Network Administration/System Administration (NTU CSIE, Spring 2024) Homework #12 - Security (Part I)

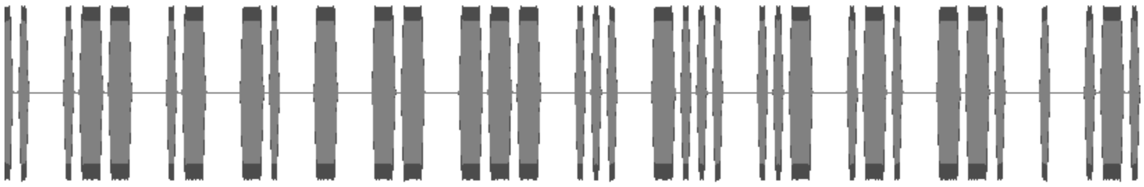
B12902110 呂承諺

June 8, 2024

1 Cryptography

(a) **Steps**

(1) Download 神祕的聲音.wav.



(2) Inspect the waveform of the audio file, and it looks like Morse code.

```
.. .-- .- -. - -- --- ... -... ..- .-. --. . .-.
```

(3) Go to Morse Code Translator | Morse Code World and paste in the Morse code.

Flag `HW12{IWANTMOSBURGER}`

(b) **Steps**

(1) Download 門上的密文.txt.

```
uj12zbf_ohetre_vf_gur_orfg
```

(2) According the problem's hint and the chiphertext's format, it looks like a Caesar cipher. Therefore, code a simple C++ program to try out all the possible rotations.

```cpp
#include <iostream>
#include <string>

int main() {
  std::string s;
  std::cin >> s;
  for (int i = 0; i < 26; ++i) {
    std::cout << "rotate " << i << ": " << s << std::endl;
    for (char& c : s) {
      if (!isalpha(c)) {
        continue;
      }
      if (c == 'z') {
        c = 'a';
      } else {
        ++c;
      }
    }
  }
  return 0;
}
```

(3) Compile and run our program. We obtain the flag with rotation 13.

```
$ g++ -o caesar_cipher caesar_cipher.cpp
$ ./caesar_cipher < 門上的密文.txt
rotate 0: uj12{zbf_ohetre_vf_gur_orfg}
rotate 1: vk12{acg_pifusf_wg_hvs_psgh}
...
rotate 13: hw12{mos_burger_is_the_best}
...
```

Flag `hw12{mos_burger_is_the_best}`

References

- Caesar cipher - Wikipedia

(c) **Steps**

(a) Download blaise 的紙條.txt.

```
Xolffp, O'zv nswo wigzzzu kpgv SSJ NIJHYI, gru U bguctk cfg kson juqv YCK COIMII fcg. Jz zz'w eah
↪  lpi dagy ffgvvck, afgzv zil hi rnzw ui gogb gd kpgv uj ktsas xvrmtucmt vlxkvdg xpl dk? M kdiku
↪  sfav kmglf, mf livx tjfy ku gyackf qyogyqjws zcgzfdg qpo knmew W'v mcbk.

Pfayaoa wuvnmfv ui tgxttwfh og gru qbbpsztk r sfwbn dkec fcyfnykv jacf. Myk si bzco jz Z iee ffwbn
↪  puy ka ggnyknmes wf sykave!
TK12{tmuzyi_nmblt_GFY_FLDUWS_nfu}

MOSBURGER
```

(b) According to the hints, this piece of text may be encrypted in Vigenère cipher with MOSBURGER as the key. Code a simple C++ program to decrypt the text.

```cpp
#include <iostream>
#include <string>

int main() {
  const std::string kKey = "MOSBURGER";
  size_t key_i = 0;

  std::string s;
  while (std::getline(std::cin, s)) {
    for (char& c : s) {
      if (!isalpha(c)) {
        continue;
      }
      char base;
      if (islower(c)) {
        base = 'a';
      } else {
        base = 'A';
      }
      c = base + ((c - base) - (kKey[key_i] - 'A') + 26) % 26;
      key_i = (key_i + 1) % kKey.size();
    }
    std::cout << s << std::endl;
  }
  return 0;
}
```

(c) Compile and run our program. The flag is in the output.

```
$ g++ -o vigenere_cipher vigenere_cipher.cpp
$ ./vigenere_cipher < blaise 的紙條.txt
Lately, I've been craving some MOS BURGER, and I notice you want some MOS
 ↪  BURGER too.
...
HW12{blaise_wants_MOS_BURGER_too}
```

Flag `HW12{blaise_wants_MOS_BURGER_too}`

References

- Vigenère cipher - Wikipedia

(d) **Steps**

(1) Download 小福門口的告示.txt.

```
Jgmh Dmovgj Wvsqzaghs,

F lzkg qlfs agssmtg rfxjs yzv ugoo. F ma uhfqfxt qz mjjhgss mx vxrzhqvxmqg fxwfjgxq qlmq zwwvhhgj
↪  mq zvh AZS NVHTGH ozwmqfzx hgwgxqoy. Fq lms wzag qz zvh mqqgxqfzx qlmq qlg rhzxq jzzh zr zvh
↪  hgsqmvhmxq ums nhzigx, hgsvoqfxt fx fxwzxdgxfgxwg qz amxy zr yzv, zvh ozymo wvsqzaghs. Kogmsg
↪  mwwgkq ay sfxwghgsq mkzoztfgs rzh mxy jfshvkqfzx qlfs amy lmdg wmvsgj.

...

Umha hgtmhjs,
mhwlgh wlgx
Amxmtgh, AZS NVHTGH
```

(2) Observe the text, and we discover that "AZS NVHTGH" is repeated many times. We suspect that it maps to "MOS BURGER".

So, we code a simple C++ program that handles character substitution. We will update the mapping gradually as we guess out more and more characters.

```cpp
#include <iostream>

int main() {
  // Gradually updated.
  const char kMap[27] = "MBCDEFERIJKLMBOPQRSGUUWXYO";

  std::string s;
  while (std::getline(std::cin, s)) {
    for (char& c : s) {
      if (!isalpha(c)) {
        continue;
      }
      char base;
      if (islower(c)) {
        base = 'a';
      } else {
        base = 'A';
      }
      c = base + kMap[c - base] - 'A';
    }
    std::cout << s << std::endl;
  }
  return 0;
}
```

(3) The decrypted for now looks something like this.

```
Jemr Dmouej Wusqomers,

F loke qlfs messmge rfxjs you ueoo. F mm urfqfxg qo mjjress mx uxrorquxmqe fxwfjexq qlmq owwurrej
↪  mq our MOS BURGER oowmqfox rewexqoy. Fq lms wome qo our mqqexqfox qlmq qle rroxq joor or our
↪  resqmurmxq ums broiex, resuoqfxg fx fxwoxdexfexwe qo mmxy or you, our ooymo wusqomers. Koemse
↪  mwwekq my sfxweresq mkooogfes ror mxy jfsrukqfox qlfs mmy lmde wmusej.
...
LU12{qle_broiex_rroxq_joor_or_MOS}
...
Umrm regmrjs,
mrwler wlex
Mmxmger, MOS BURGER
```

Referring back to the original message, we suspect the following mappings.

Chipertext	Plaintext
Jgmh	Dear
Wvsqzaghs	Customers
agssmtg	message
LU	HW
Umha hgtmhjs	Warm regards

(4) Decrypt the ciphertext with the updated mapping. Now it looks something like this.

```
...

As a gesture or goodwfoo, we'd ofie to orrer a 10% dfscouxt ox your xeet meao wfth us. ...

Thaxi you oxce agafx ror your uxderstaxdfxg, axd we oooi rorward to weocomfxg you baci to a ruooy
↪  okeratfoxao axd fmkroded MOS BURGER eekerfexce dery soox.
```

The last two paragraphs actually give us quite a lot of information.

Chipertext	Plaintext
orrer	offer
dfscouxt	discount
xgeq	next
Qlmxi	Thank
yzv	you

Chipertext	Plaintext
zxwg	once
mtmfx	again
rzh	for
vxjghsqmxjfxt	understanding
ugowzafxt	welcoming

(5) Now the message is almost cracked.

```
Dear Dalued Customers,

... Klease accekt my sincerest akologies for any disruktion this may hade caused.

...

We recognipe that this issue has been disakkointing

...

To address this issue kromktly, we'de arranged for a krofessional rekair. ... If you hade any
↪  concerns, cuestions, or feedback, klease feel free to reach out to me directly or skeak with
↪  any member of our team.
```

Chipertext	Plaintext
hgwztxfpg	recognize
Kogmsg	Please
mwwgkq	accept

Chipertext	Plaintext
cvgsqfzxs	questions
ug'dg	we've

(6) Now we have figured out the mapping. "ABC...XYZ" maps to "MBQVXIERKDPHABLZTFSGWUCNYO". We use this to crack the flag.

Flag `HW12{the_broken_front_door_of_MOS}`

(e) **Steps**

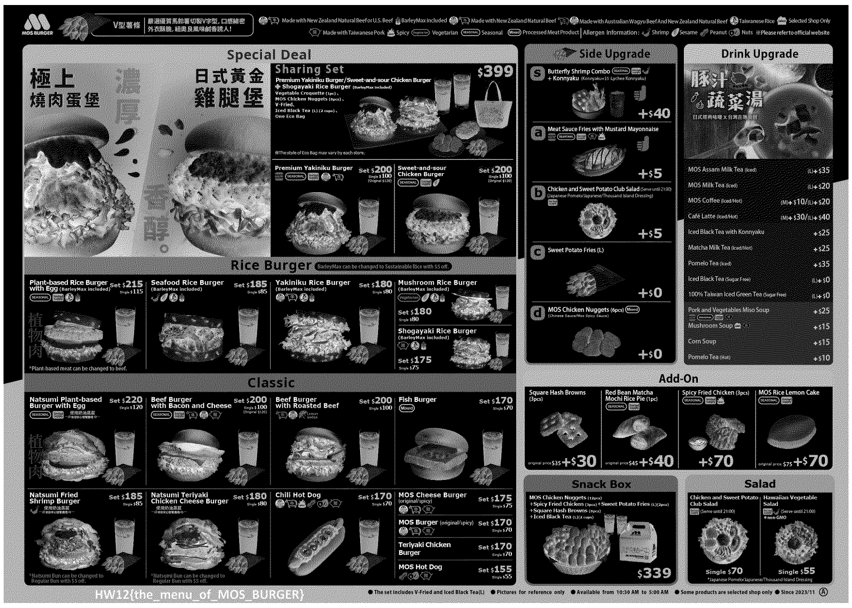
(1) Download `english-menu_A.png` and `english-menu_B.png`. We discover that they have the exact same resolution, so it make come down to some kind of filtering.

Furthermore, the two images only contain black and white pixels. We suspect that the two images can be merged pixel by pixel with bitwise XOR.

(2) Code a simple Python script that performs pixel-wise XOR with Pillow and numpy.

```python
from PIL import Image
import numpy as np


def main():
    a = Image.open('english-menu_A.png')
    b = Image.open('english-menu_B.png')
    a_array = np.asarray(a)
    b_array = np.asarray(b)

    xor_array = np.bitwise_xor(a_array, b_array)
    xor_image = Image.fromarray(xor_array)
    xor_image.save('xor.png')


if __name__ == '__main__':
    main()
```

Result The resulting image indeed contains the flag.



Flag `HW12{the_menu_of_MOS_BURGER}`

References

- Image Module - Pillow (PIL Fork) 10.3.0 documentation

2 DNS security

(a) **DNS Spoofing** is a general term that refers to any attempt to hijack the DNS service and send out malicious DNS records to clients. If DNS services are compromised, incorrect IP addresses for domains could flow around the Internet, and network traffic could be directed to unintended places. This could cause other services to become inaccessible, or even expose data to attackers.

(b) **DNSSEC** allows DNS records to be signed with public key cryptography, providing a method to authenticate responses returned from servers. A chain of trust is built from the root server down to the authoritative server. If an attacker is trying to spoof DNS records, it would not have the legitimate signature, and the recursive resolver or client can simply reject the response.

(c) **DNS Cache Poisoning** refers to an attempt to insert malicious records into the DNS cache of a client or DNS resolver. This would cause future queries to return the attacker-determined location, and network traffic could be directed unwanted locations.

(d) An **NXDOMAIN attack** is a type of denial-of-servuce attack, where the attacker floods a DNS server with queries that ask for nonexistent domains. This could overload the server or fill up the server's cache, and the service may go down.

(e) The following measures could help prevent DNS attacks:

- Opt for secure DNS protocols, such as DNSSEC, DNS over HTTPS, or DNS over TLS, to prevent DNS spoofing attacks.
- Deploy multiple redundant DNS servers with load balancing to prevent DDoS attacks.
- Limit the rate at which queries could be made.
- Restrict the hosts that can access our DNS service. For example, in a school or company, we only need to provide service to clients inside it.

The following measures could help detect DNS attacks:

- If an abnormally large amount of queries are coming from similar sources, this may be a DDoS attack.
- If an abnormally large amount of response are coming from similar sources, this may be a DNS spoofing attack.

References

- DNS spoofing - Wikipedia
- Domain Name System Security Extensions - Wikipedia
- What is DNS cache poisoning? | DNS spoofing | Cloudflare
- What is DNS Security? | DNSSEC | Cloudflare
- DNS Spoofing vs DNS Cache Poisoning - Information Security Stack Exchange

3 CIA Triad & Threat Modeling

(a) (1) Taiwan's household registration database leaks to the dark web.

Confidentiality: Sensitive information isn't kept privately enough.

Availability: According to my previous experiences, the average government employee has too much access to citizens' personal information. All they need is enter your ID number.

(2) NTU Office of Academic Affairs' website is hacked by Chinese attackers.

Confidentiality: There are security weaknesses in our system for those attackers across the Taiwan Strait to exploit.

Integrity: The web pages are maliciously modified by the attackers.

(b) **Assumptions**

- A password is required to login to any account. There is no guest account.

Threat models and countermeasures

- **Threat model**: Someone sneakily access the computer when the owner leaves it without locking the account.

  Countermeasure: Lock the account immediately whenever the owner is not using the computer.

- **Threat model**: Someone boots into the computer with a USB installation media, mounts your filesystem, and access your data. Or even they find a way to extract `/etc/shadow` from the filesystem and crack your password.

  Countermeasure: Encrypt the root filesystem, and use strong login passwords.

(c)

(d) **Assumptions**

- Members of different groups wouldn't see nor hear each other physically.

Threat models and countermeasures

- **Threat model**: Members of different groups use online live chat services to communicate with each other.

  Countermeasure: During the exam, setup firewall rules across the classroom to deny access to popular online chat services.

- **Threat model**: Members of different groups exchange data with services that would be unreasonable for us to ban for the purpose of the exam, such as GitHub or Pastebin.

  Countermeasure: Demand that students record or broadcast their screens during the exam. The screen recordings could be the TAs' reference for checking whether cheating is involved.

4 Web Security

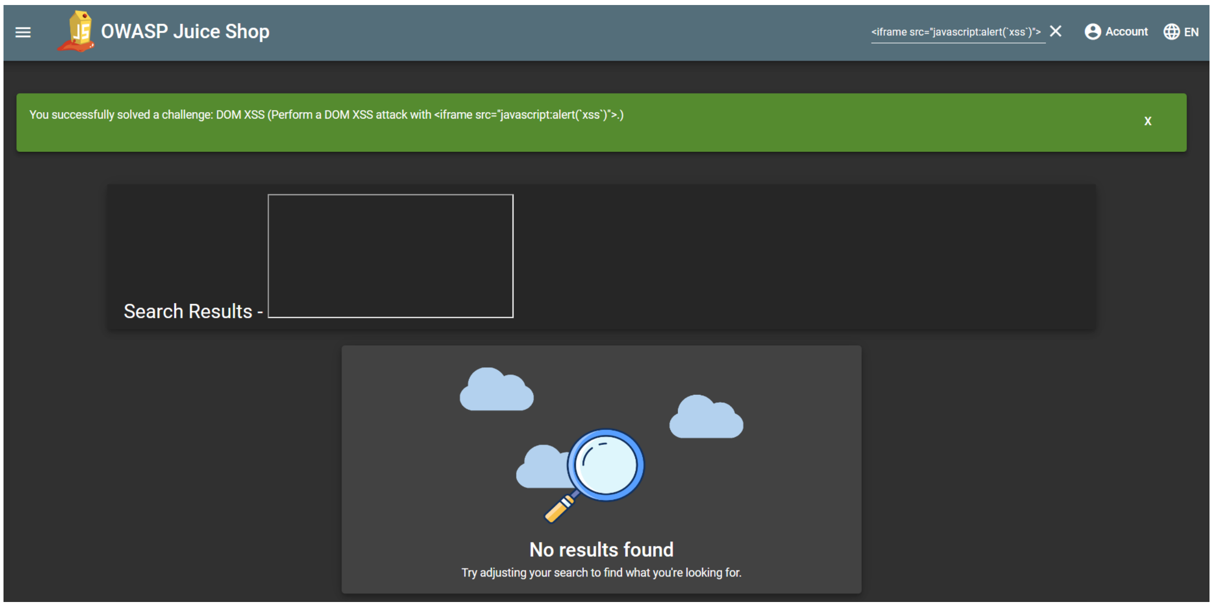
(1) OWASP Juice Shop

(a) DOM XSS

Steps

Search for `<iframe src="javascript:alert(`xss`)">` in the search box in the navbar.

Result


OWASP Juice Shop
<iframe src="javascript:alert(`xss`)">
Account
EN
X
Search Results -
No results found
Try adjusting your search to find what you're looking for.

You successfully solved a challenge: DOM XSS (Perform a DOM XSS attack with <iframe src="javascript:alert(`xss`)">.)

Explanation

This attack is a type of XSS attack, where malicious code is injected into the website. In this example, the search query is inserted into the HTML DOM, so we can forge the search query to contain HTML elements.

To prevent this type of attack, we can escape characters that are related to HTML tags.

References

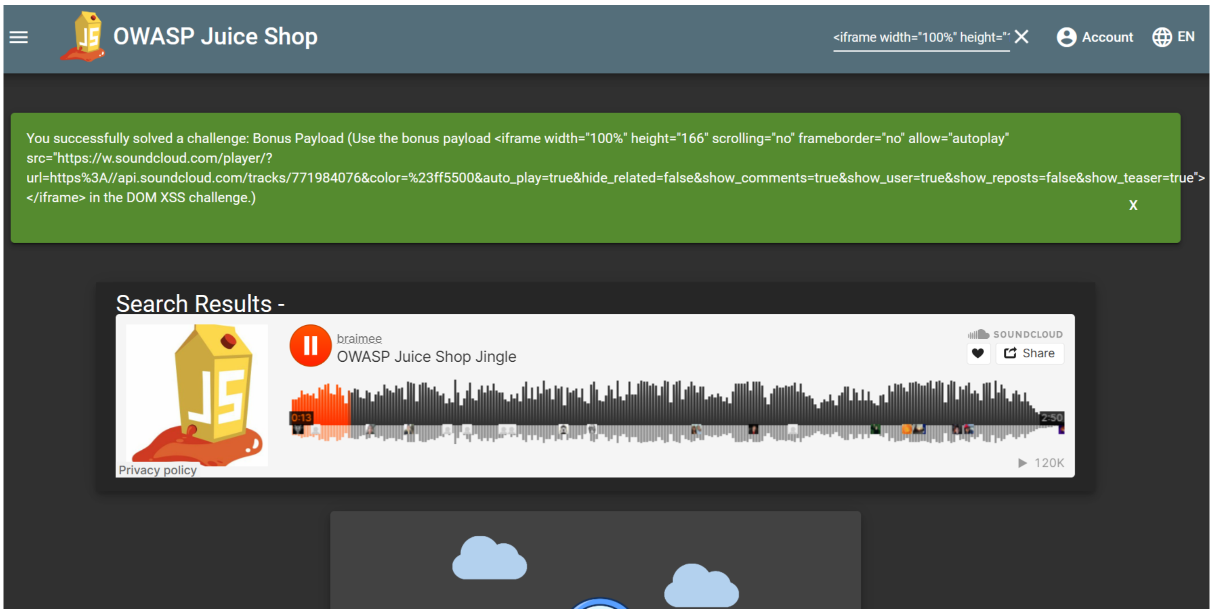
- Cross-site scripting - Wikipedia
- Cross Site Scripting (XSS) :: Pwning OWASP Juice Shop

(b) **Bonus Payload**

Steps

Search for `<iframe width="100%" height="166" scrolling="no" frameborder="no" allow="autoplay" src="https://w.soundcloud.com/player/?url=https%3A//api.soundcloud.com/tracks/771984076&color=%23ff5500&auto_play=true&hide_related=false&show_comments=true&show_user=true&show_reposts=false&show_teaser=true"></iframe>` in the search box in the navbar.

Result



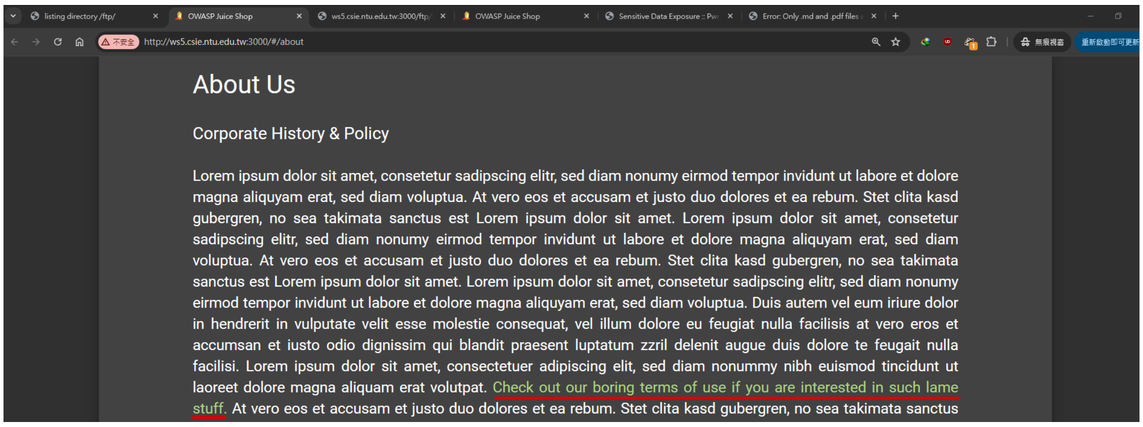
You successfully solved a challenge: Bonus Payload (Use the bonus payload <iframe width="100%" height="166" scrolling="no" frameborder="no" allow="autoplay" src="https://w.soundcloud.com/player/?url=https%3A//api.soundcloud.com/tracks/771984076&color=%23ff5500&auto_play=true&hide_related=false&show_comments=true&show_user=true&show_reposts=false&show_teaser=true"></iframe> in the DOM XSS challenge.)

Explanation Same as (a).

(c) **Confidential Document**

Steps

(i) In the "About Us" (`/#/about`) page, there is a hyperlink to `/ftp/leagl.md`.

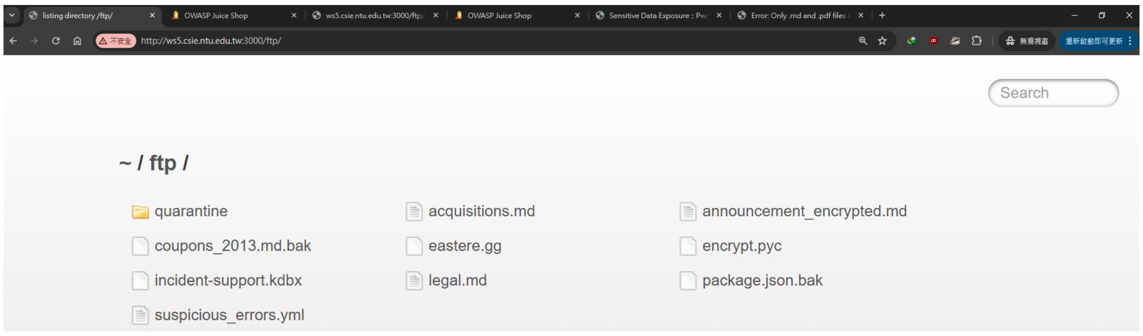

listing directory /ftp/
OWASP Juice Shop
ws5.csie.ntu.edu.tw:3000/ftp/
OWASP Juice Shop
Sensitive Data Exposure :: Pwr
Error: Only .md and .pdf files
不安全
http://ws5.csie.ntu.edu.tw:3000/#/about

About Us

Corporate History & Policy

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Check out our boring terms of use if you are interested in such lame stuff. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus

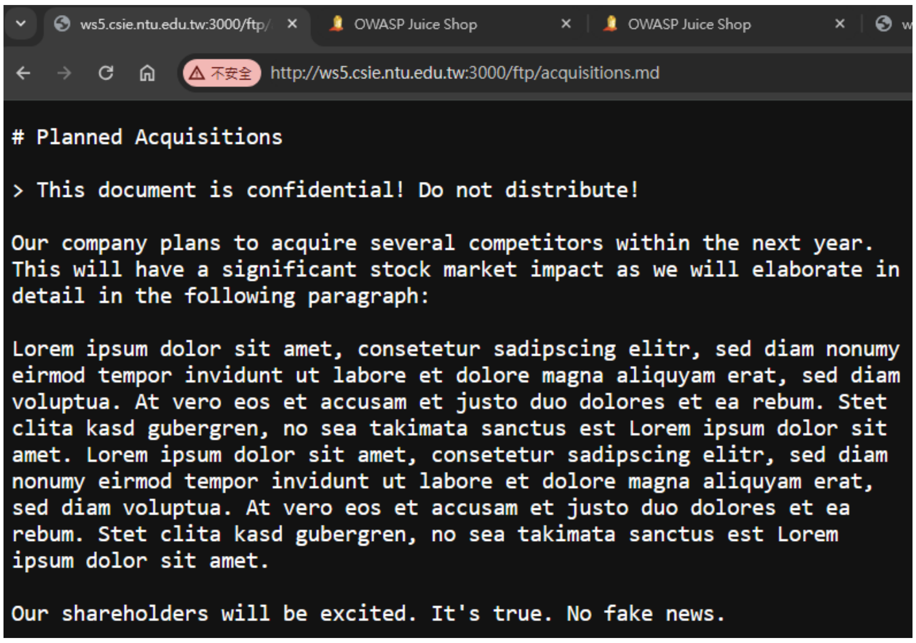
(ii) Access `/ftp/`, and a web page with directoy listings shows up.


listing directory /ftp/
OWASP Juice Shop
ws5.csie.ntu.edu.tw:3000/ftp/
OWASP Juice Shop
Sensitive Data Exposure :: Pwr
Error: Only .md and .pdf files
不安全
http://ws5.csie.ntu.edu.tw:3000/ftp/
Search

~ / ftp /

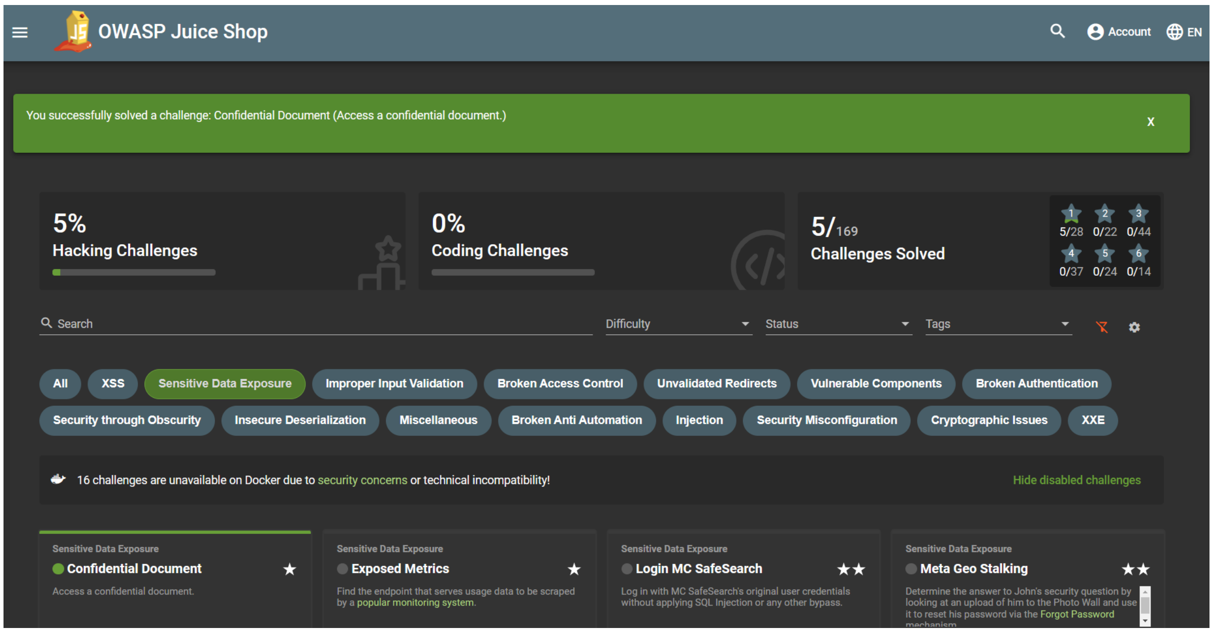
quarantine	acquisitions.md	announcement_encrypted.md
coupons_2013.md.bak	eastere.gg	encrypt.pyc
incident-support.kdbx	legal.md	package.json.bak
suspicious_errors.yml		

(iii) Access `/ftp/acquisitions.md`, and the challenge is solved.


ws5.csie.ntu.edu.tw:3000/ftp/
OWASP Juice Shop
OWASP Juice Shop
不安全
http://ws5.csie.ntu.edu.tw:3000/ftp/acquisitions.md

```
# Planned Acquisitions

> This document is confidential! Do not distribute!

Our company plans to acquire several competitors within the next year.
This will have a significant stock market impact as we will elaborate in
detail in the following paragraph:

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy
eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam
voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet
clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit
amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam
nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat,
sed diam voluptua. At vero eos et accusam et justo duo dolores et ea
rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem
ipsum dolor sit amet.

Our shareholders will be excited. It's true. No fake news.
```

Result



You successfully solved a challenge: Confidential Document (Access a confidential document.)

16 challenges are unavailable on Docker due to security concerns or technical incompatibility!

Sensitive Data Exposure
Confidential Document ★
Access a confidential document.

Sensitive Data Exposure
Exposed Metrics ★
Find the endpoint that serves usage data to be scraped by a popular monitoring system.

Sensitive Data Exposure
Login MC SafeSearch ★★
Log in with MC SafeSearch's original user credentials without applying SQL Injection or any other bypass.

Sensitive Data Exposure
Meta Geo Stalking ★★
Determine the answer to John's security question by looking at an upload of him to the Photo Wall and use it to reset his password via the Forgot Password mechanism.

Explanation

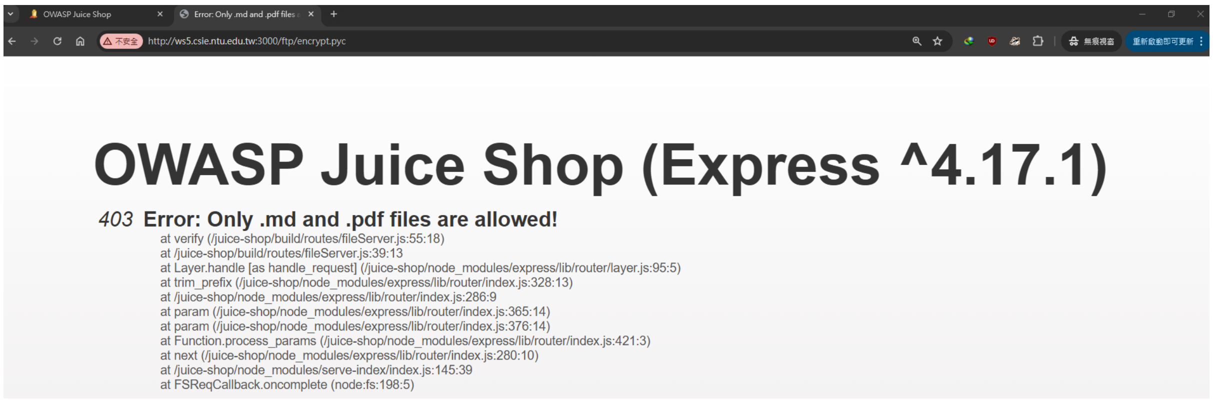
This attack is possible because a web page containing a directory listing is exposed to the public.

To prevent this type of attack, we deny directory listing web pages from being accessed publicly, and minimize the static files being served.

(d) **Error Handling**

Steps

From the directory listing (`/ftp/`), access `encrypt.pyc`. An error page appears.

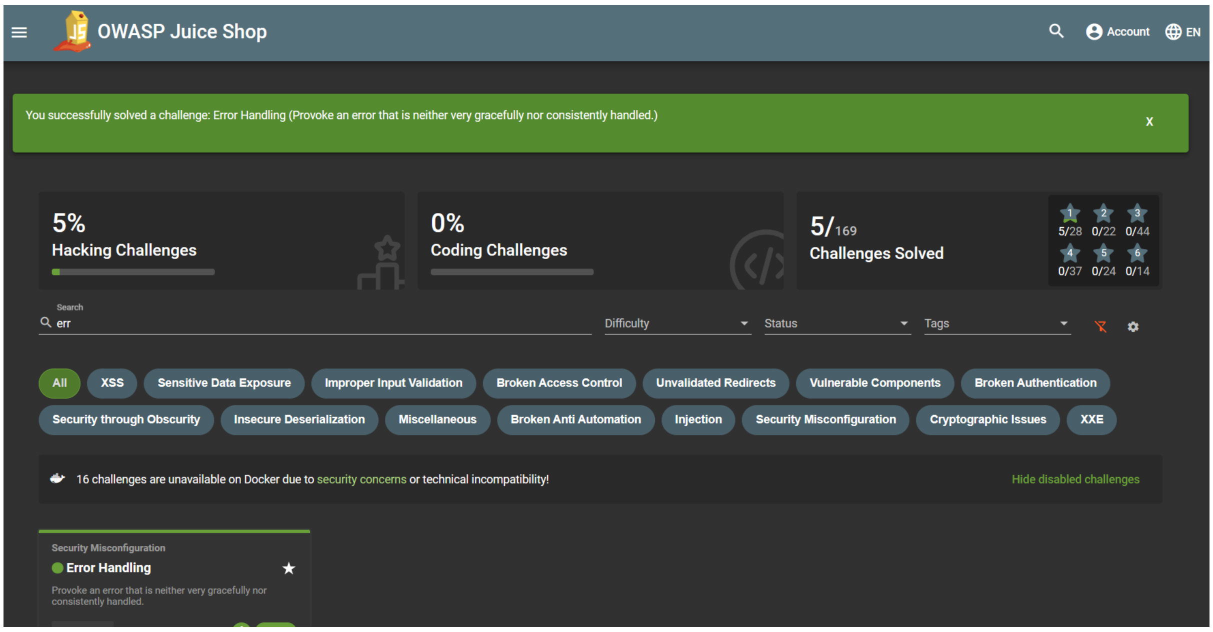


http://ws5.csie.ntu.edu.tw:3000/ftp/encrypt.pyc

OWASP Juice Shop (Express ^4.17.1)

```
403 Error: Only .md and .pdf files are allowed!
    at verify (/juice-shop/build/routes/fileServer.js:55:18)
    at /juice-shop/build/routes/fileServer.js:39:13
    at Layer.handle [as handle_request] (/juice-shop/node_modules/express/lib/router/layer.js:95:5)
    at trim_prefix (/juice-shop/node_modules/express/lib/router/index.js:328:13)
    at /juice-shop/node_modules/express/lib/router/index.js:286:9
    at param (/juice-shop/node_modules/express/lib/router/index.js:365:14)
    at param (/juice-shop/node_modules/express/lib/router/index.js:376:14)
    at Function.process_params (/juice-shop/node_modules/express/lib/router/index.js:421:3)
    at next (/juice-shop/node_modules/express/lib/router/index.js:280:10)
    at /juice-shop/node_modules/serve-index/index.js:145:39
    at FSReqCallback.oncomplete (node:fs:198:5)
```

Result



You successfully solved a challenge: Error Handling (Provoke an error that is neither very gracefully nor consistently handled.)

Search: err

16 challenges are unavailable on Docker due to security concerns or technical incompatibility!

Security Misconfiguration
Error Handling
Provoke an error that is neither very gracefully nor consistently handled.

Explanation

When errors are not properly handled, it make leak information or configuration about the server, which is a potential weakness for attackers to exploit.

To prevent this vulnerability, we have to configure the server to properly handle errors, and avoid displaying any debug information in a production environment.

References

- Security Misconfiguration :: Pwning OWASP Juice Shop

(e) **Login Admin**

Steps

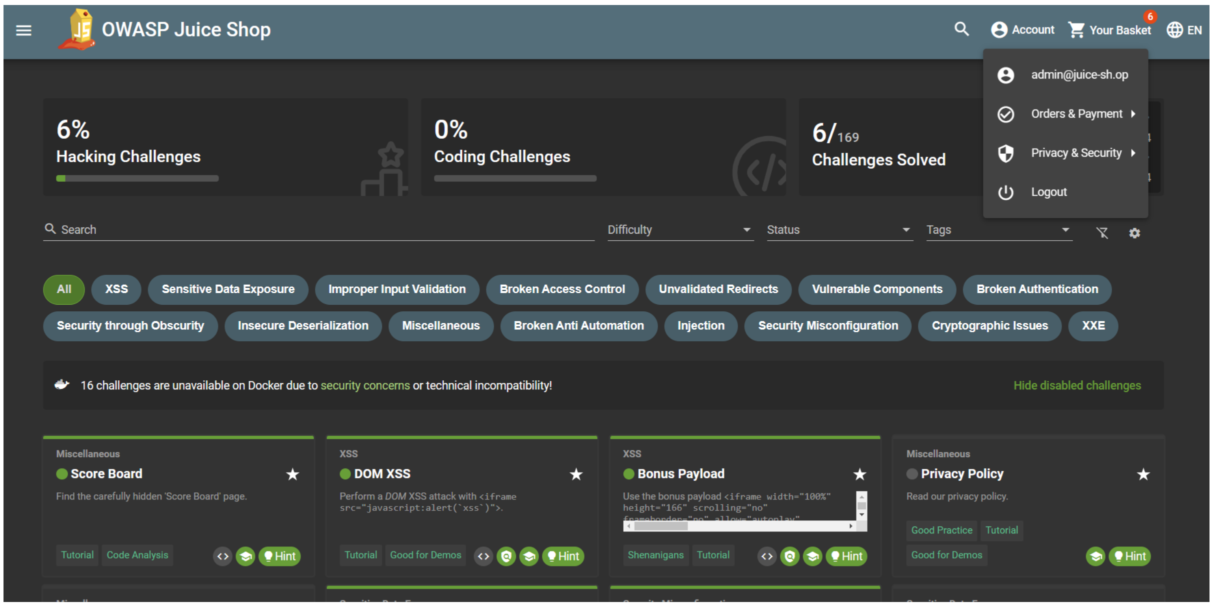
(i) In the “Login” (`/#/login`) page, type `'` as Email and a random string for the password.

(ii) In Chrome DevTools, inspect the failed `/rest/usr/login` request. The SQL command is in the response. This allows us to perform SQL injection.

```
{
  "error": {
    "message": "SQLITE_ERROR: unrecognized token:
    ↪  \"962012d09b8170d912f0669f6d7d9d07\"",
    // ...
    "sql": "SELECT * FROM Users WHERE email = ''' AND password =
    ↪  '962012d09b8170d912f0669f6d7d9d07' AND deletedAt IS NULL",
    "parameters": {}
  }
}
```

(iii) Back to the login page, type `' UNION SELECT * FROM Users --` as the email. We have successfully logged into the admin account.

Result



16 challenges are unavailable on Docker due to security concerns or technical incompatibility!

Find the carefully hidden 'Score Board' page.

Perform a *DOM* XSS attack with `<iframe src="javascript:alert(`xss`)">`.

Use the bonus payload `<iframe width="100%" height="166" scrolling="no" frameborder="no" allow="autoplay"`

Read our privacy policy.

Explanation

This attack is possible because the SQL command is included in the response, and the HTML form allows us to input special characters used in SQL commands as the email.

To prevent this type of attack, we never reveal SQL commands to the public, and enforce syntax checking for the email and password fields.

References

- SQL injection UNION attacks | Web Security Academy

(2) CSRF

Cross-site request forgery (CSRF) refers to an attack where unauthorized commands are sent to a website as if they are from a trusted user of the website. This is usually done by the attacker tricking an innocent victim into submitting a maliciously crafted web request to a website that the user has access to. Because the victim's browser has the cookie or session keys, the forged request would be sent as the victim's identity, therefore exposing weaknesses.
References

- Cross-site request forgery - Wikipedia
- CSRF - MDN Web Docs Glossary: Definitions of Web-related terms | MDN