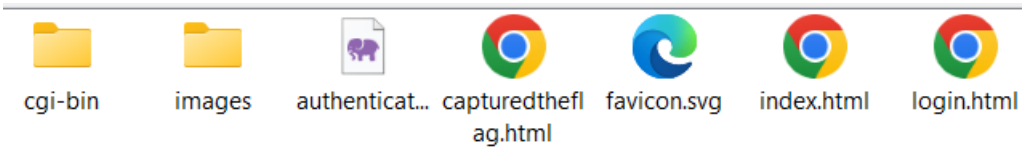
בס"ד

הסבר על הפרוייקט – אבישי גונן

בפרוייקט יש כמה שלבים:

שלב 1: הסנפה באמצעות wireshark ומציאת השם של ה domain. זה די טריוואלי.

שלב 2: נכנסים לאתר. במבנה של האתר שלנו יש 3 קבצי html, דף פתיחה, דף התחברות ודף סיום



כמו כן, דף ה login מתקשר עם קובץ נוסף שרשמתי, authentication.php, שתפקידו יהיה לנהל את הבדיקה של הסיסמא הנכונה, ו redirection לדף סיום במידה והסיסמא באמת נכונה.

הייתי חייב לעשות את זה בצד שרת, אחרת הלקוח היה יכול פשוט לעשות inspect ולראות או מה הסיסמא הנכונה, או פשוט לאן הוא אמור להגיע במידה והוא מכניס את הסיסמא הנכונה.

הנה הקוד:

```
    <script>

        document.getElementById('loginForm').addEventListener('submit',
function(event) {
            event.preventDefault(); // Prevent the form from submitting

            var username = document.getElementById('username').value;
            var password = document.getElementById('password').value;

            // Send POST request to authenticate.php
            fetch('authenticate.php', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({ username: username, password: password
})
            })
            .then(response => {
                return response.json().then(data => {
                    return {status: response.status, body: data};
                });
            })
            .then(data => {
                if (data.status === 200 && data.body.authenticated) {
                    window.location.href = data.body.redirect_url; // Redirect
on successful login
                } else {
                    document.getElementById('errorMessage').innerText =
'Invalid username or password'; // Display error message
```

```
                    document.getElementById('errorMessage').style.display =
'block'; // Show error message
                }
            })
            .catch(error => {
                console.error('Error:', error);
                document.getElementById('errorMessage').innerText = 'Failed to
authenticate. Please try again.'; // Generic error message
                document.getElementById('errorMessage').style.display =
'block'; // Show error message
            });
        });
    </script>
```

מה שקורה פה זה ככה:

כל פעם שהמשתמש מכניס סיסמא, הדפדפן עושה Http POST עם הנתונים, וזה מגיע אל השרת. בצד לקוח זה מחכה לתגובה, ובהתאם לתגובה עושה את הפעולות המתאימות. במידה והוא מקבל שגיאה, הוא מדפיס invalid, אחרת, הוא עושה redirect לדף שאותו שלח ה server.

בצד שרת יש קוד php, שתפקידו לקבל שם משתמש וסיסמא, לבדוק אם הם נכונים, ואם כן לשלוח בחזרה response עם ה redirection page המתאים. במידה והם לא נכונים, הוא ישלח קוד בהתאם.

```php
<?php
// Replace with your actual username and password
$correctUsername = 'EliCopter';
$correctPassword = 'MossadRules';

// Check if the request is a POST request
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Retrieve POST data
    $postData = json_decode(file_get_contents('php://input'), true);

    // Check if username and password are provided
    if (isset($postData['username']) && isset($postData['password'])) {
        $username = $postData['username'];
        $password = $postData['password'];

        // Simple authentication logic
        if ($username === $correctUsername && $password === $correctPassword)
{
            // Authentication successful
            $response = array('authenticated' => true, 'redirect_url' =>
'capturedtheflag.html');
            http_response_code(200);
        } else {
            // Authentication failed
            $response = array('authenticated' => false);
            http_response_code(401);
```

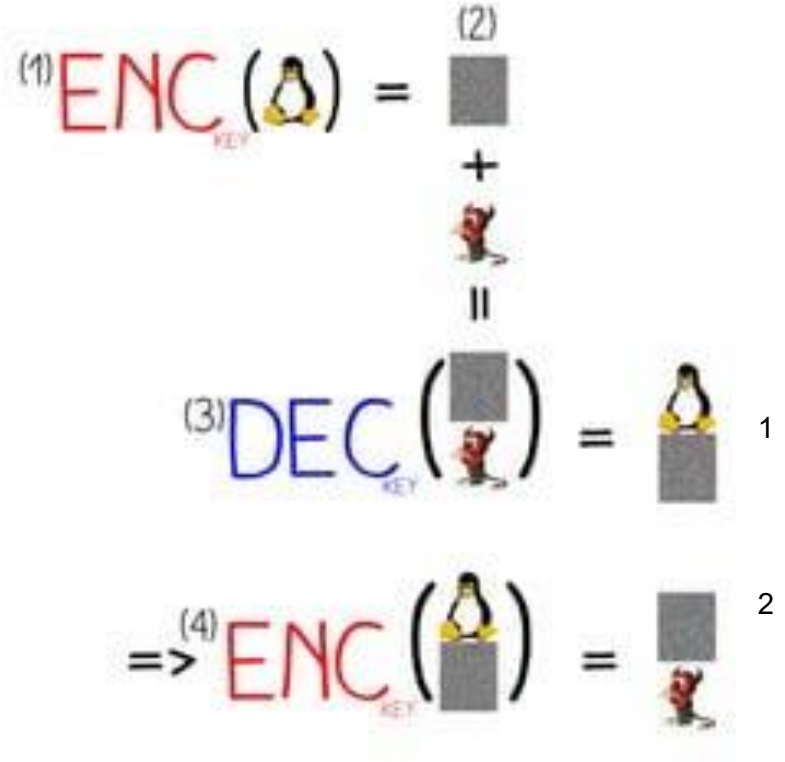
```
        }
    } else {
        // Invalid request
        $response = array('error' => 'Invalid request');
        http_response_code(400);
    }
} else {
    // Method not allowed
    $response = array('error' => 'Method not allowed');
    http_response_code(405);
}

// Return JSON response
header('Content-Type: application/json');
echo json_encode($response);
?>
```

בס"ד

עכשיו נדבר על הקטע המגניב בפרוייקט, הצפנת ה AES.

מה שקורה בעצם זה שאתה מקבל תמונה אחת, מצפין אותה -> ויוצאת תמונה אחרת.. בדר"כ, זה לא אמור לקרות. אז מה בעצם קורה?

זה מבוסס על משהו שלמדנו בקורס הקודם, יסודות באבטחת תוכנה.



זה בעצם הטריק.

בעצם, מה שאני צריך למצוא פה זה IV כזה ככה שב 16 הבתים הראשונים, הוא ייתן לי header מתאים שיגרום לי להתעלם מהאורך של הקובץ השגוי (האורך של tux), ואז הוא יציג לי את התמונה של השטן.

מה שהמשתמש מקבל בהתחלה, זה (1), ובסוף, לאחר ההצפנה מתקבל אצלו (2).

קטע 1:

פה בעצם מקבלים את שני הקבצים, ובודקים שהגודל של הקובץ הראשון הוא קטן מ 0xffff, אחרת זה לא יעבוד.

```
def main():
    KEY = b'Secr3tKeyForAES!'
    key = bytes(KEY)

    input_file_path1 = r"picture1.jpg"
    input_file_path2 = r"picture2.jpg"

    output_file_path = r"final.jpg"
```

```
    with open(input_file_path1, 'rb') as input_file1:
        content1 = input_file1.read()
        header1 = content1[:16]
    with open(input_file_path2, 'rb') as input_file2:
        content2 = input_file2.read()

    output_file = open(output_file_path, 'wb')

    original_size = len(content1)
    if original_size>=0xffff:
        print(f"ERROR, program won't work, first image is too long,
{hex(original_size)} is bigger than 0xffff")
        return
```

זה ה header שאנחנו רוצים שיתקבל בשלב 2, בתחילת הקובץ, אחרי שנעשה encrypt ל tux עם ה IV.

```
 # create the header:
    # FFD8FFFE + size_of_chunk + padding to 16 bytes
    cipher_block1 = b'\xff\xd8\xff\xfe' + original_size.to_bytes(2,
byteorder='big') + b'\x00' * 10

    cipher = Cipher(algorithms.AES(key), modes.CBC(b'\x00'*16)) # at the
beginning, iv = 0
```

זה ה header שאנחנו רוצים שיתקבל בשלב 2, בתחילת הקובץ, אחרי שנעשה encrypt ל tux עם ה IV.

לכן, אנחנו עושים את התהליך ההפוך. בתור התחלה עושים decrypt, ואז עושים xor לכל דבר עם ה header המקורי, כדי לגלות מה ה IV שבאמצעותו בתהליך ההצפנה יתקבל ה header שניסינו ליצור.

```
# decrypt first cipher_block, to get the new iv
    decryptor = cipher.decryptor()
    decrypted_cipher_block1 = decryptor.update(cipher_block1)

    iv = bytes([ decrypted_cipher_block1[i] ^ header1[i] for i in range(16) ])

    print(f"input file1 is: {input_file1.name}\ninput file2 is:
{input_file2.name}")
    print(f"IV is: {list(iv)}")
```

לבסוף, אנחנו עושים decrypt להכל ביחד, וזה התוכן שהמשתמש מקבל

```
cipher = Cipher(algorithms.AES(key), modes.CBC(iv))

    encryptor = cipher.encryptor()
    content1_encrypted = encryptor.update(content1)

    res = content1_encrypted + content2 # becuase i remove the signature,
which is: 0xFFD8FFFE, (it works also without the removing of the target.
interesting. ah, the size of the junk chunk removes the 4 bytes of the
signature, genius.)
```

בס"ד

```
decryptor = cipher.decryptor()

file_decrypted = decryptor.update(res)

output_file.write(file_decrypted)

print(f"output was writted to {output_file.name}")
```

הטריק הזה לקוח מהמקור הבא:

https://www.slideshare.net/slideshow/when-aes-a-cryptobinary-magic-trick/33101365#43

בשלב האחרון של הפרוייקט מנסים לעשות scraping לאינטרנט וצריך להשתמש בהרבה כלים, פשוט להשתמש בגוגל וזהו.

הקבצים שהשתמשתי בהם מצורפים בקובץ zip לתיקייה הגדולה של הכל.

תודה רבה על הפרוייקט הזה! מאוד נהניתי :)