

# Write-up Qualification

## Techtonic Expo CTF



Presented By:  
**LastSeenIn2026**

Sugeng Dwi Hermanto (SMKN 1 Cibinong)  
Deffreus Theda (SMA Pradita Dirgantara)  
Riduan (SMKN 2 Pangkalpinang)

# [ DAFTAR ISI ]

|  |    |
|--|----|
| [ DAFTAR ISI ].....  | 2  |
| [ WEB EXPLOITATION ].....                                    | 3  |
| 1. Beguyur.....  | 3  |
| Overview:.....   | 3  |
| Code Review:.....  | 4  |
| “TechtonicExpoCTF{d3k_k4w4_nyu54h}”.....                     | 6  |
| Flag: TechtonicExpoCTF{d3k_k4w4_nyu54h}.....                 | 6  |
| 2. Baloy.....  | 7  |
| Solution:.....   | 7  |
| Flag: TechtonicExpoCTF{g4u54h_s0k_k3r3n_l0_buk3n_b4l0y}..... | 9  |
| 3. First Injec.....  | 9  |
| Overview:.....   | 9  |
| Solution:.....   | 10 |
| Flag: TechtonicExpoCTF{f1r5t_1nj3ct_fr0m_d0ct0r}.....        | 10 |
| [ FORENSIC ].....  | 11 |
| 1. The Goat.....   | 11 |
| Solution:.....   | 11 |
| Flag: TechtonicExpoCTF{m3551_b3tt3r_th4n_r0n4ld0}.....       | 12 |
| 2. Unknown Message.....                                      | 13 |
| Overview:.....   | 13 |
| Solution:.....   | 14 |
| Flag: TechtonicExpoCTF{ma4f_b4n9_m451h_p3mul4}.....          | 14 |
| [ CRYPTOGRAPHY ].....  | 15 |
| 1. Luxardo Fernet.....                                       | 15 |
| Overview:.....   | 15 |
| Solution:.....   | 15 |
| Flag: TechtonicExpoCTF{drunk_1s_b3tt3r_f0r_h3al7h}.....      | 16 |
| 2. Asak Kawa Ge Pacak.....                                   | 16 |
| Overview:.....   | 17 |
| Solution:.....   | 17 |
| Flag: TechtonicExpoCTF{4s4k_k4w4_g3_p4c4k}.....              | 17 |
| [ REVERSE ENGINEERING ].....                                 | 18 |
| 1. Kode Nuklir.....  | 18 |
| Overview:.....   | 18 |
| Solusi:.....   | 18 |
| Flag: TechtonicExpoCTF{BOKUNOCHINCHINWACHIISAI}.....         | 19 |
| [ Pwn ].....   | 20 |
| 1. Kue Lapis.....  | 20 |
| Overview:.....   | 21 |
| Solusi:.....   | 21 |
| Flag: TechtonicExpoCTF{sp34k3r_ku_ru54k_b4n9}.....           | 23 |
| [ MISC ].....  | 24 |
| 1. Techtonic Explo.....                                      | 24 |
| Flag: TechtonicExpoCTF{w3_5t4r7_fr0m_h3r3}.....              | 24 |

# [ WEB EXPLOITATION ]

## 1. Beguyur



### Overview:

Pada tantangan kali ini, saya diberikan sebuah link url yang menuju sebuah website, langsung saja saya klik dan melihatnya, berikut tampilan website tersebut:

### Solution:

A screenshot of a simple website. The address bar shows the URL "saturn.techtonicexpo.com". The main content area has a black header with the text "Cuma sekedar HTML yang membosankan". Below the header is a dark grey navigation bar with the words "Gimana" and "Apa". The main body is light grey and contains the text "Gimana? Gimana, bagus ga web gua?".

Disini saya disuguhkan website sederhana dan memiliki 2 opsi page di web tersebut yaitu tampilan page “Gimana & Apa”.

Cuma sekedar HTML yang membosankan

Gimana      Apa

Apa

Web ini terdiri dari:

- HTML
- CSS
- JS (JavaScript)

Berdasarkan hint diatas, pada page “Apa” saya langsung saja melihat isi source code webnya, saya merasa pada tantangan kali ini flagnya terpisah-pisah pada source-code file yang berbeda.

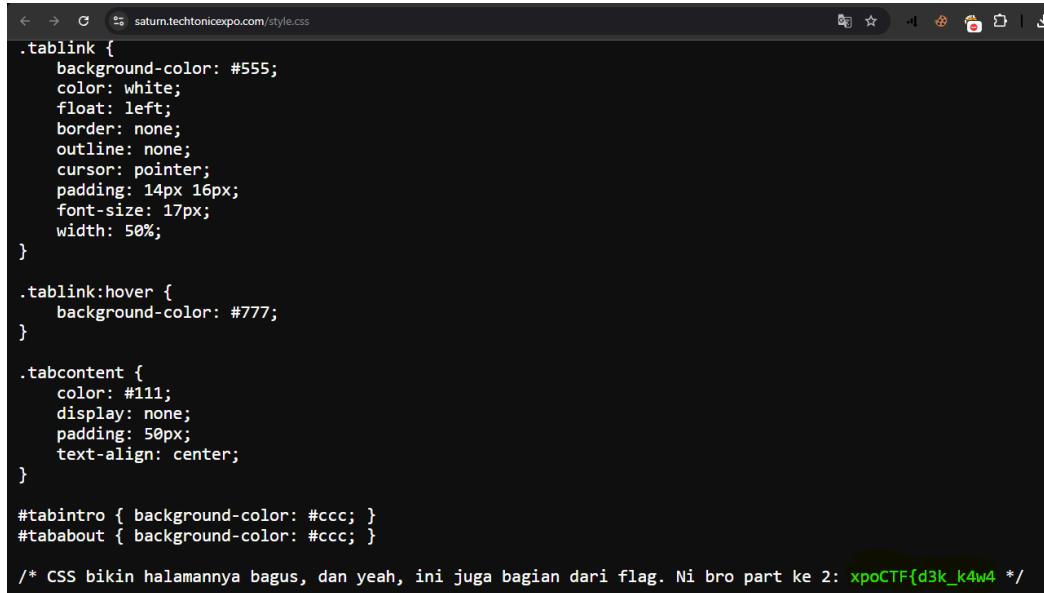
## Code Review:

### 1. Potongan Flag Pertama - Source Page

```
<!DOCTYPE html>
<html>
  <head>
    <title>Beguyur</title>
    <link href="https://fonts.googleapis.com/css?family=Open+Sans|Roboto" rel="stylesheet">
    <link rel="stylesheet" type="text/css" href="style.css">
    <script type="application/javascript" src="script.js"></script>
  </head>
  <body>
    <div class="container">
      <header>
        <h1>Cuma sekedar HTML yang membosankan</h1>
      </header>
      <button class="tablink" onclick="openTab('tabintro', this, '#222')" id="defaultOpen">Gimana</button>
      <button class="tablink" onclick="openTab('tababout', this, '#222')>Apa</button>
      <div id="tabintro" class="tabcontent">
        <h3>Gimana?</h3>
        <p>Gimana, bagus ga web gua?</p>
      </div>
      <div id="tababout" class="tabcontent">
        <h3>Apa</h3>
        <p>Web ini terdiri dari: <br/>
          HTML <br/>
          CSS <br/>
          JS (JavaScript)</p>
        <!-- Ini Part pertama: TechtonicE -->
      </div>
    </div>
  </body>
</html>
```

ini merupakan potongan flag pertamanya “TechtonicE”, ketika sudah mendapatkan potongan flag yang pertama, langsung saja saya mencari potongan flag kedua dan ketiga, berdasarkan hint sebelumnya, langsung saja saya cek file “CSS dan Javascript” tersebut.

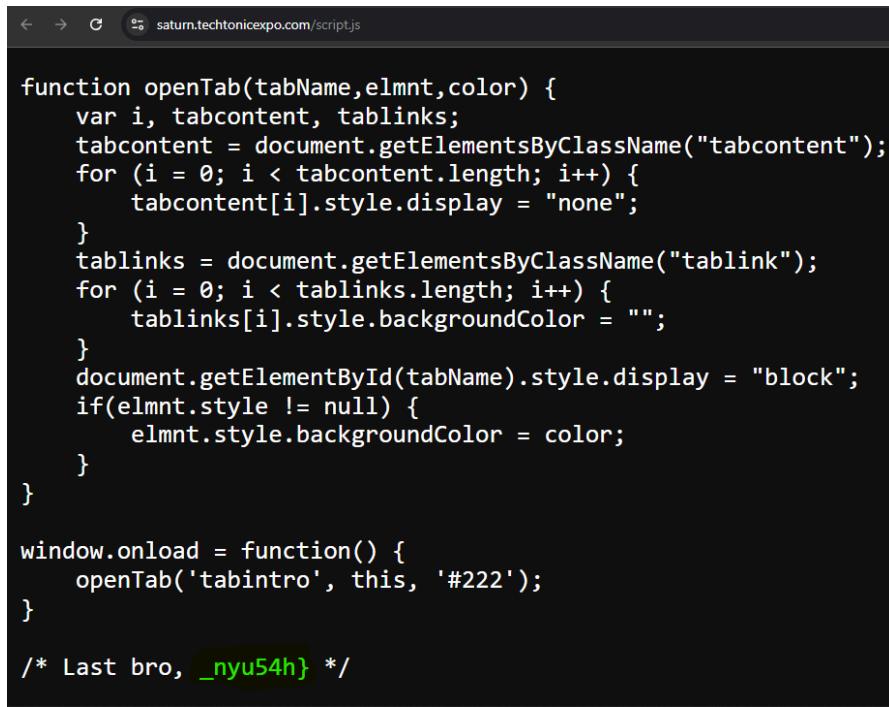
## 2. Potongan Flag Kedua - File CSS



```
.tablink {  
    background-color: #555;  
    color: white;  
    float: left;  
    border: none;  
    outline: none;  
    cursor: pointer;  
    padding: 14px 16px;  
    font-size: 17px;  
    width: 50%;  
}  
  
.tablink:hover {  
    background-color: #777;  
}  
  
.tabcontent {  
    color: #111;  
    display: none;  
    padding: 50px;  
    text-align: center;  
}  
  
#tabintro { background-color: #ccc; }  
#tababout { background-color: #ccc; }  
  
/* CSS bikin halamannya bagus, dan yeah, ini juga bagian dari flag. Ni bro part ke 2: xpoCTF{d3k_k4w4 */
```

berikut potongan flag keduanya “**xpoCTF{d3k\_k4w4}**”

## 3. Potongan Flag Ketiga - File JS (JavaScript)



```
function openTab(tabName,elmnt,color) {  
    var i, tabcontent, tablinks;  
    tabcontent = document.getElementsByClassName("tabcontent");  
    for (i = 0; i < tabcontent.length; i++) {  
        tabcontent[i].style.display = "none";  
    }  
    tablinks = document.getElementsByClassName("tablink");  
    for (i = 0; i < tablinks.length; i++) {  
        tablinks[i].style.backgroundColor = "";  
    }  
    document.getElementById(tabName).style.display = "block";  
    if(elmnt.style != null) {  
        elmnt.style.backgroundColor = color;  
    }  
}  
  
window.onload = function() {  
    openTab('tabintro', this, '#222');  
}  
  
/* Last bro, _nyu54h */
```

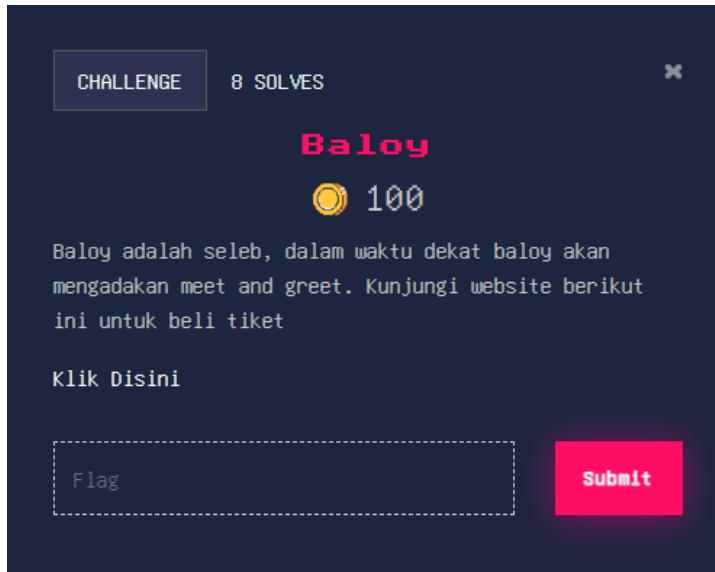
berikut potongan flag ketiganya “**\_nyu54h}**” akhirnya saya mendapatkan full potongan flagnya, berikut gabungan potongan flag yang terpisah tadi.

“**TechtonicExpoCTF{d3k\_k4w4\_nyu54h}**”

Imang-imang probset e ne ok, bulek name flag e “dek kawa nyusa” tapi bikin soal e nyusah urang lain kwkkw, tapi makasih la ok bang. Walapun nk beguyur ngegawi e.

**Flag:** **TechtonicExpoCTF{d3k\_k4w4\_nyu54h}**

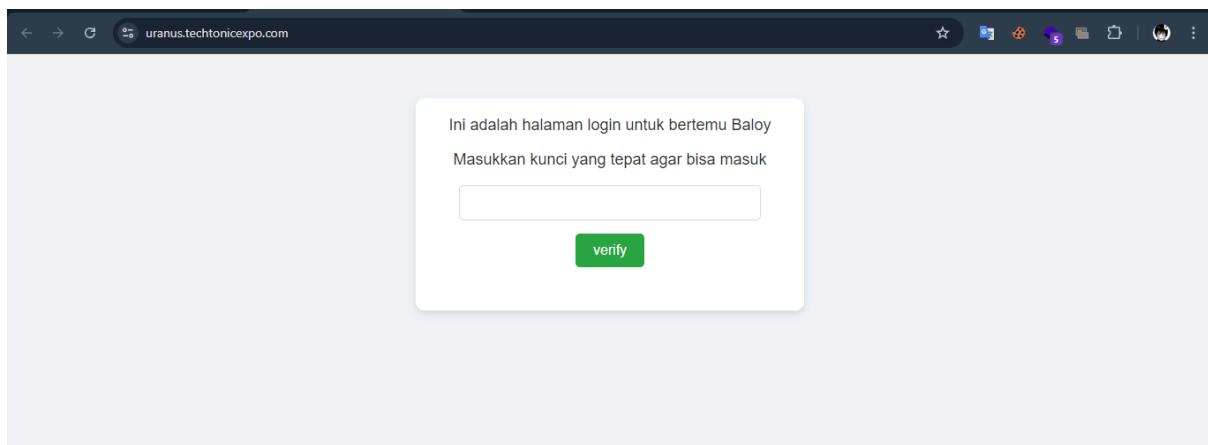
## 2. Baloy



### Overview:

Intinya saya disuruh menemukan **kunci** yang tepat agar bisa **masuk/login**.

### Solution:



Ini adalah challenge black box, karena tidak diberikan source code nya, tetapi itu bisa saya dapatkan source **code js** nya dengan **view page source**.

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Baloy Sok Keren</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>

    <script type="text/javascript" src="script.js"></script> <!-- Sini bro kalo mau ketemu baloy -->

    <div class="login-container">
        <p>Ini adalah halaman login untuk bertemu Baloy</p>
        <br>
        <p>Masukkan kunci yang tepat agar bisa masuk</p>
        <br>
        <form action="index.html" method="post">
            <input type="password" id="pass" size="8" />
            <br/>
            <input type="submit" value="verify" onclick="verify(); return false;" />
        </form>
    </div>
</body>
</html>

```

Disini diberikan sebuah hint yang sangat sus dan mengarah ke **script.js**.

```

function verify() {
    var checkpass = document.getElementById("pass").value;
    var split = 4;

    if (checkpass.substring(split*4, split*5) == '{g4u') {
        if (checkpass.substring(0, split) == 'Tech') {
            if (checkpass.substring(split*8, split*9) == 'n_10') {
                if (checkpass.substring(split*1, split*2) == 'toni') {
                    if (checkpass.substring(split*3, split*4) == 'oCTF') {
                        if (checkpass.substring(split*9, split*10) == '_buk') {
                            if (checkpass.substring(split*7, split*8) == 'k3r3') {
                                if (checkpass.substring(split*11, split*12) == '4l0y') {
                                    if (checkpass.substring(split*6, split*7) == 's0k_') {
                                        if (checkpass.substring(split*2, split*3) == 'cExp') {
                                            if (checkpass.substring(split*5, split*6) == '54h_') {
                                                if (checkpass.substring(split*10, split*11) == '3n_b') {
                                                    if (checkpass.substring(split*12, split*13) == '}') {
                                                        alert("Password Verified");
                                                    }
                                                }
                                            }
                                        }
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
    } else {
        alert("Incorrect password");
    }
}

```

Dan ternyata key nya adalah flagnya yang telah di acak.

Disini saya menyusun sesuai dengan urutan **split** nya mulai dari 0 - 13.

Berikut hasilnya setelah di urutin split 1-13:

**TechtonicExpoCTF{g4u54h\_s0k\_k3r3n\_l0\_buk3n\_b4l0y}**

**Flag: TechtonicExpoCTF{g4u54h\_s0k\_k3r3n\_l0\_buk3n\_b4l0y}**

### 3. First Inject



#### Overview:

Pada tantangan ini saya diberikan sebuah url yang menuju sebuah website, ternyata di website tersebut saya disuruh mem bypass form tersebut agar bisa mendapatkan flagnya.

## Tampilan Website Awal:

The screenshot shows a web browser window with the URL `neptunus.techtoniceexpo.com`. The page title is "Techtonic Expo". Below it is a text block: "Coba cari data rahasia dengan memanfaatkan input di bawah ini. Bisakah Anda Bypass Security nya?". A search form is present with the placeholder "Masukkan Nama Karyawan" and a blue "Cari" button.

Berdasarkan judul soal tadi “**First Injec**”, saya berasumsi jika kamu bisa membypass form web ini, saya langsung mendapatkan flagnya.

## Solution:

Saya menggunakan payload sederhana SQL Inject yaitu “' OR '1'='1” yang dimana bagian query ini **OR '1'='1** selalu **true**, sehingga query mengabaikan kondisi password dan menampilkan semua data atau memberikan akses tanpa otentikasi yang sah.

The screenshot shows the same website after a SQL injection. The search input now contains "' OR '1'='1". The results section displays a list of employees with their details. The last entry in the list is highlighted with a green box and contains the flag: "TechtonicExpoCTF{f1r5t\_1nj3ct\_fr0m\_d0ct0r}".

| ID | Nama    | Jabatan   | Gaji |
|----|---------|-----------|------|
| 1  | Alice   | Manager   | 7000 |
| 2  | Bob     | Developer | 5000 |
| 3  | flag    | flag      | flag |
| 4  | Charlie | HR        | 4000 |

**Flag: TechtonicExpoCTF{f1r5t\_1nj3ct\_fr0m\_d0ct0r}**

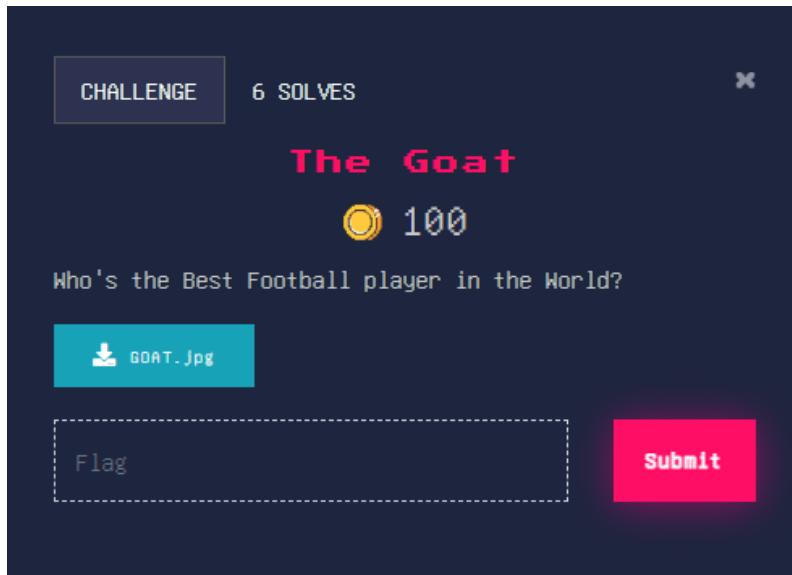
Berikut refrensi mengenai SQL Injection:

[https://cheatsheetseries.owasp.org/cheatsheets/SQL\\_Injection\\_Prevention\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html)

<https://stackoverflow.com/questions/60939830/sql-injection-or-1-1-vs-or-1-1>

# [ FORENSIC ]

## 1. The Goat



### Overview:

Challenge ini berisi file gambar dengan nama "**GOAT.jpg**" yang tampaknya memiliki data tersembunyi di dalamnya. Dengan menggunakan tools forensik seperti binwalk, saya dapat menemukan flag tersembunyi dalam gambar tersebut. Challenge ini melibatkan teknik **steganografi**.

### Solution:

```
(PwnH4x0r㉿kali)-[~/CTF/techtonic/foren]
$ file GOAT.jpg
GOAT.jpg: JPEG image data, JFIF standard 1.01, aspect ratio, density 1x1, segment length 16, progressive, precision 8, 338x601, components 3
```

Ini adalah file jpeg.

```
(PwnH4x0r㉿kali)-[~/CTF/techtonic/foren]
$ binwalk GOAT.jpg
DECIMAL      HEXADECIMAL      DESCRIPTION
0            0x0                JPEG image data, JFIF standard 1.01
33617        0x8351           Zip archive data, at least v2.0 to extract, compressed size: 341309, uncompressed size: 358345, name: GOAT.jpg
375070       0x5891E          End of Zip archive, footer length: 22

(PwnH4x0r㉿kali)-[~/CTF/techtonic/foren]
$ binwalk -e GOAT.jpg
DECIMAL      HEXADECIMAL      DESCRIPTION
0            0x0                JPEG image data, JFIF standard 1.01
33617        0x8351           Zip archive data, at least v2.0 to extract, compressed size: 341309, uncompressed size: 358345, name: GOAT.jpg
375070       0x5891E          End of Zip archive, footer length: 22
```

Setelah saya binwalk terdapat file **jpeg & zip** di dalamnya, lalu saya extract.

```
[PwnH4x0r㉿kali)-[~/CTF/techtonic/foren/_GOAT.jpg.extracted]
$ ll
total 688
-rw-rw-r-- 1 PwnH4x0r PwnH4x0r 341475 Oct 11 23:02 8351.zip
-rwxrwxrwx 1 PwnH4x0r PwnH4x0r 358345 Sep 22 06:50 GOAT.jpg
```

ini adalah isinya, lalu buka file **GOAT.jpg** untuk melihat apa isinya.



TechtonicExpoCTF{m3551\_b3tt3r\_th4n\_r0n4ld0}

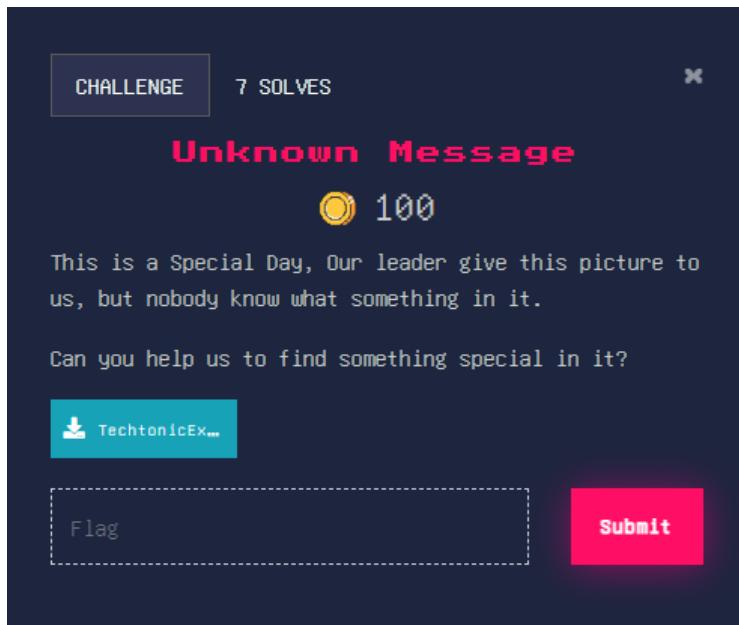
Setelah di buka terdapat sebuah flag dengan tulisan yang kecil berada di ujung bawah bagian kiri, karena saya megar untuk menulis secara manual, saya menggunakan tools **tesseract**.

```
[PwnH4x0r㉿kali)-[~/CTF/techtonic/foren/_GOAT.jpg.extracted]
$ tesseract GOAT.jpg out
Estimating resolution as 296
format = Fuzznet(key)
```

```
[PwnH4x0r㉿kali)-[~/CTF/techtonic/foren/_GOAT.jpg.extracted]
$ cat out.txt
\ r
|
J
;
mn
TechtonicExpoCTF{m3551_b3tt3r_th4n_r0n4ld0}
```

**Flag: TechtonicExpoCTF{m3551\_b3tt3r\_th4n\_r0n4ld0}**

## 2. Unknown Message



### Overview:

Challenge ini melibatkan sebuah file gambar **TechtonicExpo.jpg**.

Menggunakan **ExifTool**, kita dapat menemukan metadata gambar yang berisi informasi tersembunyi yang menjadi flag.

## Solution:

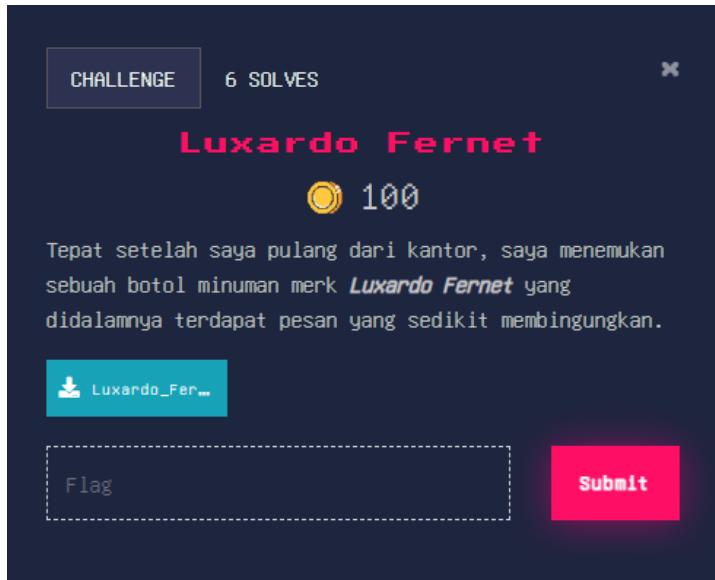
```
[PwnH4x0r㉿kali]:[~/CTF/techtonic/foren/Unknown Message]
$ exiftool TechtonicExpo.jpg
ExifTool Version Number : 12.76
File Name               : TechtonicExpo.jpg
Directory              : .
File Size               : 179 kB
File Modification Date/Time : 2024:09:27 14:07:06-04:00
File Access Date/Time   : 2024:10:11 23:21:36-04:00
File Inode Change Date/Time : 2024:10:11 23:21:10-04:00
File Permissions        : -rw-rw-r--t = Fernet(key)
File Type               : JPEG
File Type Extension    : jpg
MIME Type               : image/jpeg
JFIF Version            : 1.01
Resolution Unit          : None
X Resolution             : 1
Y Resolution             : 1
Exif Byte Order          : Big-endian (Motorola, MM)
Make                     : hPpEY0I6mteQqEV
Camera Model Name        : nLjMIb5Nhj
Software                 : p5kBxro1KqrP7KNCPcxi7yVSk
Artist                   : TechtonicExpoCTF{ma4f_b4n9_m451h_p3mul4}
Image Width              : 1675
Image Height             : 2368
Encoding Process         : Baseline DCT, Huffman coding
Bits Per Sample          : 8
Color Components          : 3
Y Cb Cr Sub Sampling    : YCbCr4:2:0 (2 2)
Image Size                : 1675×2368
Megapixels                  : 4.0
```

Disini saya menggunakan tools **exiftool**, lalu terdapat sebuah flag di bagian **Artist**.

**Flag: TechtonicExpoCTF{ma4f\_b4n9\_m451h\_p3mul4}**

# [ CRYPTOGRAPHY ]

## 1. Luxardo Fernet



### Overview:

Dalam challenge ini, kita diberikan sebuah file berisi pesan terenkripsi menggunakan skema **Fernet**, yang merupakan salah satu implementasi dari **Advanced Encryption Standard (AES)** dengan mode Cipher Block Chaining (CBC). Saya juga diberikan sebuah kunci (key) yang digunakan untuk mendekripsi pesan tersebut.

### Solution:

Langkah pertama adalah mengidentifikasi bahwa enkripsi menggunakan skema **Fernet**, yang berarti kunci dan ciphertext sudah dalam format yang sesuai untuk proses dekripsi. Kunci yang diberikan harus berupa **Base64-encoded key**, dan ciphertext harus dalam bentuk yang sama.

#### Berikut script **solve.py**:

```
from cryptography.fernet import Fernet

enc =
b'gAAAAABm7-g2B7hL2JKdzsMINdTp...sCvXFvQIzxGifLEoRG3lsGVNzoEb55W2lEtI8F'
```

```
bTZ7d6F-H1F6EfHAH5BtRh60JU8wyRFWrQAGjcQDU9KrCr4a_f2L1LG1MP4zGI2yoWeoxZf
'

key = b'itMoAAnncdjFY2P93rTAq8lcr2dNvfht6_G-43fBd1I='

fernet = Fernet(key)
decrypted_message = fernet.decrypt(enc)

print(decrypted_message.decode())

```

**Fernet(key):** Membuat objek **Fernet** menggunakan kunci yang diberikan.

**fernet.decrypt(enc):** Menggunakan kunci untuk mendekripsi pesan terenkripsi (**enc**).

**decrypted\_message.decode():** Mendekode hasil decrypt dari byte string menjadi string yang dapat dibaca manusia.

```
[└─(PwnH4x0r㉿kali)-[~/CTF/techtonic/crypto]
$ python3 solve.py
TechtonicExpoCTF{drunk_1s_b3tt3r_f0r_h3al7h}
```

## Solusi ke 2:

Cara kedua solve chall ini adalah dengan menggunakan web online fernet decode.

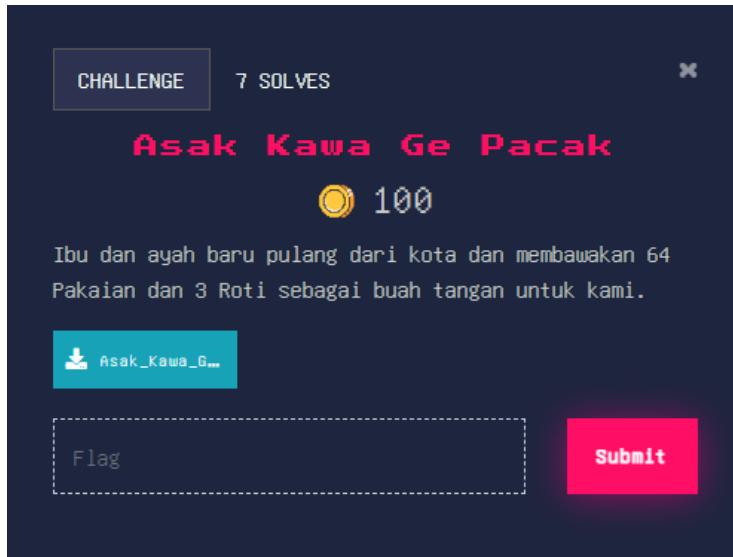
Berikut link nya: <https://asecuritysite.com/tokens/ferdecode>

The screenshot shows the Fernet (Decode) tool interface. The token input field contains: gAAAAABm7-gB7hL2JkdzMINdTp... and the key input field contains: itMoAAnn... . The decoded output is: TechtonicExpoCTF{drunk\_1s\_b3tt3r\_f0r\_h3al7h}.

Tinggal masukin enc dan key nya, dan boom dapet flag.

**Flag: TechtonicExpoCTF{drunk\_1s\_b3tt3r\_f0r\_h3al7h}**

## 2. Asak Kawa Ge Pacak



### Overview:

Challenge ini memberikan sebuah ciphertext yang tampaknya telah dienkripsi menggunakan beberapa lapisan encoding. Dari proses yang dilakukan, dapat dilihat bahwa langkah pertama adalah decoding **Base64** dan kemudian decrypting menggunakan **ROT13**. Setelah melalui kedua langkah tersebut, pesan akhirnya dapat dipecahkan menjadi flag untuk CTF.

### Solution:

```
[~] (PwnH4x0r㉿kali)-[~/CTF/techtonic/crypto]
└─$ cat Asak_Kawa_Ge_Pacak
YidSM0p3ZFdkaVlYWndVbXRqWWxCsfUzczBaali0WDNnMGFqUmZkRE5mWXpSd05IaDk9Jw==

[~] (PwnH4x0r㉿kali)-[~/CTF/techtonic/crypto]
└─$ echo "YidSM0p3ZFdkaVlYWndVbXRqWWxCsfUzczBaali0WDNnMGFqUmZkRE5mWXpSd05IaDk9Jw==" | base64 -d
b'R3JwdWdiYXZwUmtjYlBHU3s0ZjR4X3g0ajRfdDNfYzRwNHh9='

[~] (PwnH4x0r㉿kali)-[~/CTF/techtonic/crypto]
└─$ echo "R3JwdWdiYXZwUmtjYlBHU3s0ZjR4X3g0ajRfdDNfYzRwNHh9=" | base64 -d
GrpugbavpRkcbPGS{4f4x_x4j4_t3_c4p4x}base64: invalid input

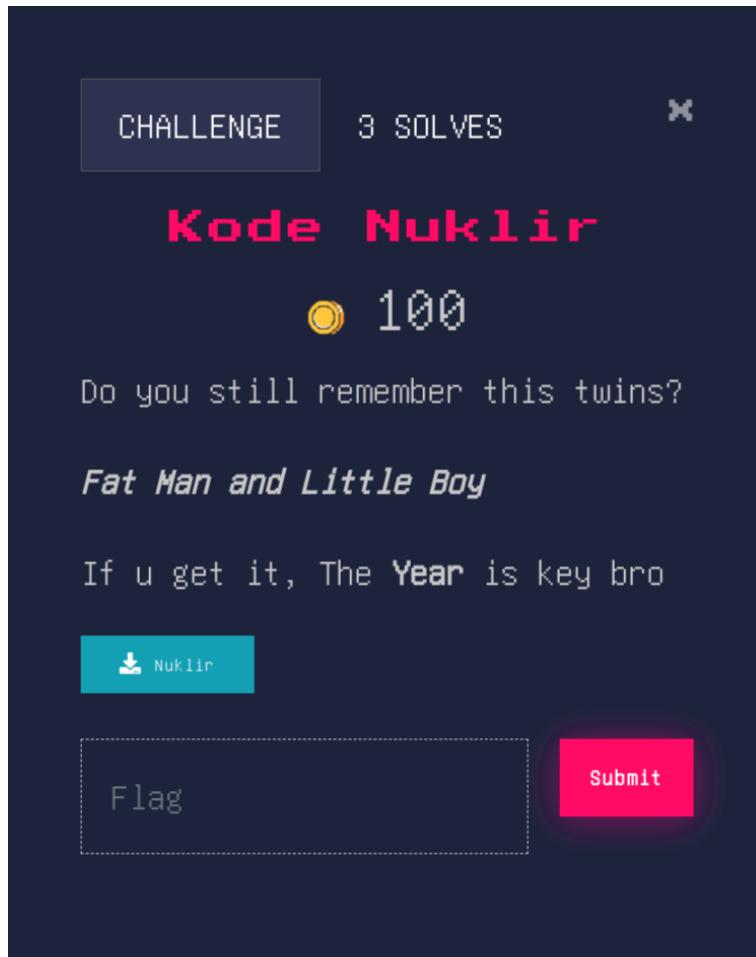
[~] (PwnH4x0r㉿kali)-[~/CTF/techtonic/crypto]
└─$ echo "GrpugbavpRkcbPGS{4f4x_x4j4_t3_c4p4x}" | tr 'A-Za-z' 'N-ZA-Mn-za-m'
TechtonicExpoCTF{4s4k_k4w4_g3_p4c4k}
```

**Flag: TechtonicExpoCTF{4s4k\_k4w4\_g3\_p4c4k}**

# [ REVERSE ENGINEERING ]

By Effie, best girl :33

## 1. Kode Nuklir



### Overview:

Challenge ini memberikan sebuah file Java class data yang perlu didekompilasi. Setelah ditelusuri, program ini memiliki data dan code untuk mencetak flag jika kode nuklir yang diberikan benar (yang dapat diakali).

### Solusi:

Kita mendapatkan sebuah file **Nuklir.class**, yang merupakan sebuah compiled Java class data, version **66.0**. Tidak seperti source code biasa, jika kita buka file tersebut, yang muncul hanyalah **gibberish**, karena .class itu

machine code, untuk dibaca komputer dan bukan dibaca manusia. Untuk mendapatkan [source codenya](#) kembali, kita perlu untuk mendekompilasikannya, salah satu caranya yaitu menggunakan [Jadx](#):

```
$ jadx Nuklir.class
INFO - loading ...
INFO - processing ...
INFO - done
$ tree Nuklir
[0] Nuklir
└── [1] sources
    └── [2] defpackage
        └── [3] Nuklir.java
```

```
public static void main(String[] strArr) {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Masukkan kode nuklir: ");
    if (!isValidKodeNuklir(scanner.nextLine(), fetchKodeNuklirFromServer())) {
        System.out.println("Maaf, Kode nuklirnya salah gan");
    } else {
        StringBuilder sb = new StringBuilder("TechtonicExpoCTF{");
        for (String str : new String[]{"2", "15", "11", "21", "14", "15", "3", "8", "9", "14", "3",
"8", "9", "14", "23", "1", "3", "8", "9", "19", "1", "9"}) {
            sb.append((char) ((Integer.parseInt(str) - 1) + 65));
        }
        sb.append("}");
        System.out.println(sb.toString());
    }
    scanner.close();
}
```

Kita diharapkan untuk melewati *if* tersebut, namun data dan kode untuk print flag sudah ada dalam blok kode *else* :

Kita dapat mengekstrak code di dalam *else* pada sebuah file baru ([\*Solve.java\*](#)) dan mencetak flagnya XD

```
public class Solve {
    public static void main(String... args) {
        StringBuilder sb = new StringBuilder("TechtonicExpoCTF{");
        for (String str : new String[]{"2", "15", "11", "21", "14", "15", "3", "8", "9", "14", "3",
"8", "9", "14", "23", "1", "3", "8", "9", "19", "1", "9"}) {
            sb.append((char) ((Integer.parseInt(str) - 1) + 65));
        }
        sb.append("}");
        System.out.println(sb.toString());
    }
}
```

Sudah, mari kita kompilasi dan jalankan:

```
$ javac Solve.java && java Solve
TechtonicExpoCTF{BOKUNOCHINCHINWACHIISAI}
```

**Flag: TechtonicExpoCTF{BOKUNOCHINCHINWACHIISAI}**

# [ Pwn ]

## 1. Kue Lapis

CHALLENGE    2 SOLVES    X

### Kue Lapis

100

Temukan cara untuk memodifikasi variabel aman kami dan ambil flag yang tersembunyi di file ini. Gunakan semua trik yang kamu miliki untuk mencapainya! Apakah kamu cukup cerdik?

[View Hint](#)

[View Hint](#)

[View Hint](#)

[View Hint](#)

[kueLapis...](#)

Flag

Submit

## Overview:

Challenge ini memberikan sebuah source code program C yang memiliki **vulnerability buffer overflow**. Dengan mengeksploitasi kerentanan tersebut, program akan memberikan kita flag yang berada di server.

## Solusi:

Kita dapat sebuah source code file bernama [kueLapis.c](#). Kita cari-cari function terpenting, dan mendapatkan *check\_win* yang harus kita pass:

```
void check_win() {
    // menjadi True kalau kedua string berbeda
    if (strcmp(safe_var, "@techtionic") != 0) {
        printf("\nSelamat! Kamu telah mengubah safe_var!\n");

        // Mengambil flag dari server
        get_flag_from_server();

        exit(0);
    } else {
        printf("Lebih semangat lagi! hehe\n");
        printf("\nBelum ada flag buat kamu :(\\n");
    }
}
```

*strcmp* berfungsi untuk membandingkan variable *safe\_var* dengan string **@techtionic**, jika beda maka return value akan menjadi negatif atau positif tergantung urutan leksikografis string pertama. Disini, code di dalam *if* akan terjalankan, dan mengambil **flag** dari server (detail *get\_flag\_from\_server* tidak penting). Dengan itu, maka tujuan utama kita adalah **mengganti (write) variable *safe\_var***.

Jika ditelusuri, vulnerability pada program ini terletak pada function ***write\_buffer***:

```
#define INPUT_DATA_SIZE 11

char *input_data;

void init() {
    printf("\nSelamat datang di TechTonic!\n");
    printf("Aku meletakkan data ku di heap, jadi seharusnya aman dari exploit :D.\n\\n");

    input_data = malloc(INPUT_DATA_SIZE);
    if (!input_data) {
        perror("Failed to allocate memory for input_data");
        exit(1);
    }
    strncpy(input_data, "TechTonic", INPUT_DATA_SIZE);

    safe_var = malloc(SAFE_VAR_SIZE);
    /* init safe_var */
}

void write_buffer() {
```

```
    printf("Data for buffer: ");
    fflush(stdout);
    scanf("%s", input_data);
}
```

## Mengapa?

`scanf` mengambil input dari user sebanyak-banyaknya (sampai ketemu newline, ), sedangkan variabel `input_data` memiliki size 11 yang belum tentu dapat menampung semua karakter (byte) input user. Bytes tambahan tersebut ter-write ke memory address berikutnya, yang ternyata `save_var` berada tepat di depannya (lihat `malloc` di kode atas). Tipe vulnerability ini disebut sebagai **Buffer Overflow**, memungkinkan kita untuk mengganti (write) ke variable `safe_var` dengan input yang melebihi size buffer `input_data`.

Program ini memberikan pilihan ketika kita menjalankannya: cetak heap, tulis ke buffer (`input_data`), cetak `save_var`, cetak flag (`check_win`), dan keluar. Hal yang perlu kita lakukan hanyalah mengeksplorasi buffer overflow untuk mengganti `safe_var` dan mencetak flag (memanggil `check_win`)!

Maka dari itu, kita solve di terminal seperti ini:

```
$ gcc kueLapis.c && ./a.out

Selamat datang di TechTonic!
Aku meletakkan data ku di heap, jadi seharusnya aman dari exploit :D.

Heap State:
+-----+-----+
[*] Address -> Heap Data
+-----+-----+
[*] 0x2446b0 -> TechTonic
+-----+-----+
[*] 0x2446d0 -> @techtonic
+-----+-----+

1. Cetak Heap:      (cetak keadaan heap saat ini)
2. Tulis ke buffer: (tulis ke blok data pribadi kamu di heap)
3. Cetak safe_var: (Lihat variabel safe_var di heap)
4. Cetak Flag:     (Coba cetak flag, semoga berhasil)
5. Keluar

Masukkan pilihan kamu: 3

Lihat variabel safe_var: @techtonic

1. Cetak Heap:      (cetak keadaan heap saat ini)
2. Tulis ke buffer: (tulis ke blok data pribadi kamu di heap)
3. Cetak safe_var: (Lihat variabel safe_var di heap)
4. Cetak Flag:     (Coba cetak flag, semoga berhasil)
5. Keluar

Masukkan pilihan kamu: 2
```

```
Data for buffer: 12345678901234567890123456789012  
1. Cetak Heap:      (cetak keadaan heap saat ini)  
2. Tulis ke buffer: (tulis ke blok data pribadi kamu di heap)  
3. Cetak safe_var: (Lihat variabel safe_var di heap)  
4. Cetak Flag:     (Coba cetak flag, semoga berhasil)  
5. Keluar
```

```
Masukkan pilihan kamu: 3
```

```
Lihat variabel safe_var:
```

```
1. Cetak Heap:      (cetak keadaan heap saat ini)  
2. Tulis ke buffer: (tulis ke blok data pribadi kamu di heap)  
3. Cetak safe_var: (Lihat variabel safe_var di heap)  
4. Cetak Flag:     (Coba cetak flag, semoga berhasil)  
5. Keluar
```

```
Masukkan pilihan kamu: 4
```

```
Selamat! Kamu telah mengubah safe_var!  
Flag: TechtonicExpoCTF{sp34k3r_ku_ru54k_b4n9}
```

**Flag: TechtonicExpoCTF{sp34k3r\_ku\_ru54k\_b4n9}**

## [ MISC ]

### 1. Techtonic Exploit



#### Overview:

Berdasarkan deskripsi yang diberikan, disini langsung saja saya menghubungkan ke service server yg diberikan oleh Techtonic dengan menggunakan terminal untuk mendapatkan flagnya.

#### Solution:

```
(wanzkey® Hengker-Bwang)-[~/TechTonic/Web-Exploit]
$ nc 147.79.68.192 24434
TechtonicExpoCTF{w3_5t4r7_fr0m_h3r3}
```

Flag: TechtonicExpoCTF{w3\_5t4r7\_fr0m\_h3r3}