vishwaCTF

CHALLENGE NAME : PROFESSORS INHERITANCE

DEV : KANISHK KUMAR & SATYAJIT BORADE

CATEGORY : STEGNOGRAPHY

LEVEL : MEDIUM-HARD

2024

Question Description:

As we dive deeper into maths it gets harder and harder. Mr. Rick a maths professor has also been facing such challenges. His mentor Mr.Newton left the key to his legacy encrypted in the following files but he has to prove that he is his student to get access to it. Help Mr.Rick solve the questions and decode the key.

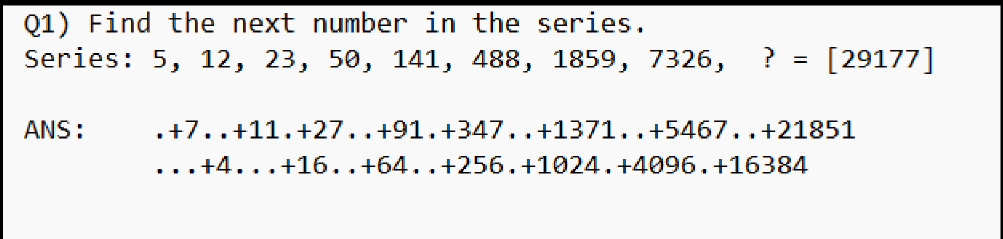
Q1) Find the next number in the series.

Series: 5, 12, 23, 50, 141, 488, 1859, 7326, ?

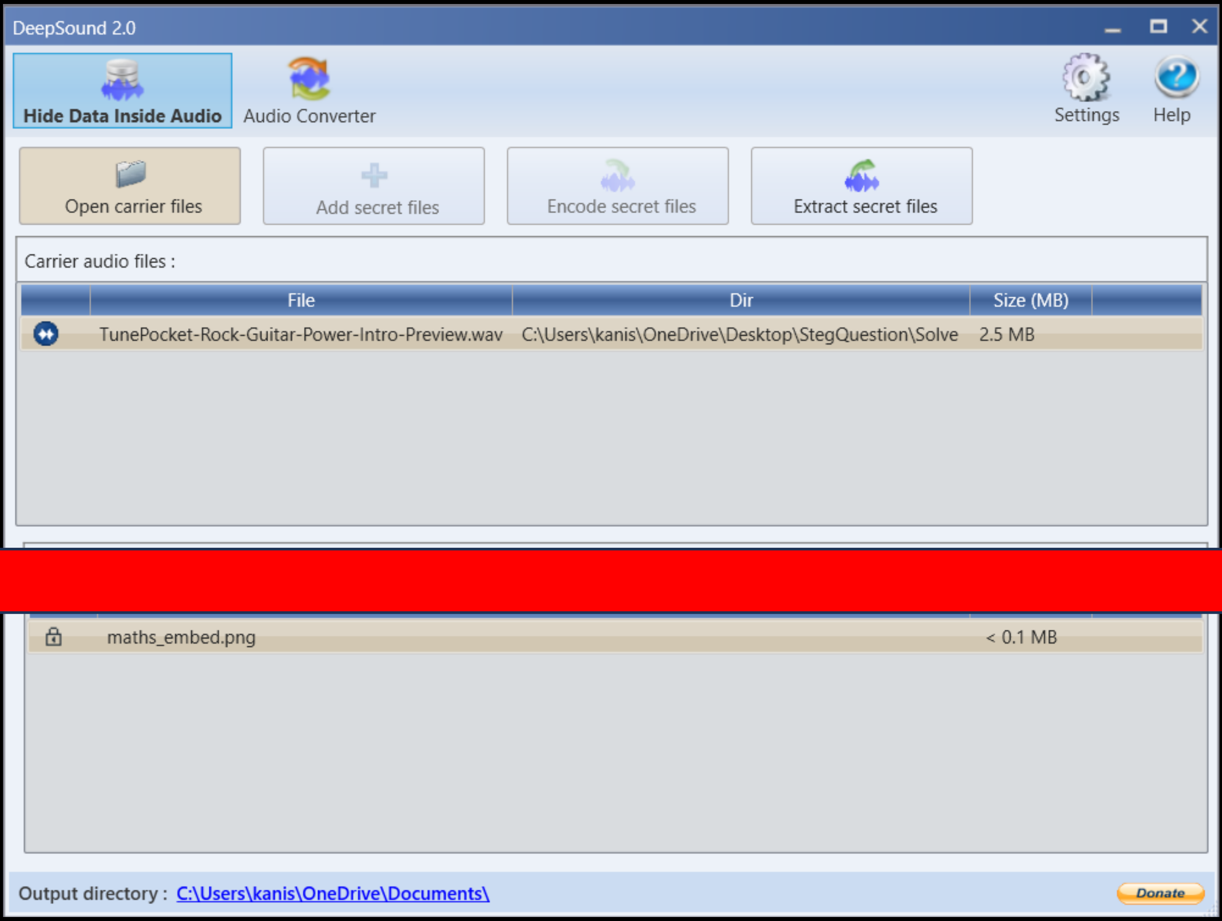
Solution:

Step 1:

Decode the given audio file using deep sound 2.0 and the answer of above question as password : 29177



```
Q1) Find the next number in the series.
Series: 5, 12, 23, 50, 141, 488, 1859, 7326,  ? = [29177]

ANS:    .+7..+11.+27..+91.+347..+1371..+5467..+21851
        ...+4...+16..+64..+256.+1024.+4096.+16384
```



Carrier audio files :

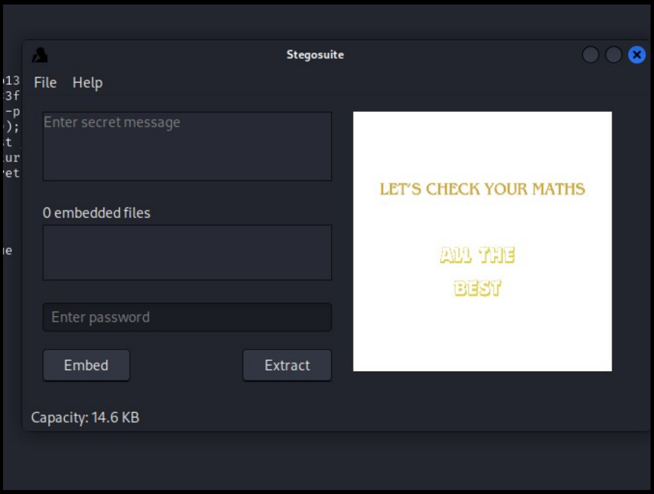
	File	Dir	Size (MB)
	TunePocket-Rock-Guitar-Power-Intro-Preview.wav	C:\Users\kanis\OneDrive\Desktop\StegQuestion\Solve	2.5 MB

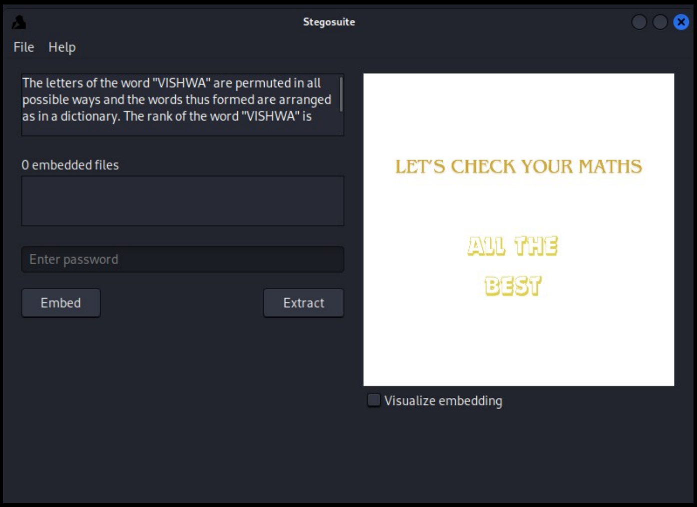
[redacted]

🔒	maths_embed.png		< 0.1 MB

Output directory : C:\Users\kanis\OneDrive\Documents\

Step 2: You will see an image named “maths_embed.png” was encrypted inside the audio file . Decrypt the images using stegosuite gui :





The letters of the word "VISHWA" are permuted in all possible ways and the words thus formed are arranged as in a dictionary. The rank of the word "VISHWA" is

You will find the question saying:

The letters of the word "VISHWA" are permuted in all possible ways and the words thus formed are arranged as in a dictionary. The rank of the word "VISHWA" is

Answer in the following format:

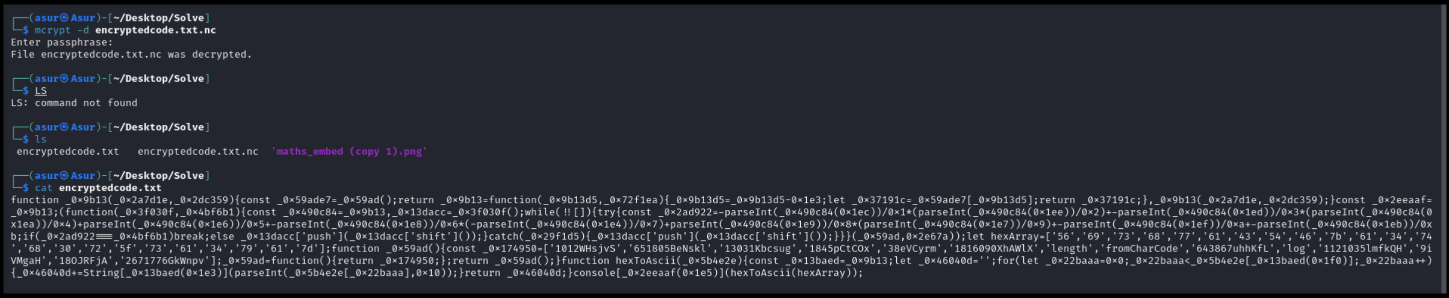
VISHWA-ANS

Example: Answer of above question is 127 then password for next file is:

VISHWA-127

The answer to this would be: VISHWA-545

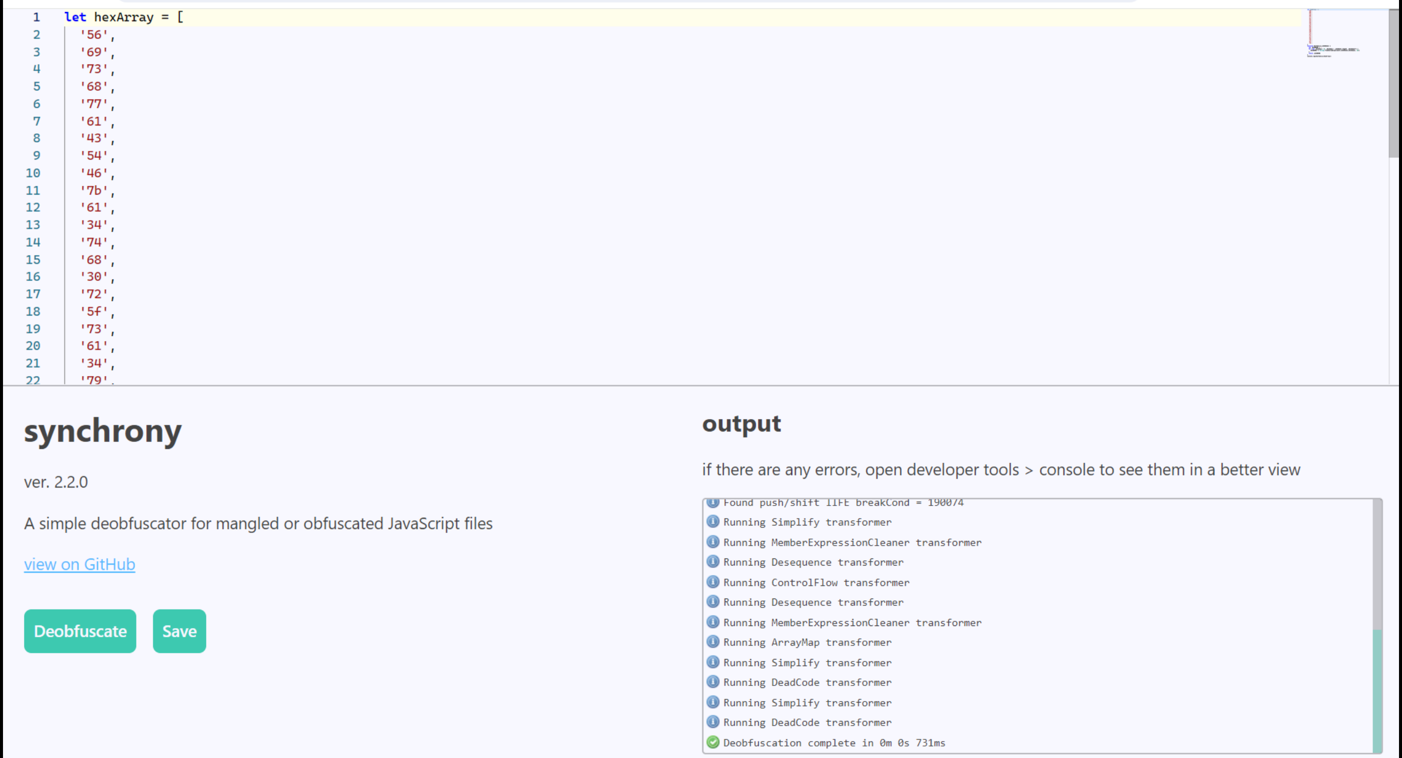
Step 3: Decrypt the “encryptedcode.txt.nc” file given along with the mp3 file , using the command `mcrypt -d encryptedcode.txt.nc ` and passphrase: VISHWA-545



```
┌──(asur㉿Asur)-[~/Desktop/Solve]
└─$ mcrypt -d encryptedcode.txt.nc
Enter passphrase:
File encryptedcode.txt.nc was decrypted.

┌──(asur㉿Asur)-[~/Desktop/Solve]
└─$ LS
LS: command not found

┌──(asur㉿Asur)-[~/Desktop/Solve]
└─$ ls
 encryptedcode.txt   encryptedcode.txt.nc  'maths_embed (copy 1).png'

┌──(asur㉿Asur)-[~/Desktop/Solve]
└─$ cat encryptedcode.txt
function _0×9b13(_0×2a7d1e,_0×2dc359){const _0×59ade7=_0×59ad();return _0×9b13=function(_0×9b13d5,_0×72f1ea){_0×9b13d5=_0×9b13d5-0×1e3;let _0×37191c=_0×59ade7[_0×9b13d5];return _0×37191c;},_0×9b13(_0×2a7d1e,_0×2dc359);}const _0×2eeaaf=
_0×9b13;(function(_0×3f030f,_0×4bf6b1){const _0×490c84=_0×9b13,_0×13dacc=_0×3f030f();while(!![]){try{const _0×2ad922=-parseInt(_0×490c84(0×1ec))/0×1*(parseInt(_0×490c84(0×1ee))/0×2)+-parseInt(_0×490c84(0×1ed))/0×3*(parseInt(_0×490c84(0
x1ea))/0×4)+parseInt(_0×490c84(0×1e6))/0×5+-parseInt(_0×490c84(0×1e8))/0×6*(-parseInt(_0×490c84(0×1e4))/0×7)+parseInt(_0×490c84(0×1e9))/0×8*(parseInt(_0×490c84(0×1e7))/0×9)+-parseInt(_0×490c84(0×1ef))/0×a+-parseInt(_0×490c84(0×1eb))/0x
b;if(_0×2ad922===_0×4bf6b1)break;else _0×13dacc['push'](_0×13dacc['shift']());}catch(_0×29f1d5){_0×13dacc['push'](_0×13dacc['shift']());}}}(_0×59ad,0×2e67a));let hexArray=['56','69','73','68','77','61','43','54','46','7b','61','34','74
','68','30','72','5f','73','61','34','79','61','7d'];function _0×59ad(){const _0×174950=['1012WHsjvS','651805BeNskl','13031Kbcsug','1845pCtCOx','38eVCyrm','1816090XhAWlX','length','fromCharCode','643867uhhKfL','log','1121035lmfkQH','9i
VMgaH','18OJRFjA','2671776GkWnpv'];_0×59ad=function(){return _0×174950;};return _0×59ad();}function hexToAscii(_0×5b4e2e){const _0×13baed=_0×9b13;let _0×46040d='';for(let _0×22baaa=0×0;_0×22baaa<_0×5b4e2e[_0×13baed(0×1f0)];_0×22baaa++)
{_0×46040d+=String[_0×13baed(0×1e3)](parseInt(_0×5b4e2e[_0×22baaa],0×10));}return _0×46040d;}console[_0×2eeaaf(0×1e5)](hexToAscii(hexArray));
```

Step 4: You will find an obfuscated JS function. You can use an online deobfuscater like :

https://deobfuscate.relative.im/ to deobfuscate the function.



```
let hexArray = [
  '56',
  '69',
  '73',
  '68',
  '77',
  '61',
  '43',
  '54',
  '46',
  '7b',
  '61',
  '34',
  '74',
  '68',
  '30',
  '72',
  '5f',
  '73',
  '61',
  '34',
  '79',
```

synchrony

ver. 2.2.0

A simple deobfuscator for mangled or obfuscated JavaScript files

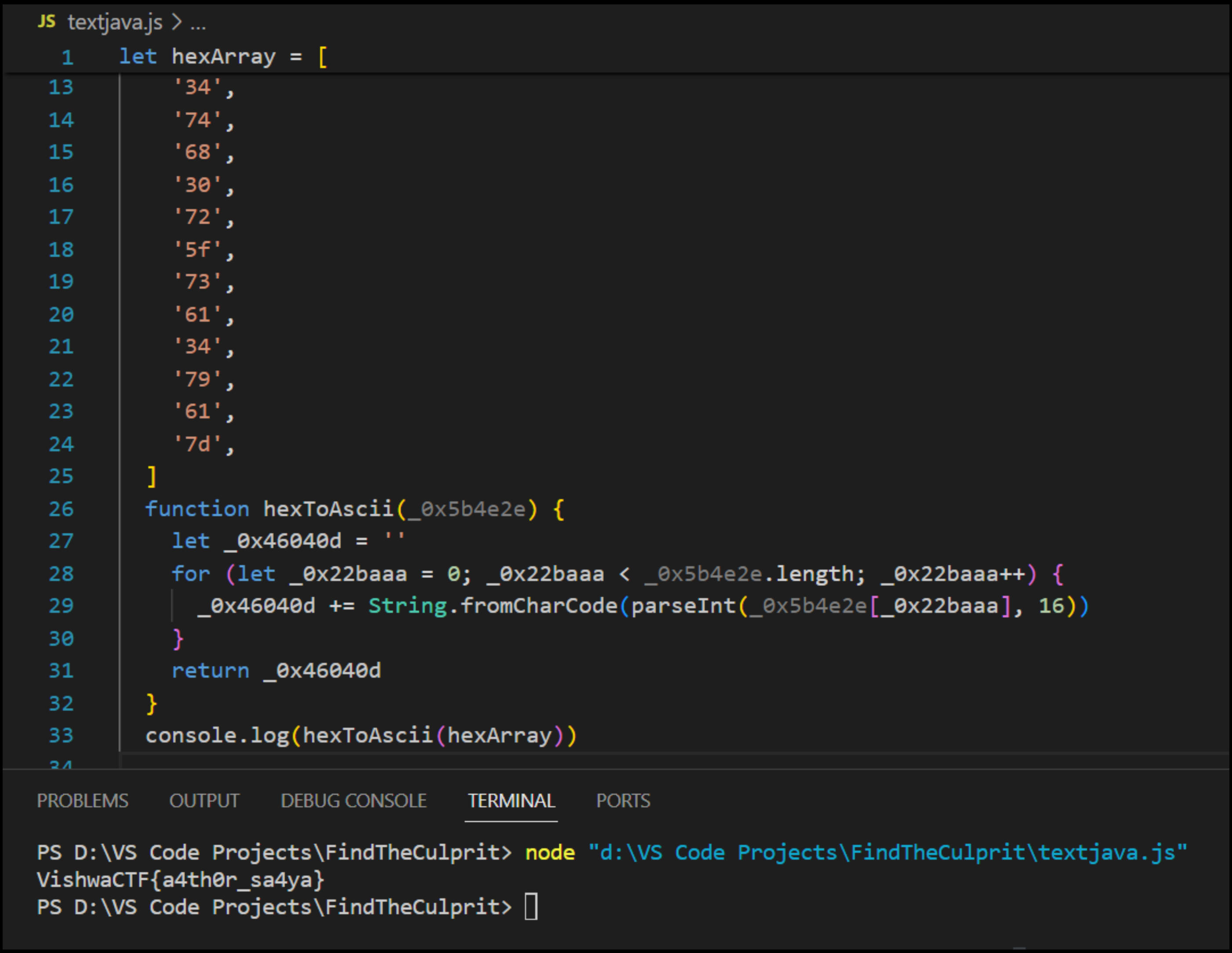
view on GitHub

output

if there are any errors, open developer tools > console to see them in a better view

```
Found push/shift IIFE breakCond = 190074
Running Simplify transformer
Running MemberExpressionCleaner transformer
Running Desequence transformer
Running ControlFlow transformer
Running Desequence transformer
Running MemberExpressionCleaner transformer
Running ArrayMap transformer
Running Simplify transformer
Running DeadCode transformer
Running Simplify transformer
Running DeadCode transformer
Deobfuscation complete in 0m 0s 731ms
```

Just run the hexArray function in VS code or any other compiler to get the flag.



```
let hexArray = [
    '34',
    '74',
    '68',
    '30',
    '72',
    '5f',
    '73',
    '61',
    '34',
    '79',
    '61',
    '7d',
  ]
  function hexToAscii(_0x5b4e2e) {
    let _0x46040d = ''
    for (let _0x22baaa = 0; _0x22baaa < _0x5b4e2e.length; _0x22baaa++) {
      _0x46040d += String.fromCharCode(parseInt(_0x5b4e2e[_0x22baaa], 16))
    }
    return _0x46040d
  }
  console.log(hexToAscii(hexArray))
```

```
PS D:\VS Code Projects\FindTheCulprit> node "d:\VS Code Projects\FindTheCulprit\textjava.js"
VishwaCTF{a4th0r_sa4ya}
PS D:\VS Code Projects\FindTheCulprit> 
```