帮助承担tryhackme撰写

翻译

weixin_26755331 于 2020-09-02 17:21:48 发布 40 收藏
文章标签： python
原文链接：https://medium.com/@guragainroshan0/help-bears-tryhackme-writeup-d0e0e7baa1a1

任务1 (Task 1)

There is nothing to do here, but this task is needed for the final task.

此处无事可做，但最终任务需要此任务。

Image for post

任务2 (Task 2)

In this task, the Obfuscated JS needs to be decoded.

在此任务中，需要对混淆的JS进行解码。

Image for post

Here is the content of the given file

这是给定文件的内容

```
É=-~-~[],ó=-~É,Ë=É<<É,þ=Ë+~[];Ì=(ó-ó)[Û=(''+{})[É+ó]+(''+{})[ó-É]+([].ó+'')[ó-É]+(!!''+'')[ó]+({}+'')[ó+ó]+
```

Here the code is obfuscated. We can see a few characters used É, ó, þ, Ì, Û, Ë. Let's beautify the code first and then rename the variables. For beautifying, I used this.

这里的代码被混淆了。 我们可以看到一些使用É，ó，þ，Ì，Û，Ë的字符。 让我们首先美化代码，然后重命名变量。 为了美化，我使用了这个 。

Renaming the variables

重命名变量

```
É : var1
ó : var2
Ë : var3
þ : var4
Ì : var5
Û : var6
```

On renaming the variables we get

重命名变量后，我们得到

```
var1 = -~-~[], var2 = -~var1, var3 = var1 << var1, var4 = var3 + ~[];var5 = (var2 - var2)[var6 = ('' + {})[
```

If we paste the code in the chrome developer console, we get an alert asking for a password.

如果将代码粘贴到chrome开发者控制台中，则会收到提示您输入密码的警报。

Image for post

Image for post

If the wrong password is given it shows fail. So we need to find what code is running. In order to debug the code, I created an HTML file and added the JS in the script tag so that debugging in chrome developer console could be done.

如果密码输入错误，则显示失败。 因此，我们需要查找正在运行的代码。 为了调试代码，我创建了一个HTML文件，并在script标签中添加了JS，以便可以在chrome开发人员控制台中进行调试。

```
<html><script>var1 = -~-~[], var2 = -~var1, var3 = var1 << var1, var4 = var3 + ~[];var5 = (var2 - var2)[var
```

On the chrome developer console, on the sources tab, we can select the HTML file and set a breakpoint. I set two breakpoints and added all variables to the watch section in order to view their values.

在chrome开发人员控制台上，在“来源”标签上，我们可以选择HTML文件并设置一个断点。 我设置了两个断点，并将所有变量添加到监视部分，以查看其值。

Image for post

Setting breakpoints and adding variables to watch.

设置断点并添加要监视的变量。

On refreshing the page, we can see the variables being populated.

在刷新页面时，我们可以看到正在填充的变量。

Image for post

Since the breakpoint is on the second line of JS so the first line sets these values to the variable. On hitting the step over next function button, we get values to var5 and var6.

由于断点在JS的第二行，因此第一行将这些值设置为变量。 按下下一步功能按钮时，我们将获得var5和var6的值。

Image for post

We can replace these values in the HTML file, to make it more readable. Here first line is replaced by values of variables var1,var2,var3,var4,var6.

我们可以在HTML文件中替换这些值，以使其更具可读性。 在此，第一行被变量var1，var2，var3，var4，var6的值替换。

Image for post

Code after replacing variables.

替换变量后的代码。

Var5 is a function so the last line has arguments to the function, lets try to find, what is exactly running.

Var5是一个函数，因此最后一行具有该函数的参数，让我们尝试查找正在运行的内容。

If we assign the parameter to a new variable, we can watch the exact parameter that is being passed. After doing that here is a code.

如果将参数分配给新变量，则可以查看正在传递的确切参数。 完成之后，这里是一个代码。

```
<html><script>var1 = 2;var2 = 3;var3 = 8;var4 = 7;var5 = (var2 - var2)[var6 = "constructor"][var6];r=(!'' +
```

Here, the parameter is assigned to variable r. Now, let's watch the variable r.

在此，参数分配给变量r。 现在，让我们看一下变量r。

Image for post

Here a text is returned.

这里返回一个文本。

```
"return'\141\75\160\162\157\155\160\164\50\47\105\156\164\162\145\172\40\154\145\40\155\157\164\40\144\145\
```

Now if we paste the result in console, we get

现在，如果将结果粘贴到控制台中，我们将得到

Image for post

This way we can get the password.

这样我们可以获取密码。

任务3 (Task 3)

This is the easiest among all the tasks.

这是所有任务中最简单的。

Image for post

Task 3 is quite easy compared to Task 2. The content in the file

与任务2相比，任务3相当容易。文件中的内容

```
var pass = unescape("unescape%28%22String.fromCharCode%2528104%252C68%252C117%252C102%252C106%252C100%252C1
```

The characters seem to be URL encoded. So we need to decode this

字符似乎是URL编码的。 所以我们需要解码

```
var a ='var pass = unescape("unescape%28%22String.fromCharCode%2528104%252C68%252C117%252C102%252C106%252C1
r =decodeURIComponent(a)/*
value of r
"var pass = unescape("unescape("String.fromCharCode%28104%2C68%2C117%2C102%2C106%2C100%2C107%2C105%2C49%2C5
*/
```

Some of the characters are decoded, but some are still in the URL encoded form so decoded it again.

有些字符已解码，但有些字符仍采用URL编码形式，因此请再次对其进行解码。

```
decodeURIComponent(r)
/*
"var pass = unescape("unescape("String.fromCharCode(104,68,117,102,106,100,107,105,49,53,54)")");"
*/
```

On running the output, we get the password.

运行输出时，我们得到密码。

Task3

任务3

Image for post

Since there was a **steg** tag on the challenge, so there must, be something to do with the image in Task 1. I tried most of the tools in order to extract the data. I failed at all of them. So used **stegcracker** to find the password and extracted the data with **steghide .**

由于是在挑战一个**STEG**标签，所以必须是事做任务1中我的形象尝试了大部分工具，以提取数据。 我都失败了。 因此使用**stegcracker**查找密码并使用**steghide**提取数据。

```
$ stegcracker bear.jpg /usr/share/wordlists/rockyou.txt
```

Extracted password as pandas

提取密码为熊猫

```
$ steghide extract -p pandas -sf bear.jpg
wrote extracted data to "challenge.txt".
```

On viewing challenge.txt using cat we just get a word

在使用cat查看Challenge.txt时，我们只需要说一个字

```
$ cat challenge.txt
Grizzly!
```

This didn’t make any sense, so used vim to view the file and got some Unicode characters.

这没有任何意义，因此使用vim查看文件并获得一些Unicode字符。

```
<200c><200c><200c><200c><200d><200c><200d><202c>Grizzly<200c><200c><200c><200c><200d><202c><feff><200c><200
```

On googling around, I found this website which explains Unicode Steganography with Zero-Width Characters. Used this site to decode the text. On directly pasting the characters didn’t work. So, used **xclip** to copy the characters to clipboard and paste the characters there

在四处搜寻时，我发现了这个网站， 该网站介绍了零宽度字符的Unicode隐写术。 使用此站点解码文本。 在直接粘贴字符时无效。 因此，使用**xclip**将字符复制到剪贴板并将字符粘贴到剪贴板

```
$ cat challenge.txt | xclip -selection clipboard
```

Image for post

This time I got the flag. But this was not the flag. The text is encoded. So this needs to be decoded to get the flag. In order to decode, we need to find the encoding used. On googling more about ciphers I found this website explaining various ROT ciphers.

这次我拿到了国旗。 但这不是旗帜。 文本被编码。 因此，需要对此进行解码以获得标志。 为了解码，我们需要找到所使用的编码。 通过进一步搜索密码，我发现该网站解释了各种ROT密码。

Image for post

The encoded text has the properties of ROT47 cipher. Then I used this to decode the given text. Again there is no flag but this string

编码的文本具有ROT47密码的属性。 然后，我用它来解码给定的文本。 再次没有标志，但是这个字符串

```
YjNhcnNfZzAwbmFfcGEkJF90NGVfMFNDUA==
```

By looking at the == we can know its a base64 encoded string. Decoded it to get the flag.

通过查看==，我们可以知道它是base64编码的字符串。 对其进行解码以获取标志。

```
$ echo "YjNhcnNfZzAwbmFfcGEkJF90NGVfMFNDUA==" | base64 -d
```

Finally, this is the flag.

最后，这是标志。

Hope you learned something by reading this writeup. For every JS operation, I used chrome developer console. If you feel anything could be done easily, feel free to suggest.

希望您通过阅读这篇文章学到一些东西。 对于每个JS操作，我都使用了chrome开发人员控制台。 如果您觉得任何事情都可以轻松完成，请随时提出建议。

> 翻译自: https://medium.com/@guragainroshan0/help-bears-tryhackme-writeup-d0e0e7baa1a1