DamCTF - 10/09/2020 - 10/11/2020

web/finger-warmup

This was an individual effort

The website at damctf.xyz is no longer up, so I could not link to the challenge.

Description:

> A finger warmup to prepare for the rest of the CTF, good luck!
>
> You may find this or this to be helpful.

Solution:

1. Clicking the link on the page at https://finger-warmup.chals.damctf.xyz/ (no longer up), there's a link to a similar looking page with a different url.
2. Doing this continuously kept bringing up similar pages, and the pages had the instruction to keep clicking the link on the page
3. I initially wrote a script to automate this process in AutoHotKey, which would continuosly click on the same part of the page. The script is as follows:

```
#NoEnv  ; Recommended for performance and compatibility with future
AutoHotkey releases.
; #Warn  ; Enable warnings to assist with detecting common errors.
SendMode Input  ; Recommended for new scripts due to its superior speed and
reliability.
SetWorkingDir %A_ScriptDir%  ; Ensures a consistent starting directory.
#MaxThreadsPerHotkey 3

^z::
Toggle := !Toggle
Loop
{
    If (!Toggle)
        Break
    Click
    Sleep 83 ; Make this number higher for slower clicks, lower for faster.
}
```

4. After a while, I decided to switch to python, so I could free up my mouse pointer and run the scipt in the background. The python script follows the link on each page until it finds a page with different contents. The script is as follows:

```
import requests

i=1
cats = "wrong"
href = 'jdaqfazy4wafobkn2hi3nt'
res = requests.get('https://finger-warmup.chals.damctf.xyz/' + href).text
while res.find('click here, if you are patient enough I will give you the
flag') > -1:
    href = res[9:res.find('">c')]
    res = requests.get('https://finger-warmup.chals.damctf.xyz/' +
href).text
    i += 1
    if i > 1500:
        print('https://finger-warmup.chals.damctf.xyz/' + href)
        i = 1
        f = input("continue?")
print('https://finger-warmup.chals.damctf.xyz/' + href)
```

5. Eventually, this took us to a page at https://finger-warmup.chals.damctf.xyz/giz93go1o1bpg4avsls5gh which I have archived at https://archive.vn/DdGeO where the flag was revealed.

dam{I_hope_you_did_this_manually}

misc/pretty-good (OSINT)

This was an asynchronous group effort, but I'm not sure if the other people were in CIS4204

The website at damctf.xyz is no longer up, so I could not link to the challenge.

Description:

> Nothing like getting a project assignment on a Friday afternoon.
>
> Flag is the IP address (ex: dam{169.254.42.42})

Solution:

1. A rules document was linked here: https://docs.google.com/document/d/199VdTFQ9BAn8ewlsoV7_LEeFbh1LqHCixX9v7_fXlSw/
2. Following the attached pdf (no longer can be found), there was an attached picture. A reverse image search led to a blog, whose archive we found here: https://web.archive.org/web/20201005005536/https://rayhaanhodgson.com/.
3. It was mentioned on the blog that the author had a GitHub profile.
4. Opening up GitHub pages related to the author's name and email revealed a GitHub commit with deleted sensitive info. (The relevant GitHub page is no longer up)
5. Viewing the diff showed an IP address, which was the flag.

dam{182.24.89.5}

web/there-is-no-war-in-graphql

This was an individual effort

The website at damctf.xyz is no longer up, so I could not link to the challenge.

Description:

> I did not record the exact challenge description, but here is a summary:
>
> Given a GraphQL explorer (at graphql.chals.damctf.xyz/graphql, no longer up), find when the next Eclipse is.

Solution:

1. Snooping around the schema and using recommended fields gave an idea of how the API worked.
2. In the rules document, point 3 gives an example flag.
3. Creating a query for planetariumToday gave an idea of what the date format looks like.

   { day: d, month: m, year: y (99 was "today"), era: 'AG' }
4. As much as I tried I couldn't get planetariumBatch to output the future eclipse dates
5. Instead, I wrote a script in node.js that queries planetariumOne for each upcoming date, one-by-one, until I found one with an eclipse. The script is as follows:

```
const fetch = require("node-fetch");

var going=false;

let oneQuery = (day, month, year, era) => {
    return `query EclipsesOne {
    planetariumOne(
      date: {
          day: ${day},
          month: ${month},
          year: ${year},
          era: "AG"
      }
    ){
      date{
        day
        month
        year
        era
      }
      eclipse
      __typename
    }
  }`;
}

const checkEclipse = (planetariumOne) => {
    if(planetariumOne.eclipse === true){
        console.log(planetariumOne.date)
        going=true
        return planetariumOne
    }
    if(planetariumOne.eclipse === false){
        return false
    }
    console.log("rip")
    return "error"
}


const sendQuery = (query) => {
    fetch('https://graphql.chals.damctf.xyz/graphql?', {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json',
          'Accept': 'application/json'
        },
        body: JSON.stringify({
          query
        })
    })
    .then(r => r.json())
    .then(result => checkEclipse(result.data.planetariumOne))
}

sendQuery(oneQuery(1,1,100,"AG"))

y=99
while (y<103 && !going){
    for(i=1; i<=12; ++i){
        for(j=1; j<=28; ++j){
            sendQuery(oneQuery(j,i,y,"AG"))
        }
        if([1,3,4,5,6,7,8,9,10,11,12].includes(i)){
            sendQuery(oneQuery(29,i,y,"AG"))
            sendQuery(oneQuery(30,i,y,"AG"))
            if([1,3,5,7,8,10,12].includes(i)){
                sendQuery(oneQuery(31,i,y,"AG"))
            }
        }
        else if((y%4 == 0) && (y%100 != 0)){
            sendQuery(oneQuery(29,i,y,"AG"))
        }
    }
    console.log(y)
    ++y
}
```

6. Turns out the next eclipse is on { day: 11, month: 7, year: 104, era: 'AG' }
7. Running planetariumOne with this date revealed the flag.

dam{InvAd3_w1T0ut_4_w4R}

CyberYoddha CTF 2020 - 10/30/2020 - 11/01/2020

trivia/Trivia 1

This was an individual effort

The website at https://cyberyoddha.baycyber.net/ is no longer up, so I could not link to the challenge.

Description:

Who created linux? {no wrapper needed}

Solution:

1. Googled "creator of linux"
2. The answer was Google's feature snippet

Linus Torvalds

trivia/Trivia 2

This was an individual effort

The website at https://cyberyoddha.baycyber.net/ is no longer up, so I could not link to the challenge.

Description:

Who’s operating system was IBM going to buy before MS-DOS? {no wrapper needed}

Solution:

1. Googled the description as-is
2. This led to a link at https://arstechnica.com/gadgets/2017/07/ibm-pc-history-part-2/
3. I tried every name on that website until I got an answer

Gary Kildall

trivia/Trivia 3

This was an individual effort

The website at https://cyberyoddha.baycyber.net/ is no longer up, so I could not link to the challenge.

Description:

Which company invented the original hadoop software? {no wrapper needed}

Solution:

1. Searched for "hadoop" on Wikipedia
2. One of the notable companies who worked on Wikipedia in this article was Yahoo! https://en.wikipedia.org/wiki/Apache_Hadoop

Yahoo

trivia/Trivia 4

This was an individual effort

The website at https://cyberyoddha.baycyber.net/ is no longer up, so I could not link to the challenge.

Description:

Microsoft has been threatened by a secret hacker for the last couple of years. This hacker has been infiltrating their network and exposing secret information about them to the world. Microsoft is determined to catch this hacker. They set up a computer with vulnerabilities and attempt to lure this hacker into trying to hack this computer in order to reveal his origins. What is this type of system called? {no wrapper needed}

Solution:

1. Googled "attempt to lure hackers"
2. This brought me to a website at https://www.csoonline.com/article/3384702/what-is-a-honeypot-a-trap-for-catching-hackers-in-the-act.html which contained the name of the term

Honeypot

trivia/Trivia 6

This was an individual effort

The website at https://cyberyoddha.baycyber.net/ is no longer up, so I could not link to the challenge.

Description:

A Hacker infiltrated one of Microsoft's servers and set up malware inside. The malware laid dormant for months, being unnoticed by the server admins. On Thanksgiving Day, the malware was activated, and it crashed all of the servers and the entire network. What is this type of malware called?{no wrapper needed}

Solution:

1. Googled "dormant malware that eventually crashes servers"
2. This brought me to a website at https://www.uscybersecurity.net/logic-bombs/ whose title contained the name of the term

Logic Bomb

trivia/Trivia 7

This was an individual effort

The website at https://cyberyoddha.baycyber.net/ is no longer up, so I could not link to the challenge.

Description:

What built-in Windows tool can you use to repair corrupted files? {no wrapper needed}

Solution:

1. Googled "built-in Windows tool can you use to repair corrupted files"
2. This brought me to a Microsoft Support post at https://support.microsoft.com/en-us/help/929833/use-the-system-file-checker-tool-to-repair-missing-or-corrupted-system whose title contained the name of the term

System File Checker

trivia/Trivia 8

This was an individual effort

The website at https://cyberyoddha.baycyber.net/ is no longer up, so I could not link to the challenge.

Description:

A Hacker infiltrated one of Microsoft's servers and set up malware inside. The malware laid dormant for months, being unnoticed by the server admins. On Thanksgiving Day, the malware was activated, and it crashed all of the servers and the entire network. What is this type of malware called? {no wrapper needed}

Solution:

1. There was an image attached which I put into a reverse image search (image is no longer available)
2. Google's related search included the phrase "Haskell icon" and I I knew that Haskell was a programming language

Haskell

Forensics/Flag Delivery

This was an individual effort

The website at https://cyberyoddha.baycyber.net/ is no longer up, so I could not link to the challenge.

Description:

Our good friend Yeltsa Kcir ordered a flag for us from the renowned flag delivery service. We got their letter today, but we can’t see the flag they sent us. Apparently Yeltsa has been talking with the scientist Odec Esrom. Can you find the flag he hid for us?

Solution:

1. A file was attached with encoded text (the characters {"D", " ", "?", "M", "6" } were the only ones). Unfortunately, I was not able to save this file.
2. Odec Esrom is "morse code" reversed (almost).
3. Since Morse Code has only two characters (3 with word breaks), I just had to look for two items to assign dots and dashes to. "D" and "?M6" looked to be the two.
4. After trying the scheme, D to dash, ?M6 to dot, we had a morse code string.
5. Decoding it revealed the flag

CYCTFR3@D_B3TW33N_TH3_L1N3S

Password Cracking/secure (I think?)

This was an individual effort

The website at https://cyberyoddha.baycyber.net/ is no longer up, so I could not link to the challenge.

Description:

> smh I can't even crack this password: b0439fae31f8cbba6294af86234d5a28 Note: Don't use a flag wrapper for this challenge

Solution:

1. We were given an hash to crack
2. I googled "hash cracker" and that brought me to https://crackstation.net/
3. Entering the unsalted hash, revealed that it's an md5 hash and revealed the result, which was the flag.

securepassword

Affinity CTF Lite - 11/16/2020 - 11/17/2020

stego/astatine

I was aided on this one by noopnoop, who hinted that Astatine was code for ASCII85.

The website at https://2020.affinityctf.com/challenges is no longer up, so I could not link to the challenge.

Description:

> Can you read the message?

5t4=2<(;4P0Q^YXDIYA21Ltn

Solution:

1. The title of the challenge, "Astatine" is a hint. Astatine is 85th on the periodic table.
2. ASCII85 is an existing encoding method
3. A converter from between text and scii85 can be found here: https://cryptii.com/pipes/ascii85-encoding . One could use this to decode and get the flag from the given string.

AFFCTF{n0t_3nc0d3d}

Perfect Blue CTF - 12/04/2020 - 12/06/2020

misc/not-stego

I was aided on this one by noopnoop, who suggested that the bytes in the data section should interpreted as text.

This ctf did not have individually linked challenges, so I could not link to the challenge. The ctf was hosted at https://ctf.perfect.blue/.

Description:

Hallmark of a good CTF is inclusion of Steganography. You asked and we delivered, or didn't we?

Solution:

1. A disassembled .data section of a file was included as an image.
2. Typing the hex byte data into a converter, we got a string of text: "Here's my my link: https://pastebin.com/j6Xd9GNM <-- Hehe! See if you can RE me"
3. Following the link had a file with the contents: "Here's a flag for your efforts: pbctf{3nc0d1ng_w1th_ass3mbly}."

pbctf{3nc0d1ng_w1th_ass3mbly}

crypto/Ainissesthai

This was an individual effort

This ctf did not have individually linked challenges, so I could not link to the challenge. The ctf was hosted at https://ctf.perfect.blue/.

Description:

A team of codebreakers had to invent computers to break this cipher. Can you figure out what the flag is?

Remote: nc ainissesthai.chal.perfect.blue 1
Note: enter flag as pbctf{UPPERCASEPLAINTEXT}

Solution:

1. A few pieces of evidence led me to beleive that this was an Enigma Machine challenge:

   Googling "Ainissesthai" brings up a website https://wordinfo.info/results/from%20ainissesthai which mentions enigma

   The Enigma Machine generally takes in all capital letters with no other characters,

   The provided source code had Enigma Machine in it

2. From prior knowledge, I knew the problem with the Enigma Machine was that a letter could not map to itself. By running the service that provided the enigma output multiple times, I could determine which letters *couldn't* be in the output. Once 25 letters couldn't be in any each spot, I knew what the remaining character must be.

3. I wrote a python script that strips the newlines from the output, so I could check character-by-character:

   ```
   import sys

   text = "\""
   for line in sys.stdin:
       text = text + line
       if len(text)>=899:
           break

   text = text.replace('\n', '')
   text = text + "\",\n"

   outF = open("alltxt.txt", "a")
   outF.write(text)
   print(text)
   ```

   To run this from the terminal, I piped the output into the python script with:

   ```
   nc ainissesthai.chal.perfect.blue 1 | python3 enigmatext.py
   ```

4. To run this multiple times from the terminal, I wrote an autohotkey script that repeatedly ran the above command by closing the connection and then resending the command:

   ```
   #NoEnv  ; Recommended for performance and compatibility with future
   AutoHotkey releases.
   ; #Warn  ; Enable warnings to assist with detecting common errors.
   SendMode Input  ; Recommended for new scripts due to its superior speed and
   reliability.
   SetWorkingDir %A_ScriptDir%  ; Ensures a consistent starting directory.
   #MaxThreadsPerHotkey 3

   ^k::
   Toggle := !Toggle
   Loop
   {
       If (!Toggle)
           Break
       Send ^{C}
       Sleep 83 ;
   ```

```
    Send {Up}
    Sleep 83 ;
    Send {Enter}
    Sleep 500 ;
}
```

5. After a while, I sent the input to a JavaScript file which used the previous outputs to determine which possible letters could be the ones used in the original text.

```
let a = [...] //nc output text entries
let letts = [

"A","B","C","D","E","F","G","H","I","J","K","L","M","N","O","P","Q","R","S",
"T","U","V","W","X","Y","Z"
]

//Array of empty arrays
let arrs = [];
for(let i =0; i<a[0].length; ++i){
    arrs.push([])
}

//adds letter to spot in array if it shows up in one of the texts
for (let i = 0; i<a.length; ++i){
    for (let j = 0; j<a[i].length; ++j){
        if(!arrs[j].includes(a[i][j])){
            arrs[j].push(a[i][j])
        }
    }
}


let brrs = [];
for(let i =0; i<a[0].length; ++i){
    brrs.push([])
}

//adds letter to spot in array if it doesn't up in one of the texts
for (let i = 0; i<arrs.length; ++i){
    for (let j = 0; j<letts.length; ++j){
        if(!arrs[i].includes(letts[j])){
            brrs[i].push(letts[j])
        }
    }
}

let ans=""
for (let i = 0; i<brrs.length; ++i){
    if (brrs[i].length == 1){
        ans+=brrs[i][0]
    }
    else{
        ans+="_"
    }
}
console.log(ans)
```

6. After running it with a long input string, I got a partially completed result:

```
__TALFLAWINE_I_MAFATA__LA_IN__I_MA__T_L__A_INENIGMAFATAL__AW_NEN__M_FA_ALFLA
W_NENIGMA_AT_LF_A_INE_I_MA____LFLA__NE_IGMAFA_ALFLAWINE__GMA_ATALFLA__N_NI_M
_F_TALF_AW____I__AFA_A_FL_W__ENIG_A__TALF_A_IN__I_MAF_TA_FL_WIN_NIG_AFATALFL
A_I_E__GMAFATALF_AW_NE_IG_A____LF_A__NENIG_AFATAL_LAW_N_NI_MA__TA__LA_INEN_G
MAFATAL_LAW__EN__M__AT___LAWINENIGMA__TALFLA_I___I_M____ALFLAWINENI_MA___ALF
_AWIN_NIG_AFATA__LAWINENIGM_FATALFLAW_N_NIGM___T__FLAWI__NIG__FAT_L__A__NENI
_M_FATAL___W__ENI_MAFA__L_LAWINE_I_MAF_T_LF___I_E_I_MAFA_AL__A_INENIGMAF_T_L
FLAWIN__IGMAFAT_L___W_NENI_MAF_T_LFL_W_NE__GMA_AT___LAWINENIGMAF_TA___A__NEN
I_MA_A____LA_IN___G__FA_ALFLAW_NENIG_AFAT____AWINE_IG_A_ATA_FLA__N_N_GMAFAT_
_FL_WINENI__AF__A_F__WINENIGM_FAT_L__AWINENIGMAFATALFLA_I__NIG__F_TA_FL_W__E
_IG_A_A__L_LAWINEN_G_A_ATALF__WI_EN_GMAF_TA_FLAWI__NI_MAFATALF__W__ENIGM___T
_L_LAW_NENI_MA
```

7. By inspection, this was enough to figure out that the phrase "FATALFLAWINENIGMA" was being repeated over and over again.

pbctf{FATALFLAWINENIGMA}