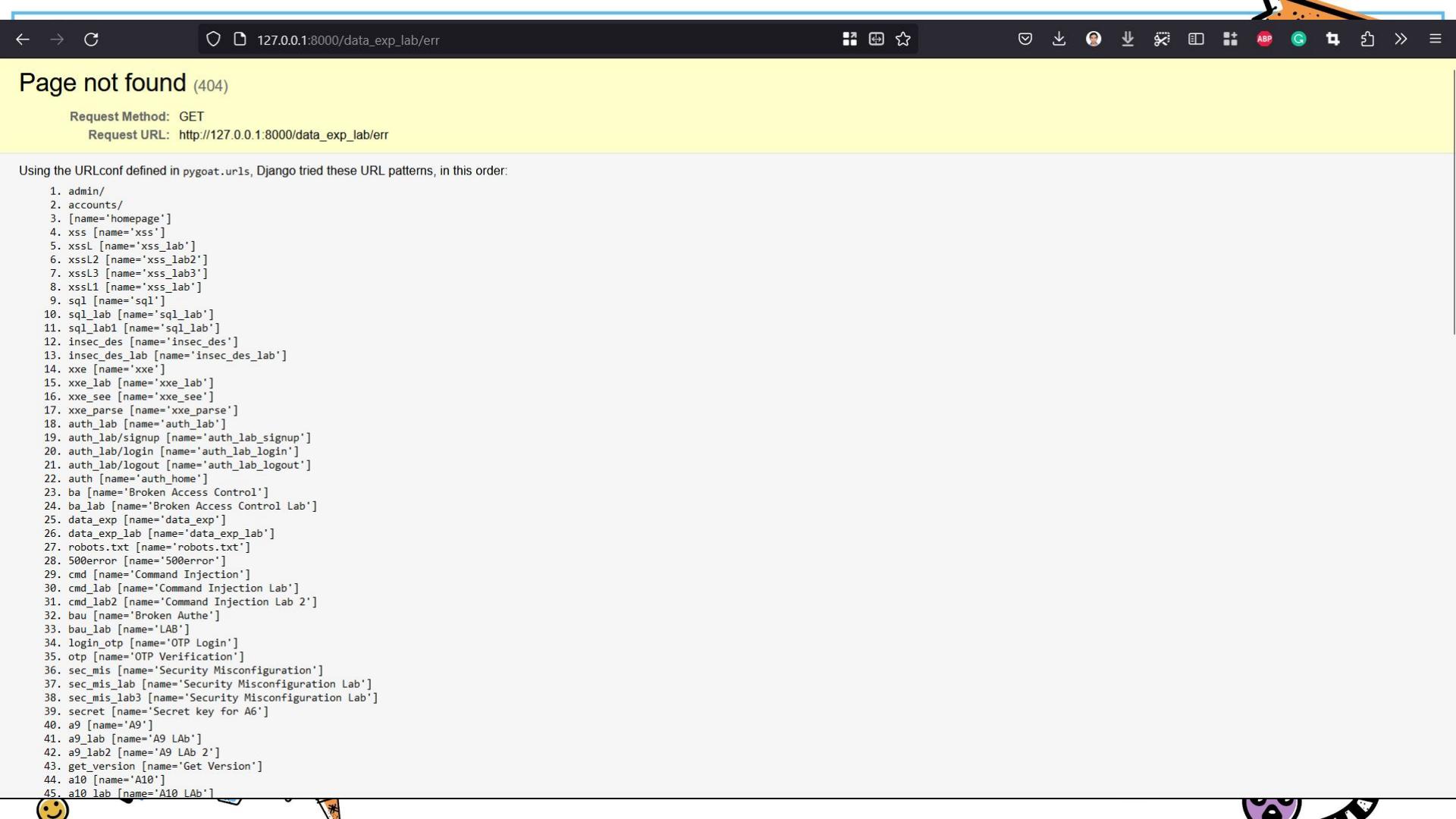


[Home](#)[OWASP TOP 10 2021](#)[SANS 25 Vulns](#)[Mitre top 25 Vulns](#)[OWASP TOP 10 2017](#)[Theme](#) [Logout](#)

Sensitive Data Exposure

Can you find a page to trigger 500 error? Can you find 'SENSITIVE_DATA'?

[Back to Lab Details](#)



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

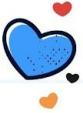
COOKIES

Variable	Value
csrftoken	'tL26KDRKqzZKjZqQVmGp8vxq3Eqp3Wq6'
sessionid	'566p3imfhk9s6z5q9vbzu9h8komcmckh'
ph_foZTeM1Aw8dh5WkaofxTYiInBhS4XzTzRqLs50kVziw_posthog	'%7B%22distinct_id%22%3A%2218944b1837dc65-04cb8cb6691d8-d505429-144000-18944b1837e1347%22%2C%22%24device_id%22%3A%2218944b1837dc65-04cb8cb6691d8-d505429-144000-18944b1837e1347%22%2C%22%24user_state%22%3A%22anonymous%22%2C%22extension_version%22%3A%221.5.%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%2C%22%24enabled_feature_flags%22%3A%7B%22enable-session-recording%22%3Afalse%2C%22%24sourcing%22%3Afalse%2C%22only-company-edit%22%3Afalse%2C%22job-lists%22%3Afalse%7D%2C%22%24feature_flag_payloads%22%3A%7B%27D%7D'
messages	'W1siX19qc29uX211c3NhZ2UiLDAsMjUsI1J1Z21zdHJhdGlvbibZdwNjZXNzZnVsLiIsIiJdXQ:1qKD+J:0TRQXkvUuaQmBD7vfc2x1rz10ZZCqBvNnhKobsNh8dc'
auth_cookie	'eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyIjoibm90X2FkbWluIiwiZXhwIjoxNjg5NjY2MzAwLCJpYXQiOjE2ODk2NjI3MD89.Ls7y0ISayQjPeZgt48QR_sZnvq9bbsQt-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_SETTINGS 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.



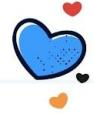


A6: Vulnerable and Outdated Components

The application might be **vulnerable**:

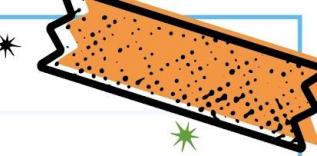
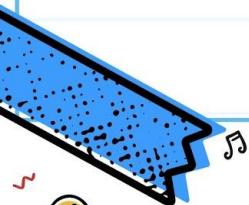
- If the **software used is vulnerable, unsupported, or out of date**. This includes the OS, web/application server, database management system (DBMS), applications, APIs and all components, runtime environments, and libraries.
- If you do not **scan for vulnerabilities** regularly and subscribe to security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





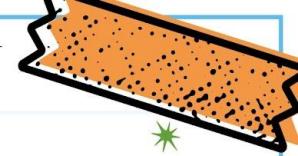
Pygoat Lab - Objective

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





Demo



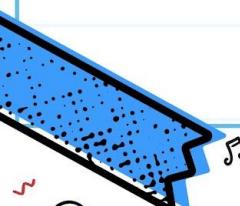


Pygoat Lab - Vulnerable Components

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\n2  !!python/object/new:tuple\n3  - !!python/object/new:map\n4    - !!python/name:eval\n5    - [ print("RCE EXPLOIT!") ]\n6\n7
```





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



Home

OWASP TOP 10 2021

SANS 25 Vulns

Mitre top 25 Vulns

OWASP TOP 10 2017

Yaml To Json Converter

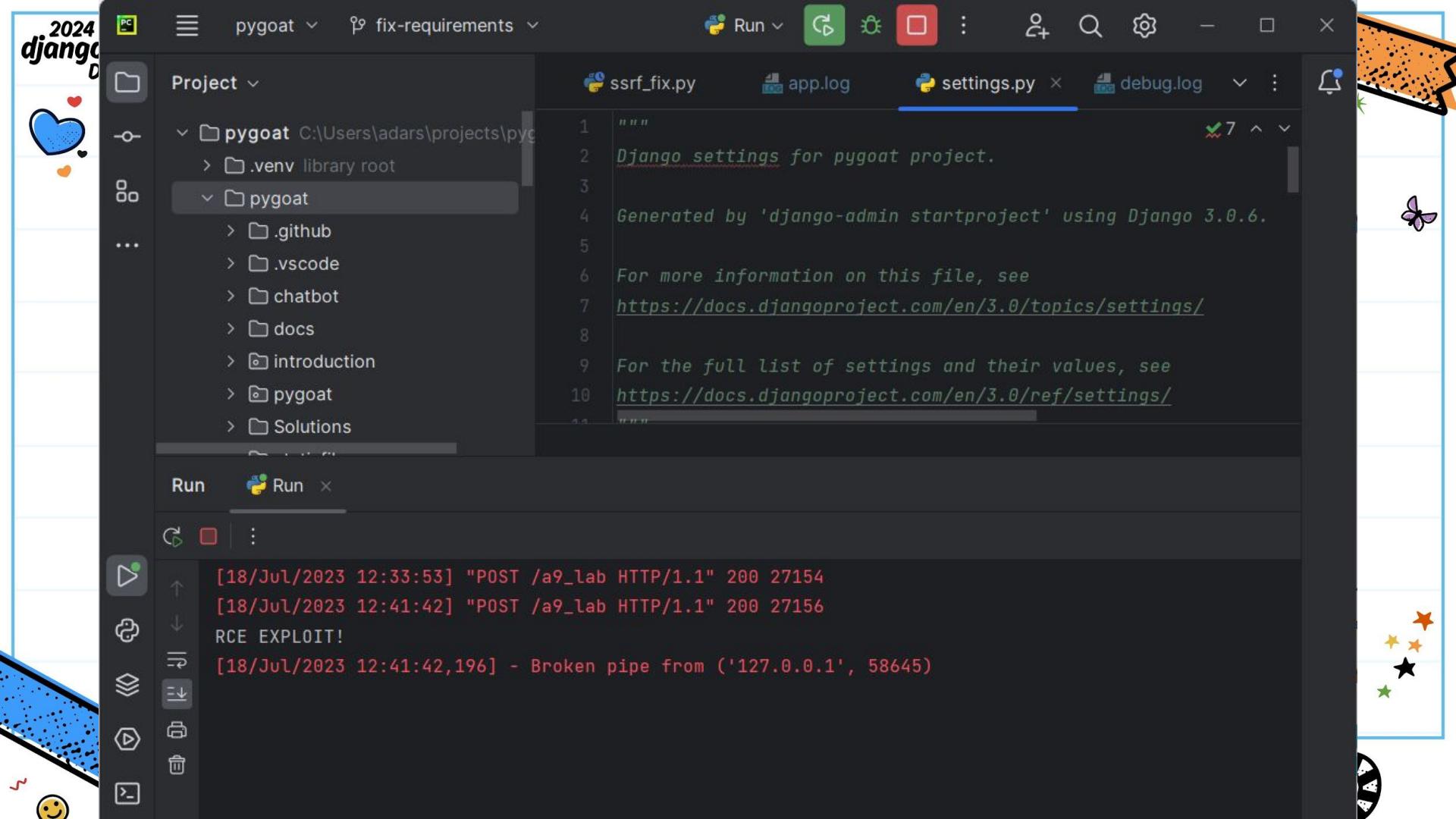
 Browse... No file selected. Upload

Here is your output:

(None,)

Check Django Terminal for Command's output

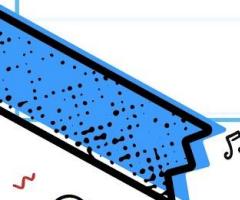
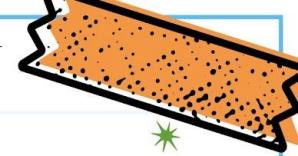
 Get Version Back to Lab Details





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).

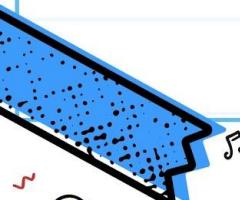




A7: Identification and Authentication Failures

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

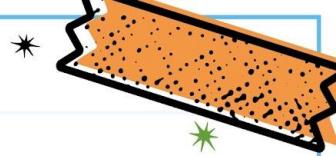
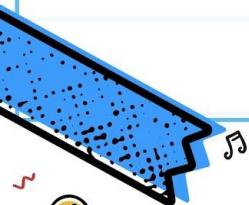
This can let attackers get access to passwords, keys, or tokens, or take advantage of other mistakes in how the app works to pretend to be someone else, either for a little while or forever.





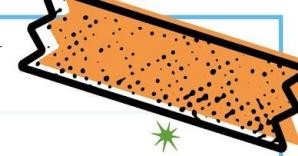
Pygoat Lab - Objective

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.





PyGoat

Home ▾

OWASP TOP 10 2021 ▾

SANS 25 Vulns ▾

Mitre top 25 Vulns ▾

OWASP TOP 10 2017 ▾

Login as Admin

[Login With OTP](#)[Log in](#)[Back to Lab Details](#)



PyGoat

Home

OWASP TOP 10 2021

SANS 25 Vulns

Mitre top 25 Vulns

OWASP TOP 10 2017

Login Through Otp

Send OTP

Enter Your OTP:

Log in

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_foTzTeM1Aw8dh5WkaofxTYiInBhS4XzTzRqLs50kVziw_posthog=%7B%22distinct_id%22%3A%2218944b1837dc65-04cb8cbb6691d8-d505429-144000-18944b1837e1347%22%2C%22device_id%22%3A%2218944b1837dc65-04cb8cbb6691d8-d505429-144000-18944b1837e1347%22%2C%22user_state%22%3A%22anonymous%22%2C%22extension_version%22%3A%221.5.5%22%2C%22session_recording_enabled_server_side%22%3A%22false%2C%22%24autocapture_disabled_server_side%22%3A%22false%2C%22%24active_feature_flags%22%3A%5B%5D%2C%22%24enabled_feature_flags%22%3A%7B%22enable-session-recording%22%3A%22false%2C%22sourcing%22%3A%22only-company-edit%22%3A%22false%2C%22job-lists%22%3A%22false%7D%2C%22%24feature_flag_payloads%22%3A%7B%2D%7D; messages=W1siX19qc29uX21lc3NhZ2UiLDAsMjUsI1JlZ2lzdHJhdGlvbIBzdWNjZXNzZnVsLiIsIiJdXQ:1qKDrJ; OTRQXkvUuaQmBD7vfc2x1rz10ZZCqBvNnhKobsNh8dc; auth_cookie=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2Vybjoibm90X2FkbWluIiwiZXhwIjoxNjg5NjY2MzAwLCJpYXQiOjE2ODk2NjI3MDB9.Ls7y0ISAyQJPeZgt48QR_sZnvq9b0sQt-RfOiH2yeeY; token=gASVAAAAAAAACMEmldudHJvZHvjdGlvbis52awV3c5SMCFRlc3RVc2VylJOUKYGUFSMBWFkbWluleEsAc2Iu
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

 Alerts 0 5 8 7 Main Proxy: localhost:8081

Export

Current Scans 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0



Fuzz Locations

Header: Text

POST http://127.0.0.1
host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5735.135 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.9
Referer: http://
Content-Type: application/x-www-form-urlencoded
Content-Length: 10
Origin: http://127.0.0.1:8000

otp=1234



Type: Numberzz

From: 100

To: 999

Increment: 1

Generate Preview

Save...

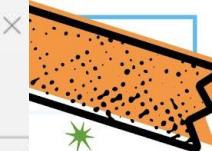
Payloads Preview:

100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115

Cancel

Add

200 OK



Add...

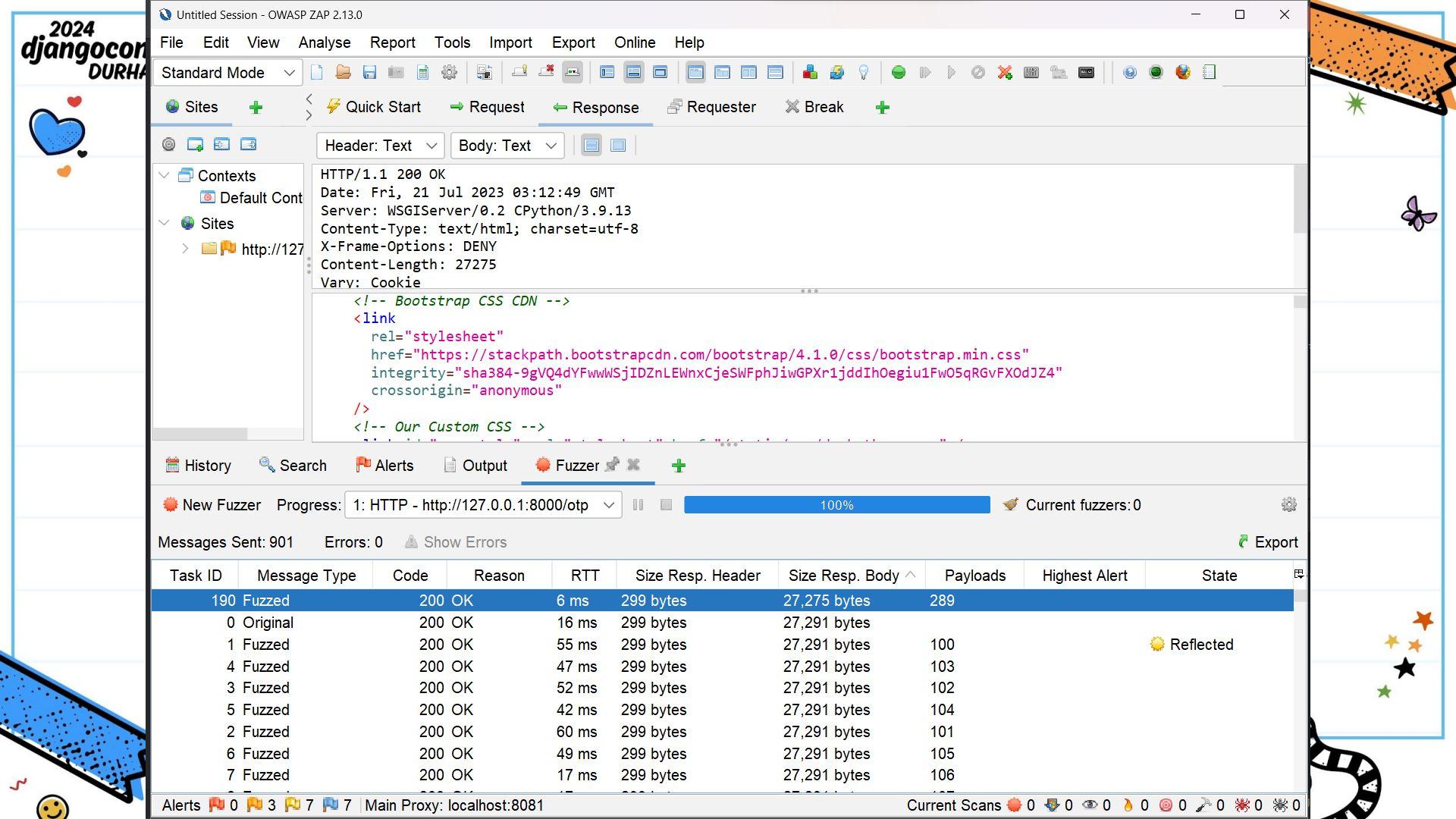
Remove

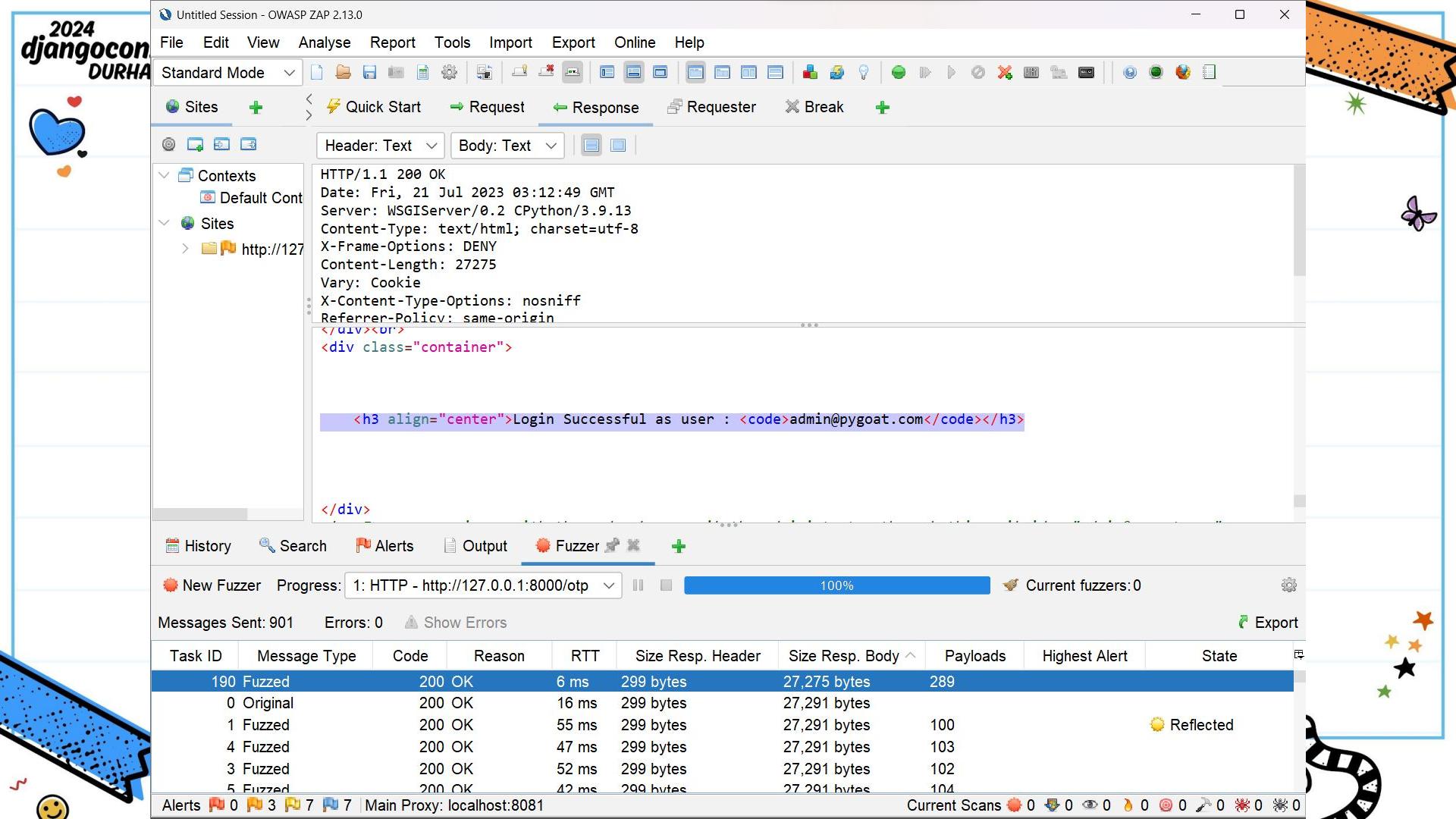
Payloads...

Processors...

?

Set Cancel

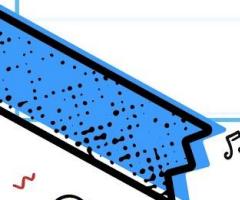
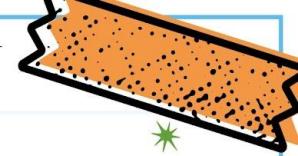






Mitigation

- 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/>

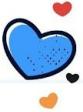
Pygoat Project and Demos:

<https://github.com/adeyosemanputra/pygoat>

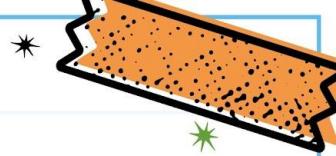
<https://github.com/adarshdigievo/pygoat> [Fork]

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





Learning resources



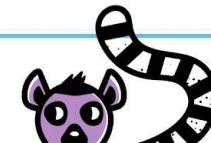
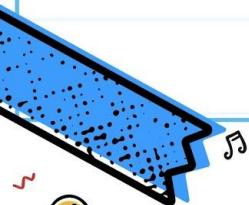
Hacksplaining: <https://www.hacksplaining.com/owasp>



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

Owasp Cheatsheets:

<https://cheatsheetseries.owasp.org/IndexTopTen.html>



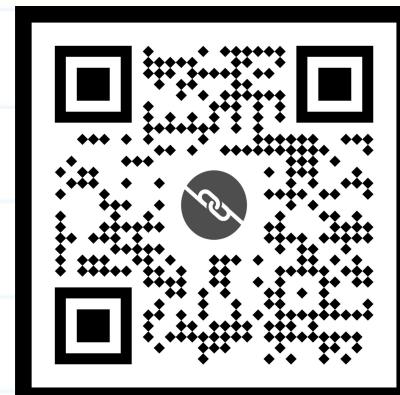


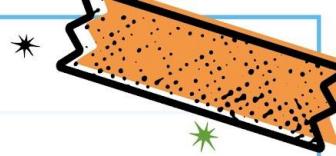
Thank You

Get the talk materials & connect with me



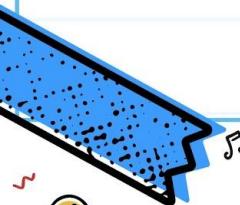
linkhq.co/adarsh

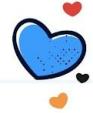




Bonus Slides

Remaining OWASP top 10 vulnerabilities

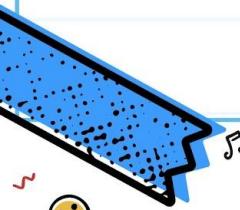


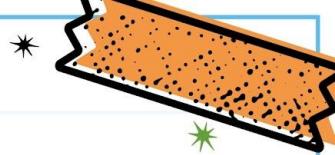


A1 Broken Access Control

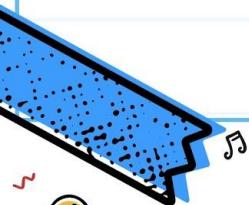
Access control, sometimes called authorization, is how a web application grants **access** to content and functions to some users and not others.

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





PyGoat Lab 1



2024
django

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

2024
django

Broken Access Control.

127.0.0.1:8000/broken_access_lab_1

Home

OWASP TOP 10 2021

SANS 25 Vulns

Mitre top 25 Vulns

OWASP TOP 10 2017

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

Request body parameters

Request cookies

Name	Value
csrfmiddlewaretoken	ekOgXzhz6d1JmKK2loFXUKM1fPzm8YEq
sessionid	oljmdnv2y8w0uvna6kxo6uvrvm26kc

Request headers

Name	Value
Host	127.0.0.1:8000
Content-Length	30
Cache-Control	max-age=0
sec-ch-ua	"NotA-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/537.36 (KHTML, like Gecko) Chrome/112.0.5613.127 Safari/537.36
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
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

20
djan

Broken Access Control.

127.0.0.1:8000/broken_access_lab_1



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



PG

Admins Have the Secretkey

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
ONLY_F0R_4DM
1N5

Back to Lab Details



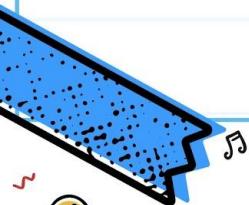
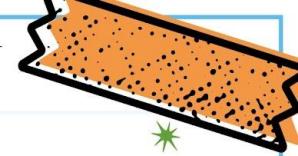
Vulnerable Code

```
if request.COOKIES.get('admin') == "1":  
    return render(  
        request,  
        'Lab_2021/A1_BrokenAccessControl/broken_access_lab_1.html',  
        {  
            "data": "ONLY_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

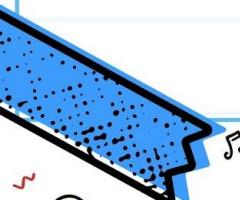
- Except for public resources, **deny by default**.
- Implement access control mechanisms once and **re-use** them throughout the application.
- **Rate limit** API and controller access to minimize the harm from automated attack tooling.
- Stateful **session identifiers** should be invalidated on the 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.

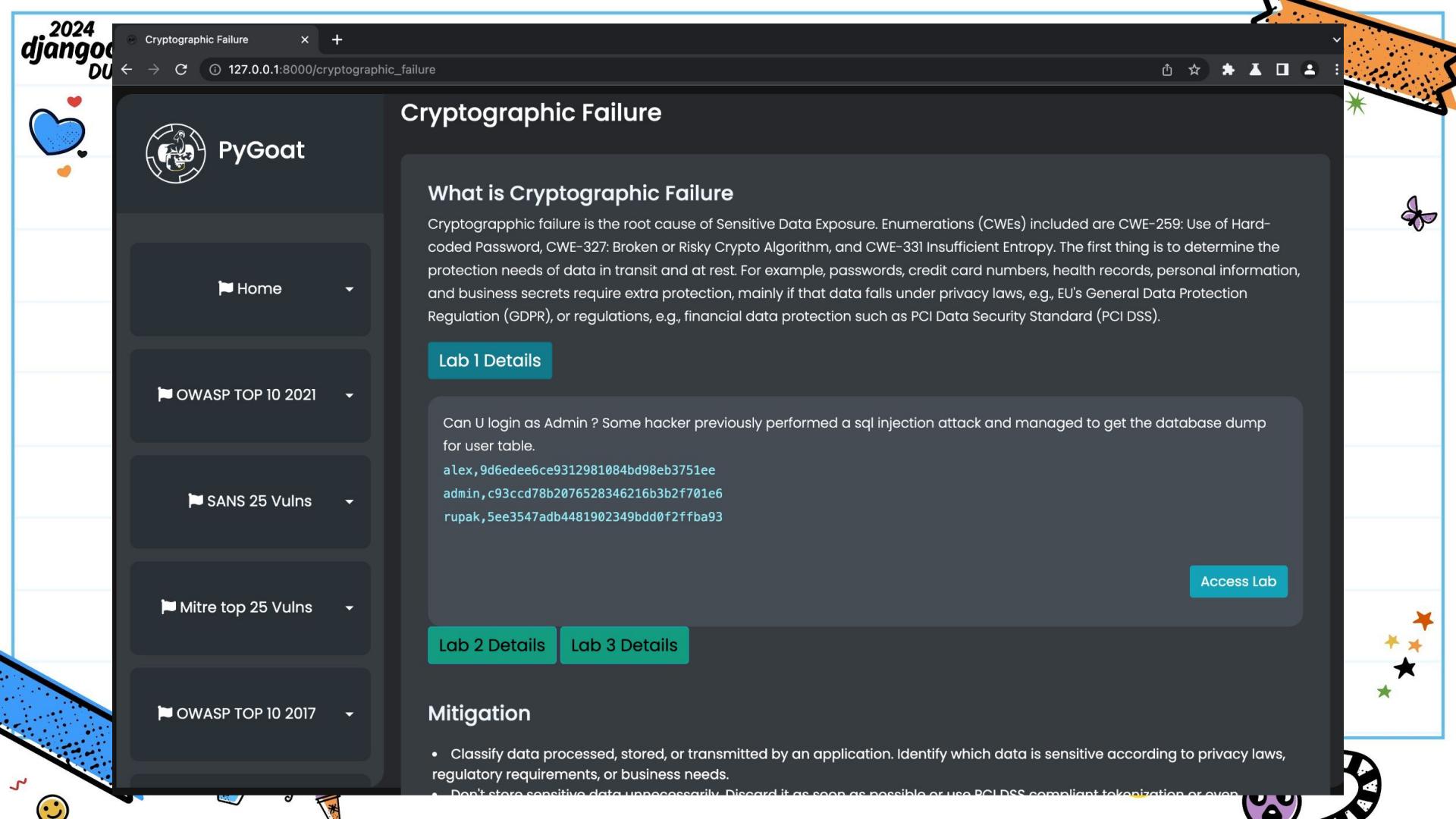




A2 Cryptographic Failures

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.





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

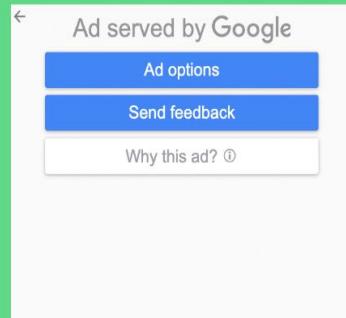


Md5 Decrypt & Encrypt

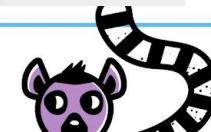
c93ccd78b2076528346216b3b2f701e6

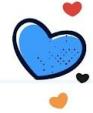
Encrypt

Decrypt



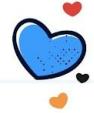
c93ccd78b2076528346216b3b2f701e6 : admin1234





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. 😎



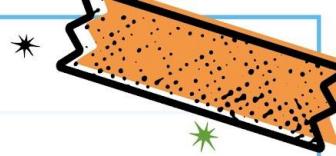
A4: Insecure Design

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

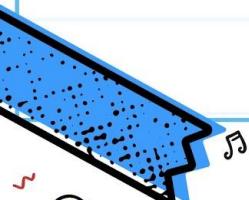


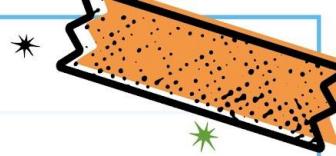
This lab helps you to get an idea of how Insecure Design can result in major Security flaw. In the next page, user can get 5 free tickets for a Movie. But he/she have to wait until all the tickets are sold out. For this particular situation, we can get advantage of the Insecure Design and somehow get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

Hint

Logout and then think.

[Access Lab](#)

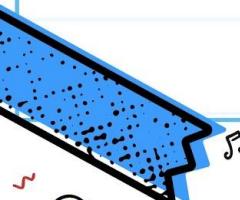




This website is giving everyone free tickets (upto 5 per person).
And the movie will be public when all the tickets will be sold.



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 Tickets

5



Claim

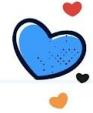
Watch Movie

Tickit

Watch

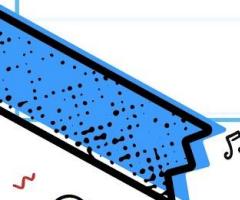
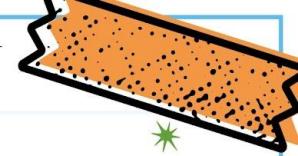
My Tickets

QwLmUHyVoO
qFHNJBKSGc
FfxkTjRDMj
YGaiuFeku
qzauAhPmrw



Mitigation

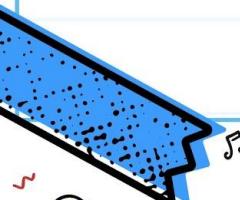
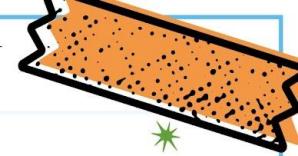
- Establish and use a secure development lifecycle with AppSec professionals to help evaluate and design security and privacy-related controls
- Establish and use a library of secure design patterns or paved road ready to use components
- Use threat modeling for critical authentication, access control, 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)

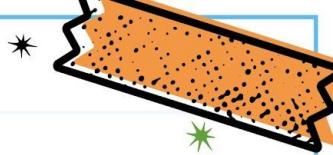




Mitigation

- 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

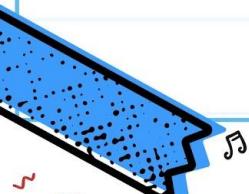
Software and data integrity failures relate to code and infrastructure that does not protect against integrity violations. An example of this is where an application relies upon plugins, libraries, or modules from untrusted sources, repositories, and content delivery networks (CDNs).

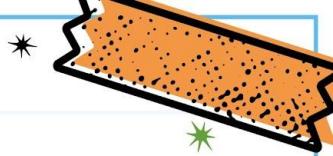


Eg: Insecure Deserialization

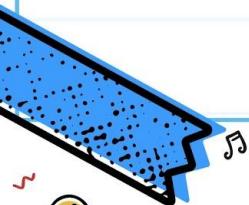
Applications and APIs will be vulnerable if they deserialize hostile or tampered objects supplied by an attacker. This can result in two primary types of attacks:

- 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





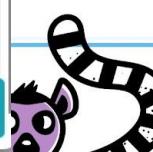
PyGoat

 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

The screenshot shows a Django application interface. On the left, there is a sidebar with a blue header containing a circular logo with a person icon and the letters 'PG'. Below the logo, the sidebar has four items: 'Home', 'OWASP TOP 10 2021', 'SANS 25 Vulns', and 'Mitre top 25 Vulns', each with a flag icon. The main content area has a light blue header with the text 'Here is your download' and a large 'Hey john,'. Below this, there is a red button labeled 'Download' with a white arrow icon. A tooltip for this button says 'a#download_link | 29.3167 x 22.4'. At the bottom right of the main content area is a 'Back to Lab Details' button.

a#download_link | 29.3167 x 22.4

Here is your download

Hey john,

Download

Back to Lab Details

Inspector

Search HTML

```
<div>
  <div>
    <div>
      <div>
        <div>
          <div>
            <div>
              <div>
                <div>
                  <div>
                    <div>
                      <div>
                        <div>
                          <div>
                            <div>
                              <div>
                                <div>
                                  <div>
                                    <div>
                                      <div>
                                        <div>
                                          <div>
                                            <div>
                                              <div>
                                                <div>
                                                  <div>
                                                    <div>
                                                      <div>
                                                        <div>
                                                          <div>
                                                            <div>
                                                              <div>
                                                                <div>
                                                                  <div>
                                                                    <div>
                                                                      <div>
                                                                        <div>
                                                                          <div>
                                                                            <div>
                                                                              <div>
                                                                                <div>
                                                                                  <div>
                                                                                    <div>
                                                                                      <div>
                                                                                        <div>
                                                                                            <div>
                                                                                              <div>
                                                                                                <div>
                                                                                                  <div>
                                                                                                    <div>
                                                                                                      <div>
                                                                                                        <div>
                                                                                                          <div>
                                                                                                            <div>
                                                                                                              <div>
                                                                                                                <div>
                                                                                                                  <div>
                                                                                                                    <div>
                                                                                                                      <div>
                                                                                                                        <div>
                                                                                                                          <div>
                                                                                                                            <div>
                                                                                                                              <div>
                                                                                                                                <div>
                                                                                                                                  <div>
                                                                                                                                    <div>
                                                                                                                                      <div>
                                                                                                                                        <div>
                                                                                                                                          <div>
                                                                                                                                            <div>
                                                                                                                                              <div>
                                                                                                                                                <div>
                                                                                                  <div>
                                                                                                    <div>
                                                                                                      <div>
                                                                                                        <div>
                                                                                                          <div>
                                                                                                            <div>
                                                                                                              <div>
                                                                                                                <div>
                                                                                                                  <div>
                                                                                                                    <div>
                                                                                                                      <div>
                                                                                                                        <div>
                                                                                                                          <div>
                                                                                                                            <div>
                                                                                                                              <div>
                                                                                                                                <div>
                                                                                                                                  <div>
                                                                                                                                    <div>
                                                                                                                                      <div>
                                                                                                                                        <div>
                                                                                                                                          <div>
                                                                                                                                            <div>
................................................................
```

Filter Layout Computed

:hover .cl ▾ Flexbox



Vulnerability

Page is vulnerable to XSS.

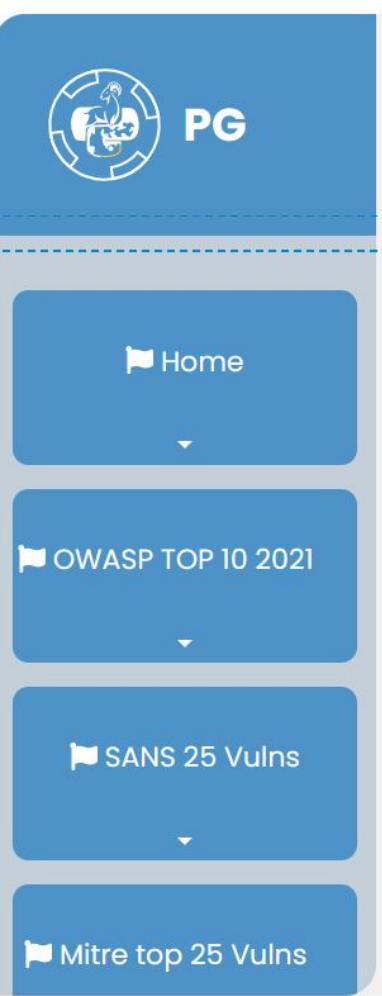
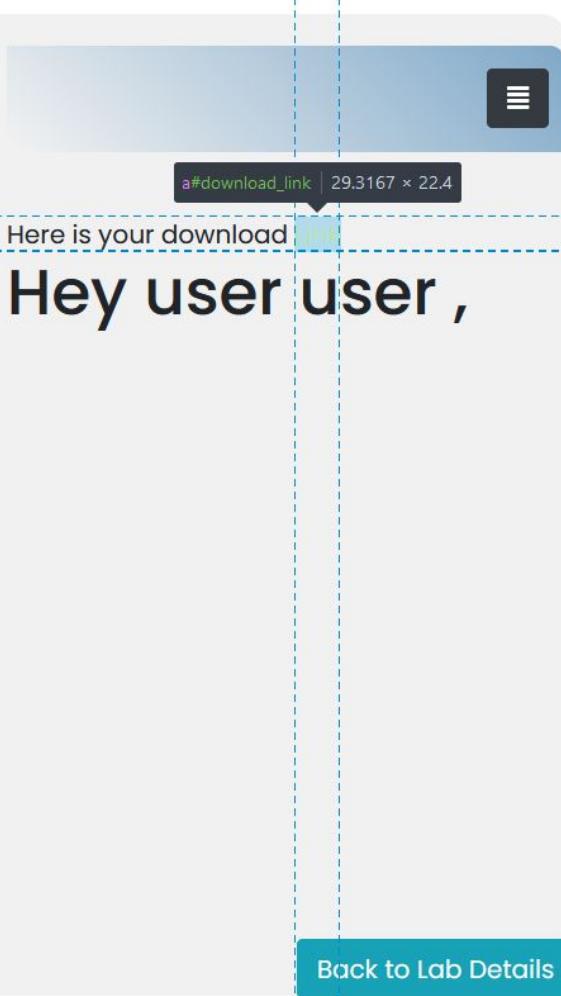
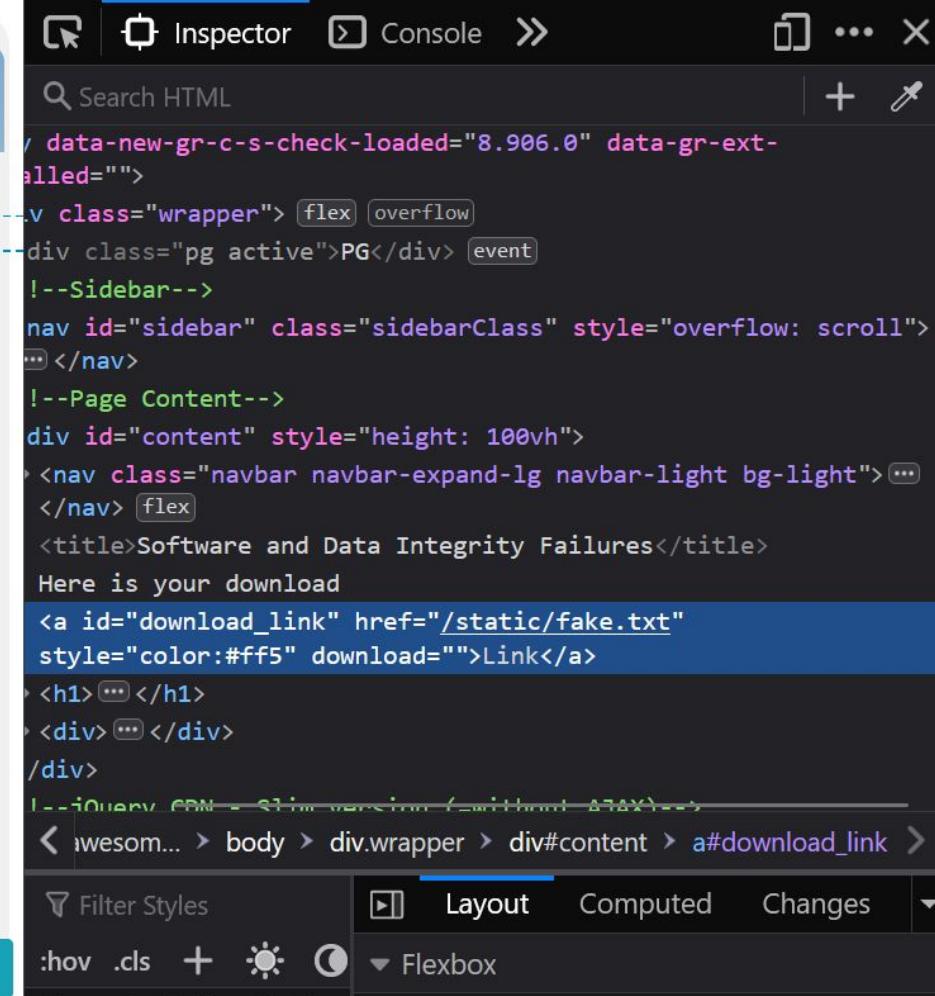
Name is directly printed from the url param.

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



A sidebar on the left side of the page containing four items: "Home", "OWASP TOP 10 2021", "SANS 25 Vulns", and "Mitre top 25 Vulns". Each item has a small icon and a downward arrow indicating it's a dropdown menu.The main content area features a large blue header with the letters "PG" and a circular logo. Below the header, a message says "Here is your download" followed by a link labeled "Link". A large, bold, black h1 tag displays the text "Hey user user ,". At the bottom of this section is a blue button with white text that says "Back to Lab Details".A developer tools window titled "Inspector" showing the HTML code for the page. The code includes a navigation bar, a sidebar, and the main content area. In the content area, it shows the "download_link" element with its href attribute set to "/static/fake.txt". The "Layout" tab is selected at the bottom of the inspector.



Mitigation

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

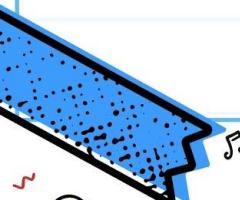
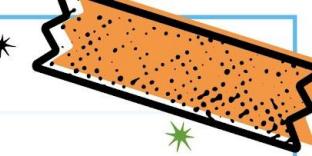
As a user we should always cross-check signatures for verification of Data Integrity. Checksums should be provided for downloads so that it can be cross checked from user end.





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





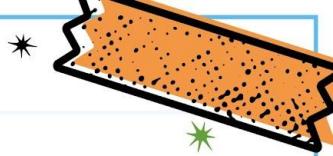
A09: Security Logging and Monitoring

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





Home

OWASP TOP 10 2021

SANS 25 Vulns

Mitre top 25 Vulns

john

Password

Log in

Logs are strictly monitored

[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  WARNING:django.request:Not Found: /a10_lab/debug  
2  WARNING:django.request:Not Found: /favicon.ico  
3  
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
```

```
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 ERROR:root:2023-07-17 15:30:31.781079:127.0.0.1:Python version is outdated. Update from evil.com/py3
9 ERROR:root:2023-07-17 15:20:04.464767:127.0.0.1: dolor sit amet, consectetur adipiscing elit. Aenean commo
10
11
```



Mitigation

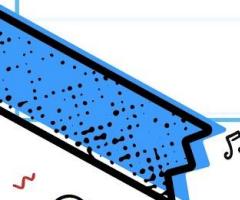
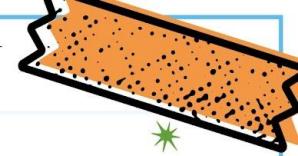
Ensure all login, access control, and server-side input validation failures can be logged with sufficient user context to identify suspicious or malicious accounts and held for enough time to allow delayed forensic analysis.

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.





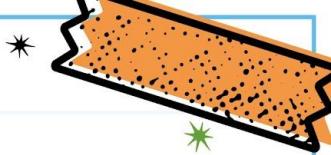
A10: Server Side Request Forgery (SSRF)

- 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



[Home](#)[OWASP TOP 10 2021](#)[SANS 25 Vulns](#)[Mitre top 25 Vulns](#)[OWASP TOP 10 2017](#)

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](#)


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

Read Blog

form | 70.45 x 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

Search HTML

```

<div id="content" style="height: 100vh">
  <nav class="navbar navbar-expand-lg navbar-light bg-light">...</nav>
  <div style="display:flex;flex-direction:column;align-items:center">
    <div style="display:flex;flex-direction:row;align-items:center; margin:15px">
      <form method="post" action="/ssrf_lab">...</form>
      <form method="post" action="/ssrf_lab">...</form>
      <form method="post" action="/ssrf_lab">...</form>
      <form method="post" action="/ssrf_lab">...</form>
      <input type="hidden" name="blog" value="templates/Lab/ssrf/blogs/blog4.txt">
      <button class="btn btn-info" type="submit">Blog4</button>
    </form>
    <div>...</div>
    <div>...</div>
    <div>...</div>
    <button class="col12 btn btn-info" style="position : fixed ; right :330px; bottom : 7px">Hint</button> event
    <div class="lab code">Try to find a .env file</div>
    <button class="col12 btn btn-info" style="position : fixed ; right :200px; bottom : 7px">View Code</button> event
  </div>
  <div>...</div>
  <div>...</div>
  <!--jQuery CDN - Slim version (=without AJAX)-->
<active.fontawesom... > body > div.wrapper > div#content > div > div > form > input >

```

Filter Styles

:hover .cls + ☀️ ☇ ⓘ

Layout Computed Changes

Flexbox



Home

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](#)

Network

All	HTML	CSS	JS	XHR	Fonts	Images	Media	WS	Other
Sta...	M...	Domain	File						
200	PO...	127.0....	ssrf_lab						
200	GET	127.0....	dark-theme.css						
				26 requests	810.19 kB / 335.60 kB transferred			Finish: 850 ms	DOMContentLoaded
	Headers	Cookies	Request	Response	Timings				

POST http://127.0.0.1:8000/ssrf_lab

Status	200 OK
Version	HTTP/1.1
Transferred	30.34 kB (29.91 kB size)
Referrer Policy	same-origin
Request Priority	Highest
Response Headers (434 B)	
Content-Length:	29907
Content-Type:	text/html; charset=utf-8
Cross-Origin-Opener-Policy:	same-origin
Date:	Mon, 17 Jul 2023 05:39:20 GMT
Referrer-Policy:	same-origin
Server:	WSGIServer/0.2 CPython/3.9.13
Set-Cookie:	csrf_token=tL26KDRKqzZKjZqQVmGp8vxq3Eqp3Wq6; expires=Mon, 15 Jul 2024 05:39:20 GMT; Max-Age=3144960; Path=/; SameSite=Lax
Vary:	Cookie
X-Content-Type-Options:	nosniff

Raw



Home

Read Blog

[Blog1](#) [Blog2](#) [Blog3](#) [Blog4](#)

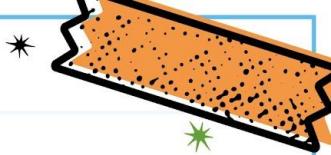
The purpose is to give both developers and testers a platform for learning how to test applications and how to code securely.

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](#)

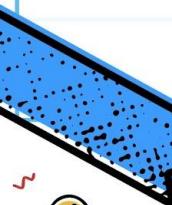
Network traffic captured in browser developer tools:

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
				26 requests	810.19 kB / 335.60 kB transferred	Finish: 850 ms	DOMContentLoaded
Headers Cookies Request Response Timings							
Filter Request Parameters							
Form data							
Raw <input type="checkbox"/>							
csrfmiddlewaretoken: "8a90UVKsRlTyUErbkTZ8z9vFUoYLQtibrL1Kuor27Kl83tHR55vnxuSVNSe0Jf y7"							
blog: "templates/Lab/ssrf/blogs/blog4.txt"							



Vulnerable Code

```
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')
```





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](#)

Search HTML

```
<nav id="sidebar" class="sidebarClass" style="overflow: scroll"></nav>
<!--Page Content--&gt;
&lt;div id="content" style="height: 100vh"&gt;
  &lt;nav class="navbar navbar-expand-lg navbar-light bg-light"&gt;&lt;/nav&gt;
  &lt;title&gt;SSRF LAB&lt;/title&gt;
  &lt;div style="display:flex;flex-direction:column;align-items:center"&gt;
    &lt;div&gt;&lt;/div&gt;
    &lt;div style="display:flex;flex-direction:row;align-items:center; margin:15px"&gt; &lt;flex&gt;
      &lt;form method="post" action="/ssrf_lab"&gt;&lt;/form&gt;
      &lt;form method="post" action="/ssrf_lab"&gt;&lt;/form&gt;
      &lt;form method="post" action="/ssrf_lab"&gt;&lt;/form&gt;
      &lt;form method="post" action="/ssrf_lab"&gt;
        &lt;input type="hidden" name="csrfmiddlewaretoken" value="71DqB3hTZA7aD7av75VrR17DByZiwxAZqJWmbwYtfZWKWqbShrGPmu" /&gt;
        &lt;input type="hidden" name="blog" value="views.py" /&gt;
        &lt;button class="btn btn-info" type="submit"&gt;Blog4&lt;/button&gt;
      &lt;/form&gt;
    &lt;/div&gt;
    &lt;div&gt;&lt;/div&gt;
  &lt;/div&gt;
  &lt;button class="coll2 btn btn-info" style="position : fixed ; right : bottom : 7px"&gt;Hint&lt;/button&gt; event
  &lt;div class="lab code"&gt;Try to find a .env file&lt;/div&gt;
&lt;/div&gt;</pre>


Filter Styles Layout Computed Changes



:hov .cls + ☀️ ☔️ 📁 Flexbox


```



Home

OWASP TOP 10 2021

SANS 25 Vulns

Mitre top 25 Vulns

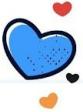
OWASP TOP 10 2017

Read Blog

[Blog1](#) [Blog2](#) [Blog3](#) [Blog4](#)

```
import hashlib from django.shortcuts import render,redirect from django.http import HttpResponseRedirect, 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 decarator 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 request.user.is_authenticated: return render(request, "Lab/XSS/xss.html") else: return redirect('login')
```

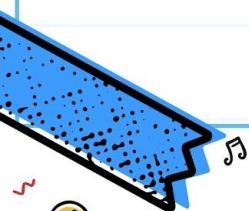
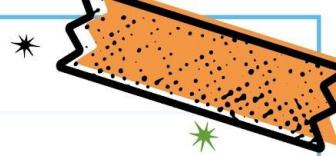
[Hint](#) [View Code](#) [Back to Lab Details](#)

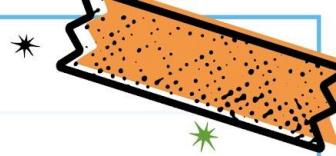


Mitigation

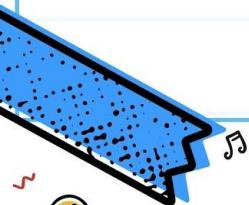
From Application layer:

- 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

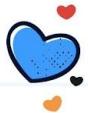




```
def ssrf_lab(request):
    if request.user.is_authenticated:
        if request.method == "GET":
            return render(request, "Lab/ssrf/ssrf_lab.html", {"blog": "Read Blog About SSRF"})

    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')
```



Thank You

Get the talk materials & connect with me



linkhq.co/adarsh

