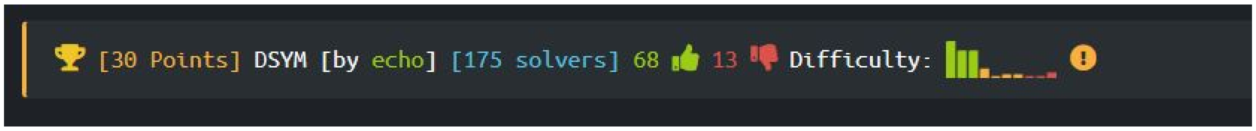
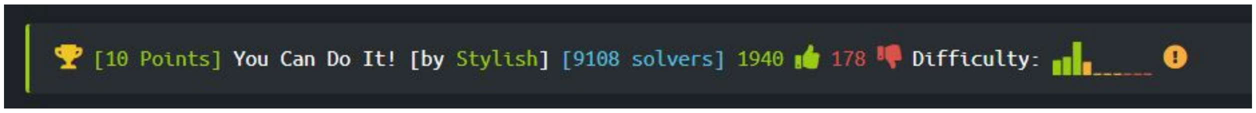
CS 373 Final Write-Up
James Tyler Ball

SUMMARY

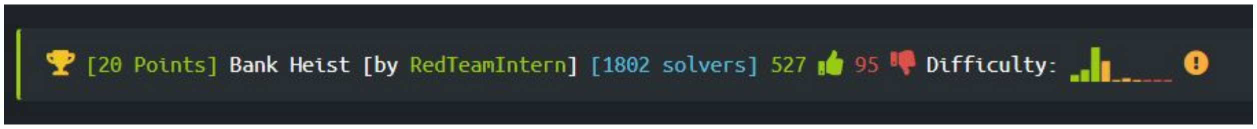
For the hack the box final, at the time of writing I completed Bank Heist, You can do it, and DSYM for a total of 60 points. My profile name for confirmation is Finbonkle



[30 Points] DSYM [by echo] [175 solvers] 68 13 Difficulty:



[10 Points] You Can Do It! [by Stylish] [9108 solvers] 1940 178 Difficulty:

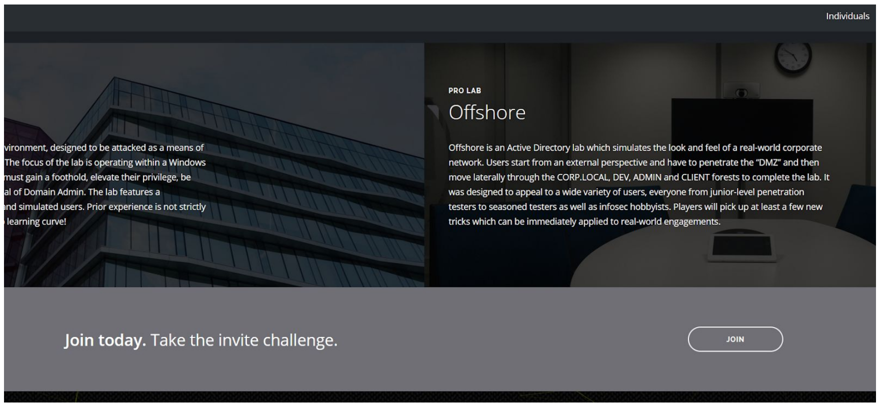


[20 Points] Bank Heist [by RedTeamIntern] [1802 solvers] 527 95 Difficulty:

https://www.hackthebox.eu/home/users/profile/161652

OBTAIN AN ACCOUNT ON HACK THE BOX

To start, I followed the “Individuals” link, and scrolled down until I found the option to join:



Individuals

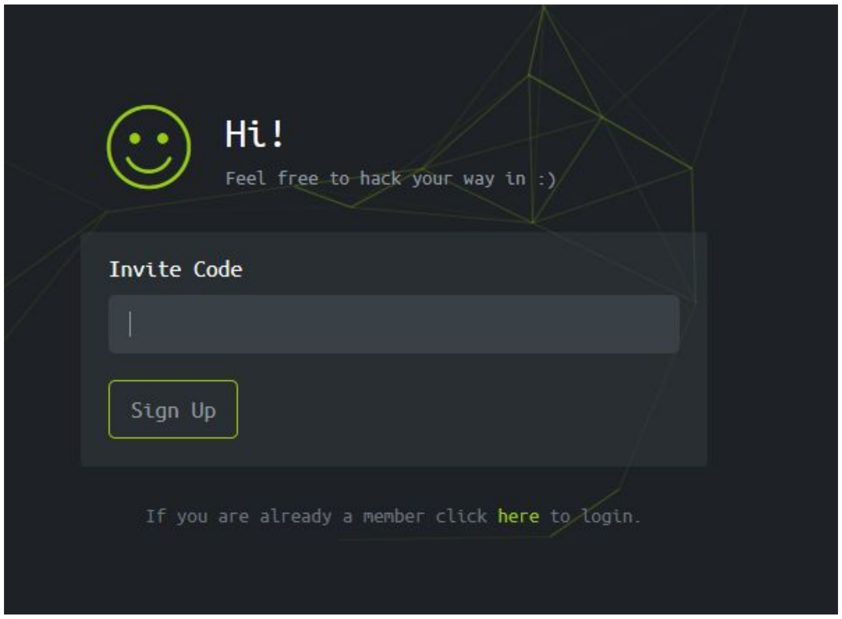
vironment, designed to be attacked as a means of
The focus of the lab is operating within a Windows
must gain a foothold, elevate their privilege, be
al of Domain Admin. The lab features a
nd simulated users. Prior experience is not strictly
learning curve!

PRO LAB

Offshore

Offshore is an Active Directory lab which simulates the look and feel of a real-world corporate network. Users start from an external perspective and have to penetrate the “DMZ” and then move laterally through the CORP.LOCAL, DEV, ADMIN and CLIENT forests to complete the lab. It was designed to appeal to a wide variety of users, everyone from junior-level penetration testers to seasoned testers as well as infosec hobbyists. Players will pick up at least a few new tricks which can be immediately applied to real-world engagements.

Join today. Take the invite challenge.

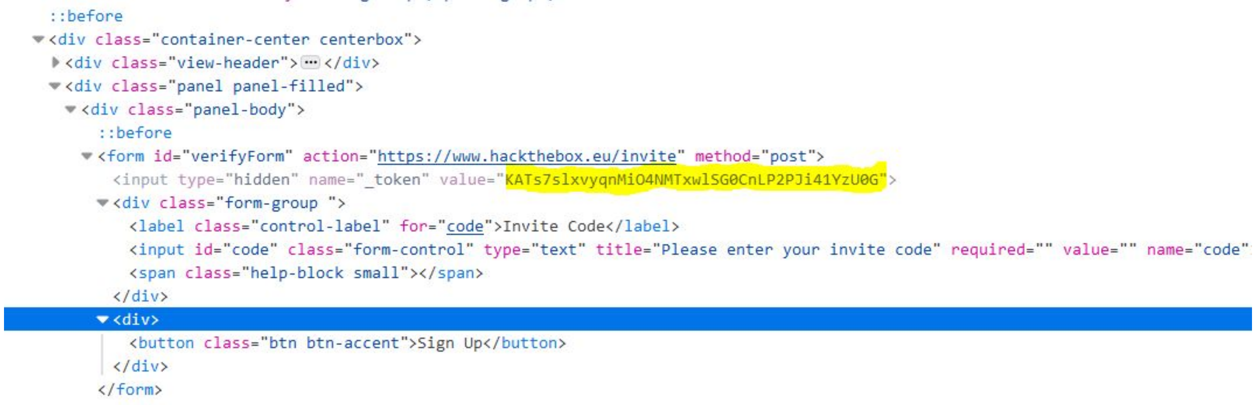


Hi!

Feel free to hack your way in :)

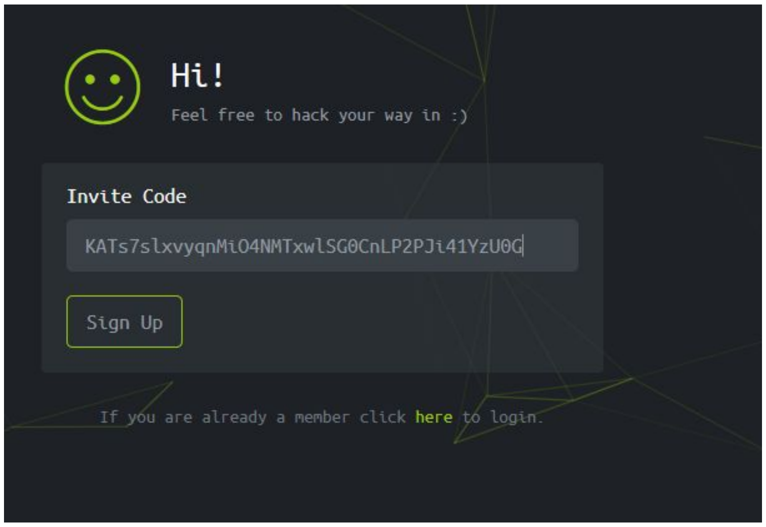
If you are already a member click here to login.

I’m invited to “hack my way in”. My toolset is relatively limited as a novice, but I know from web development classes that I can take a more technical look at this page:



```
    ::before
  ▼<div class="container-center centerbox">
    ▶<div class="view-header">⋯</div>
    ▼<div class="panel panel-filled">
      ▼<div class="panel-body">
          ::before
        ▼<form id="verifyForm" action="https://www.hackthebox.eu/invite" method="post">
            <input type="hidden" name="_token" value="KATs7slxvyqnMiO4NMTxwlSG0CnLP2PJi41YzU0G">
          ▼<div class="form-group ">
              <label class="control-label" for="code">Invite Code</label>
              <input id="code" class="form-control" type="text" title="Please enter your invite code" required="" value="" name="code"
              <span class="help-block small"></span>
            </div>
          ▼<div>
              <button class="btn btn-accent">Sign Up</button>
            </div>
          </form>
```

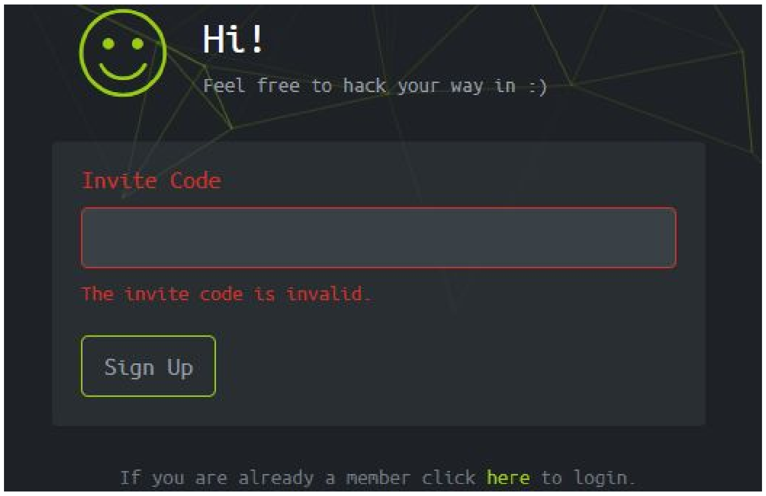
There is a very odd looking string in the html, and I doubt we'll get points off for getting an account via guess and check…



```
KATs7slxvyqnMiO4NMTxwlSG0CnLP2PJi41YzU0G
```

If you are already a member click here to login.

That didn't work.



The invite code is invalid.

If you are already a member click here to login.

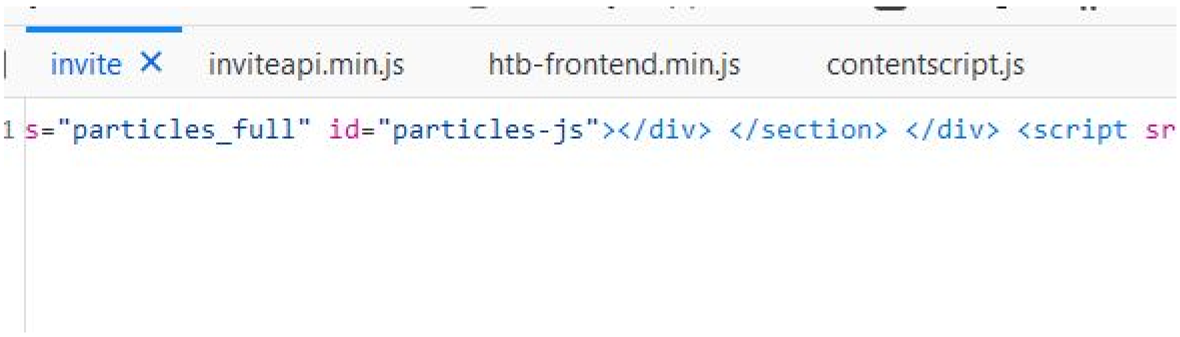
I don't want to turn to Google just yet, so let's dig through the rest of the page. At no point do I see anything obvious like a commented out code or link to send invites. Let's keep digging…

On the Console tab is this spooky image:

```
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
.                                                                                 .
.                      uuuuuuu                                                    .
.                  uu$$$$$$$$$$$uu                                                .
.               uu$$$$$$$$$$$$$$$$$uu                                             .
.              u$$$$$$$$$$$$$$$$$$$$$u                                            .
.             u$$$$$$$$$$$$$$$$$$$$$$$u                                           .
.            u$$$$$$$$$$$$$$$$$$$$$$$$$u                   K E E P  C A L M      .
.            u$$$$$$$$$$$$$$$$$$$$$$$$$u                                          .
.            u$$$$$$"   "$$$"   "$$$$$$u                        A N D             .
.            "$$$$"      u$u       $$$$"                                          .
.             $$$u       u$u       u$$$                        H A C K            .
.             $$$u      u$$$u      u$$$                                           .
.              "$$$$uu$$$   $$$uu$$$$"                         T H I S            .
.               "$$$$$$$"   "$$$$$$$"                                             .
.                 u$$$$$$$u$$$$$$$u                             B O X             .
.                  u$"$"$"$"$"$"$u                                                .
.       uuu        $$u$ $ $ $ $u$$       uuu                                      .
.      u$$$$        $$$$$u$u$u$$$       u$$$$                                     .
.       $$$$$uu      "$$$$$$$$$"     uu$$$$$$                                     .
.     u$$$$$$$$$$$uu    """""    uuuu$$$$$$$$$$                                   .
.     $$$$"""$$$$$$$$$$uuu   uu$$$$$$$$$"""$$$"                                   .
.      """      ""$$$$$$$$$$$uu ""$"""                                            .
.                uuuu ""$$$$$$$$$$uuu                                             .
.       u$$$uuu$$$$$$$$$uu ""$$$$$$$$$$$uuu$$$                                    .
.       $$$$$$$$$$""""           ""$$$$$$$$$$$"           HackTheBox v0.9.3       .
.        "$$$$$"                      ""$$$$""           info@hackthebox.eu       .
.          $$$"                         $$$$"                                     .
.                                                                                 .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

Not much to be gained at first glance.

On the debugger page is some js code. I'm going to sort through this like I did the html.

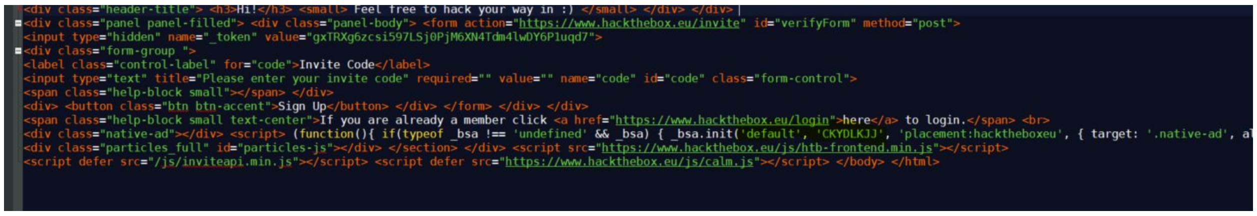


```
s="particles_full" id="particles-js"></div> </section> </div> <script sr
```

The first tab is html and says “invite”. Seems like a good place to start. I’ll copy this into notepad++. Everything is on one line, so I’ll start inserting newlines. A few things jump out at me.

- First, there are a lot of long, unintuitive strings like the first one we tried, and I don’t want to try them all if I can help it.
- Second, there aren’t a lot of comments on this page.

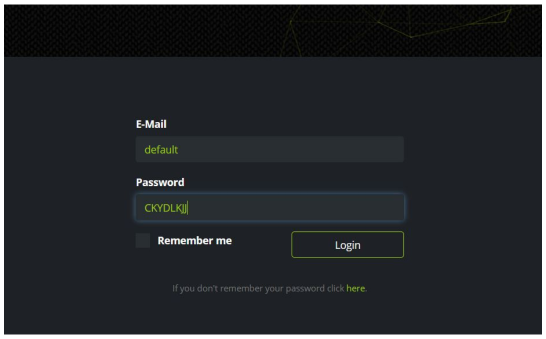
- Third, I see a ‘default’ string and then some jumbled letters



```
<div class="header-title"> <h3>Hi!</h3> <small> Feel free to hack your way in :) </small> </div> </div>
<div class="panel panel-filled"> <div class="panel-body"> <form action="https://www.hackthebox.eu/invite" id="verifyForm" method="post">
<input type="hidden" name="_token" value="gxTRXg6zcsi597LSj0PjM6XN4Tdm4lwDY6P1uqd7">
<div class="form-group ">
<label class="control-label" for="code">Invite Code</label>
<input type="text" title="Please enter your invite code" required="" value="" name="code" id="code" class="form-control">
<span class="help-block small"></span> </div>
<div> <button class="btn btn-accent">Sign Up</button> </div> </form> </div> </div>
<span class="help-block small text-center">If you are already a member click <a href="https://www.hackthebox.eu/login">here</a> to login.</span> <br>
<div class="native-ad"></div> <script> (function(){ if(typeof _bsa !== 'undefined' && _bsa) { _bsa.init('default', 'CKYDLKJJ', 'placement:hacktheboxeu', { target: '.native-ad', al
<div class="particles_full" id="particles-js"></div> </section> </div> <script src="https://www.hackthebox.eu/js/htb-frontend.min.js"></script>
<script defer src="/js/inviteapi.min.js"></script> <script defer src="https://www.hackthebox.eu/js/calm.js"></script> </body> </html>
```

- Finally, I see links to the other js files in the debugger

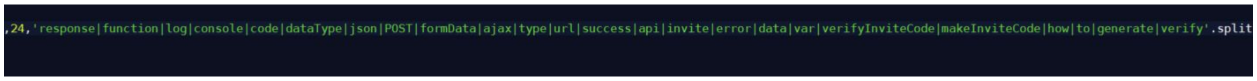
Let’s try the default and hopefully password in case the lower hanging fruit gets us anywhere.



E-Mail: default

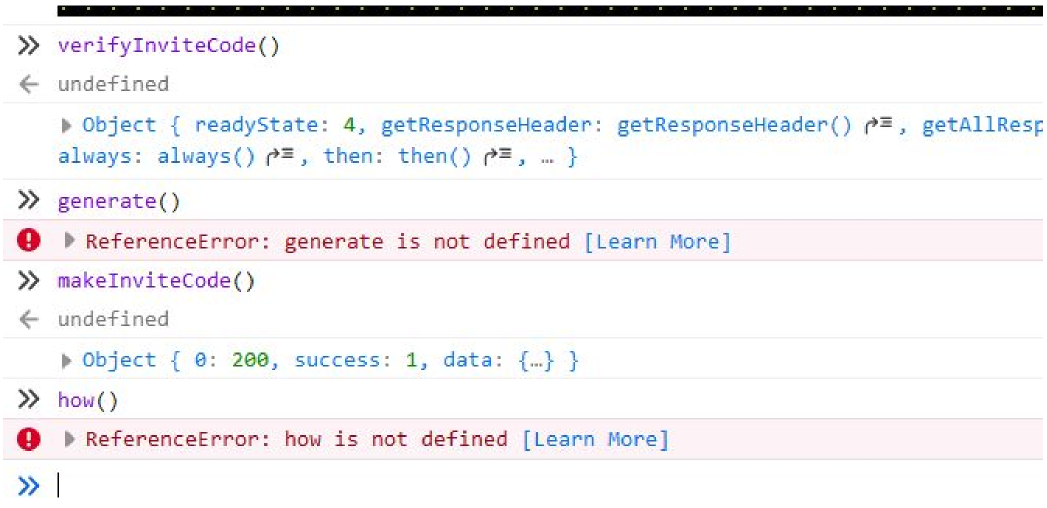
Password: CKYDLKJJ

No luck. Let’s look at the js. This section of the “inviteapi” has a bit called verifyInviteCode.



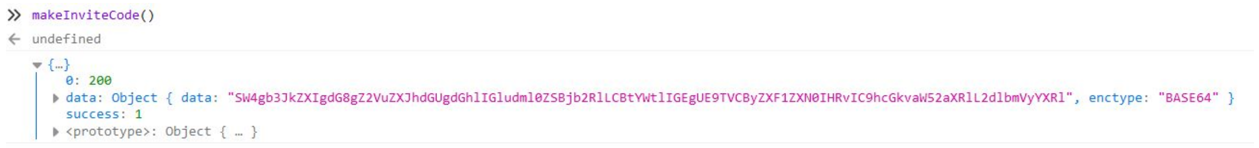
```
,24,'response|function|log|console|code|dataType|json|POST|formData|ajax|type|url|success|api|invite|error|data|var|verifyInviteCode|makeInviteCode|how|to|generate|verify'.split
```

I had to go back and refresh on how to use the debugger, but once you factor into the process that someone bothered to make a skull and crossbones on the Console tab, I think I’m headed in the right direction. So I’m going to start launching these functions:



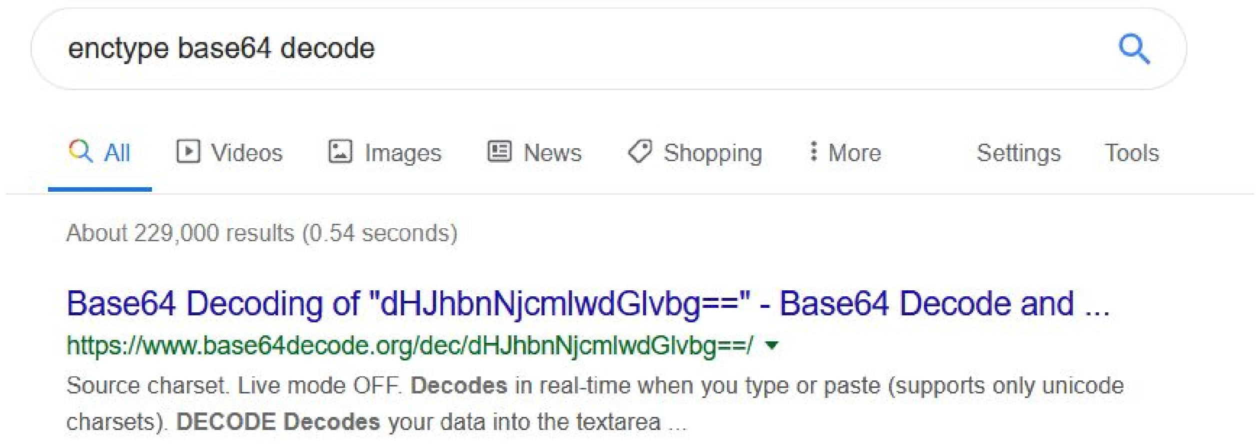
```
>> verifyInviteCode()
<- undefined
   ▶ Object { readyState: 4, getResponseHeader: getResponseHeader() ↱≡, getAllResp
   always: always() ↱≡, then: then() ↱≡, … }
>> generate()
❗ ▶ ReferenceError: generate is not defined [Learn More]
>> makeInviteCode()
<- undefined
   ▶ Object { 0: 200, success: 1, data: {…} }
>> how()
❗ ▶ ReferenceError: how is not defined [Learn More]
>> |
```

I see a very promising result under makeInviteCode():



```
>> makeInviteCode()
<- undefined
 ▼ {…}
    0: 200
  ▶ data: Object { data: "SW4gb3JkZXIgdG8gZ2VuZXJhdGUgdGhlIGludml0ZSBjb2RlLCBtYWtlIGEgUE9TVCByZXF1ZXN0IHRvIC9hcGkvaW52aXRlL2dlbmVyYXRl", enctype: "BASE64" }
    success: 1
  ▶ <prototype>: Object { … }
```

Now this might be a code. Before I just copy and paste the data string, I see enctype: BASE64. So this potential code is encrypted? To Google!



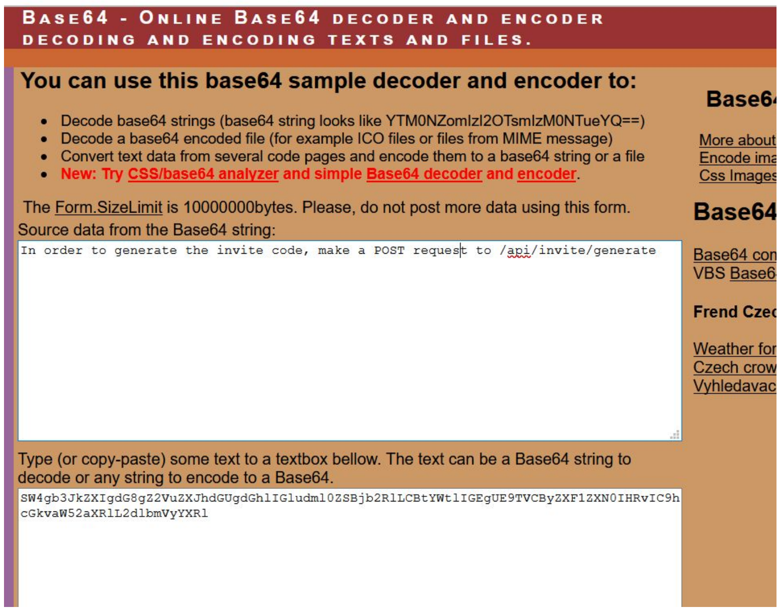
enctype base64 decode

All | Videos | Images | News | Shopping | More | Settings | Tools

About 229,000 results (0.54 seconds)

Base64 Decoding of "dHJhbnNjcmlwdGlvbg==" - Base64 Decode and ...
https://www.base64decode.org/dec/dHJhbnNjcmlwdGlvbg==/

Source charset. Live mode OFF. **Decodes** in real-time when you type or paste (supports only unicode charsets). **DECODE Decodes** your data into the textarea ...



BASE64 - ONLINE BASE64 DECODER AND ENCODER
DECODING AND ENCODING TEXTS AND FILES.

You can use this base64 sample decoder and encoder to:

- Decode base64 strings (base64 string looks like YTM0NZomIzI2OTsmIzM0NTueYQ==)
- Decode a base64 encoded file (for example ICO files or files from MIME message)
- Convert text data from several code pages and encode them to a base64 string or a file
- **New: Try CSS/base64 analyzer and simple Base64 decoder and encoder.**

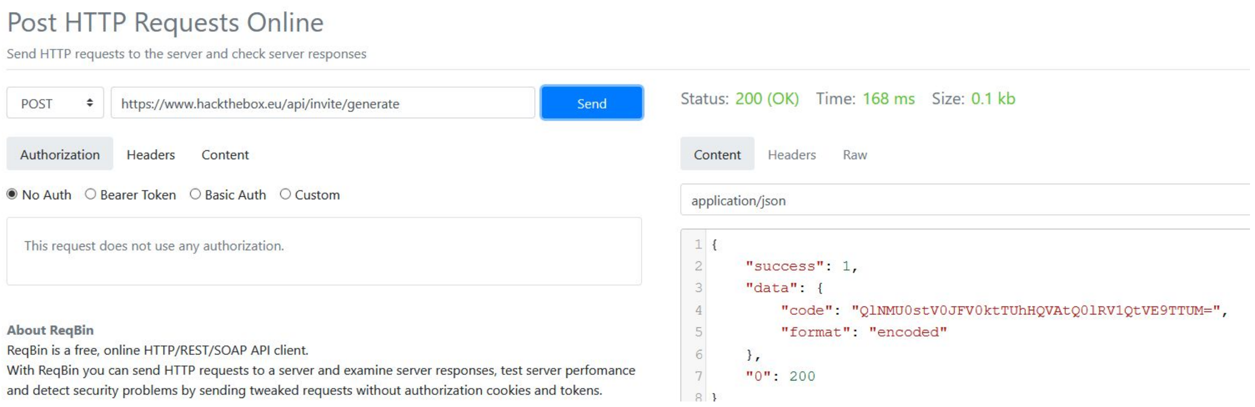
The Form.SizeLimit is 10000000bytes. Please, do not post more data using this form.
Source data from the Base64 string:

```
In order to generate the invite code, make a POST request to /api/invite/generate
```

Type (or copy-paste) some text to a textbox bellow. The text can be a Base64 string to decode or any string to encode to a Base64.

```
SW4gb3JkZXIgdG8gZ2VuZXJhdGUgdGhlIGludml0ZSBjb2RlLCBtYWtlIGEgUE9TVCByZXF1ZXN0IHRvIC9h
cGkvaW52aXRlL2dlbmVyYXRl
```

I am on the right path for sure. Rather than break out my old web dev tools, I invoked the power of Google search and found this:



Post HTTP Requests Online

Send HTTP requests to the server and check server responses

https://www.hackthebox.eu/api/invite/generate

Status: 200 (OK) Time: 168 ms Size: 0.1 kb

[✓ No Auth] Bearer Token Basic Auth Custom

This request does not use any authorization.

application/json

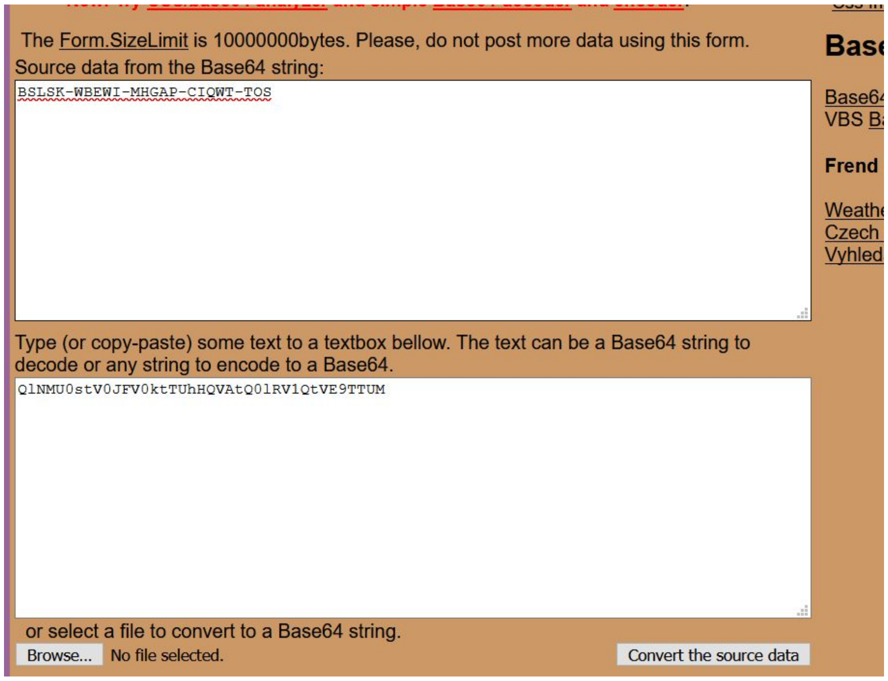
```
{
    "success": 1,
    "data": {
        "code": "Q1NMU0stV0JFV0ktTUhHQVAtQ01RV1QtVE9TTUM=",
        "format": "encoded"
    },
    "0": 200
}
```

About ReqBin

ReqBin is a free, online HTTP/REST/SOAP API client.

With ReqBin you can send HTTP requests to a server and examine server responses, test server perfomance and detect security problems by sending tweaked requests without authorization cookies and tokens.

Another encoded object. Back to the decoder…



The Form.SizeLimit is 10000000bytes. Please, do not post more data using this form.

Source data from the Base64 string:

```
BSLSK-WBEWI-MHGAP-CIQWT-TOS
```

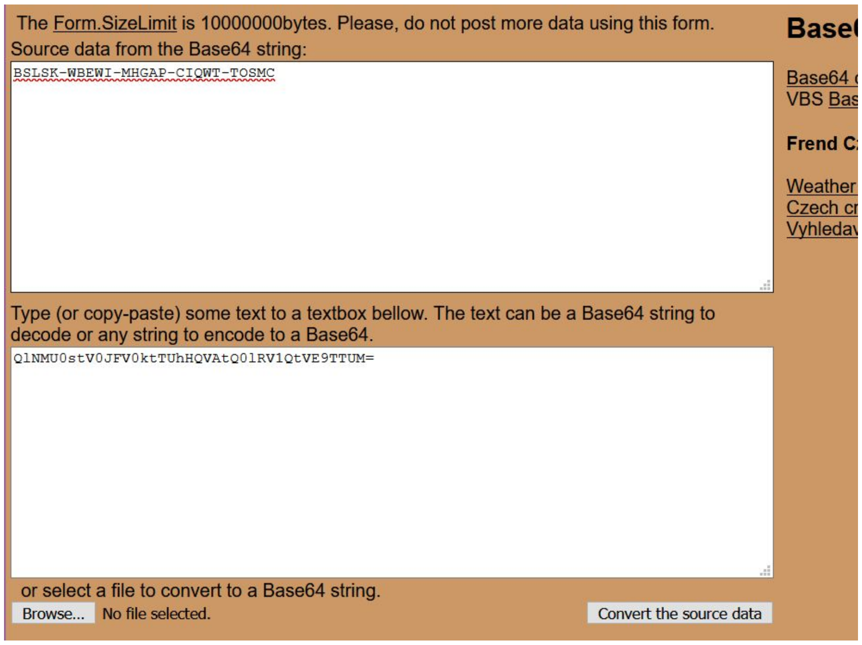
Type (or copy-paste) some text to a textbox bellow. The text can be a Base64 string to decode or any string to encode to a Base64.

```
Q1NMU0stV0JFV0ktTUhHQVAtQ01RV1QtVE9TTUM
```

or select a file to convert to a Base64 string.

No file selected.

This looks really good. Let’s try it out... no good. Closer inspection reveals I forgot the equals sign:



The Form.SizeLimit is 10000000bytes. Please, do not post more data using this form.

Source data from the Base64 string:

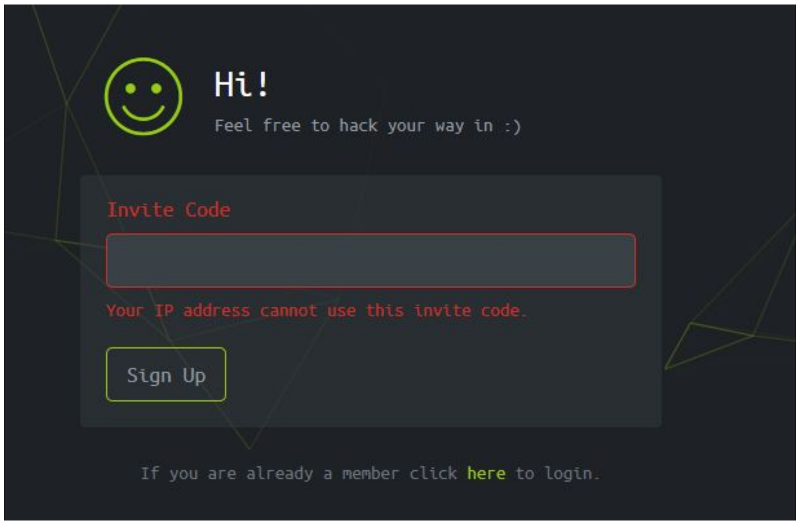
```
BSLSK-WBEWI-MHGAP-CIQWT-TOSMC
```

Type (or copy-paste) some text to a textbox bellow. The text can be a Base64 string to decode or any string to encode to a Base64.

```
QlNMU0stV0JFV0ktTUhHQVAtQ0lRV1QtVE9TTUM=
```

or select a file to convert to a Base64 string.

Interesting. I think I need to make the POST call from my own machine.



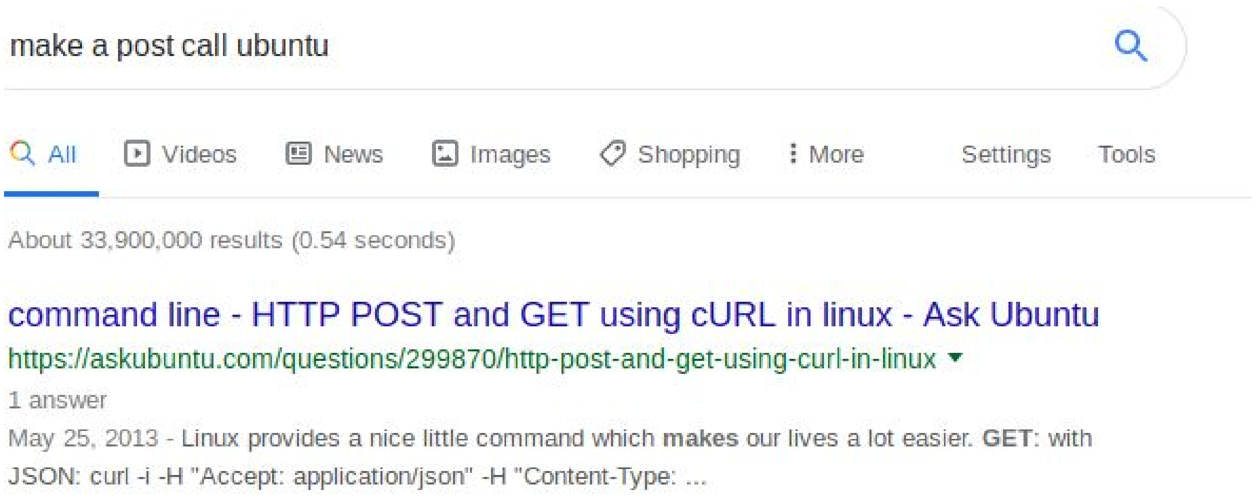
Hi!

Feel free to hack your way in :)

Invite Code

Your IP address cannot use this invite code.

If you are already a member click here to login.

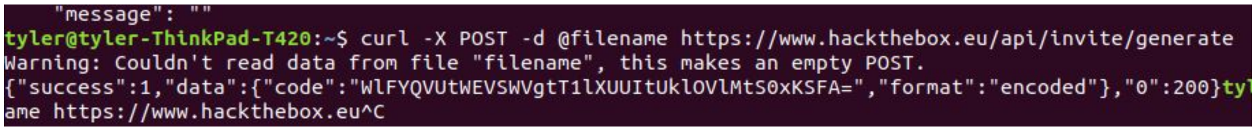


About 33,900,000 results (0.54 seconds)

command line - HTTP POST and GET using cURL in linux - Ask Ubuntu
https://askubuntu.com/questions/299870/http-post-and-get-using-curl-in-linux

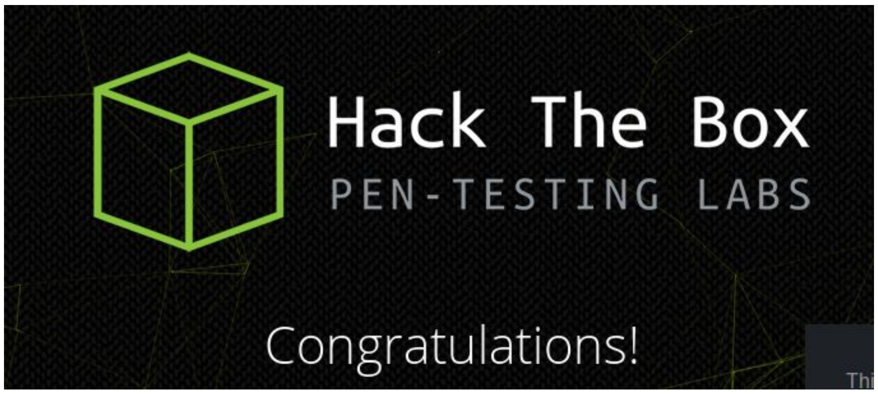
1 answer

May 25, 2013 - Linux provides a nice little command which **makes** our lives a lot easier. **GET**: with JSON: curl -i -H "Accept: application/json" -H "Content-Type: ...



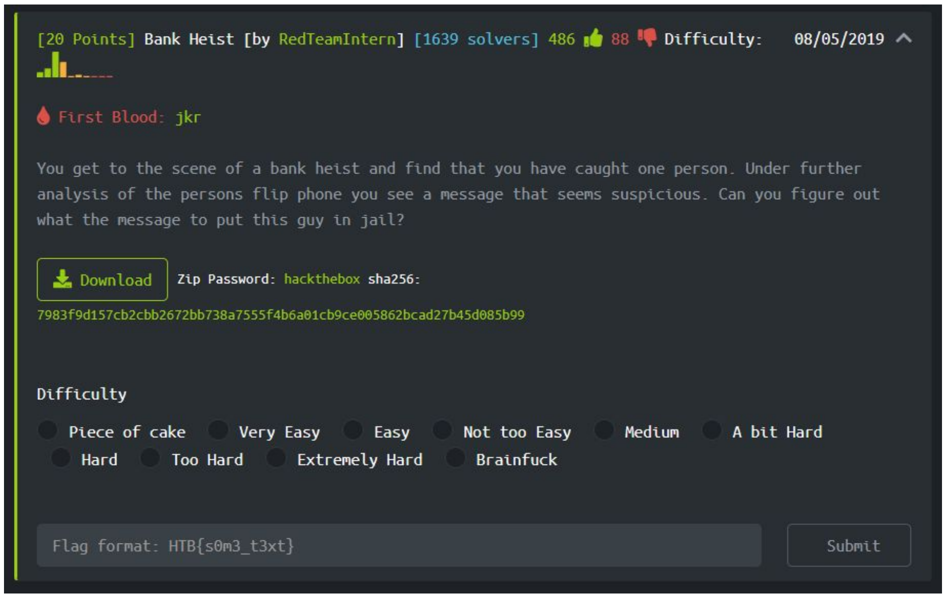
```
    "message": ""
tyler@tyler-ThinkPad-T420:~$ curl -X POST -d @filename https://www.hackthebox.eu/api/invite/generate
Warning: Couldn't read data from file "filename", this makes an empty POST.
{"success":1,"data":{"code":"WlFYQVUtWEVSWVgtT1lXUUItUklOVlMtS0xKSFA=","format":"encoded"},"0":200}ty
ame https://www.hackthebox.eu^C
```

Let’s try one more time:



Now it’s time for challenges. Let’s start with an easier one:

BANK HEIST


[20 Points] Bank Heist [by RedTeamIntern] [1639 solvers] 486 88 Difficulty: 08/05/2019

First Blood: jkr

You get to the scene of a bank heist and find that you have caught one person. Under further analysis of the persons flip phone you see a message that seems suspicious. Can you figure out what the message to put this guy in jail?

Download Zip Password: hackthebox sha256: 7983f9d157cb2cbb2672bb738a7555f4b6a01cb9ce005862bcad27b45d085b99

Difficulty

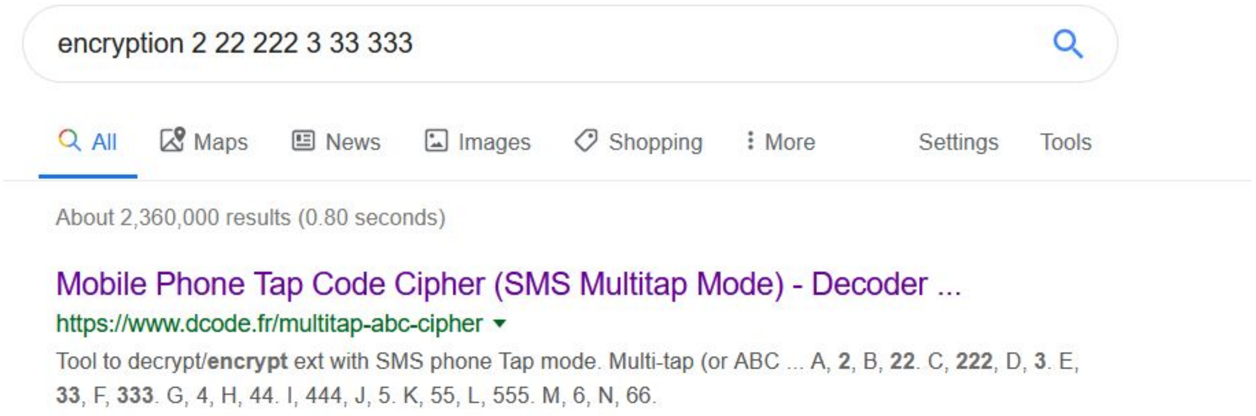
Piece of cake, Very Easy, Easy, Not too Easy, Medium, A bit Hard, Hard, Too Hard, Extremely Hard, Brainfuck

Flag format: HTB{s0m3_t3xt} Submit

The only thing I know about encryption is from my Abstract Algebra class in college. So first step is to start googling on decryption methods. This returned an overwhelming number of results. I eventually asked someone where I could learn more about encryption methods. I was pointed to this site:

https://www.dcode.fr/en

Then I started doing searches on segments of the code itself:

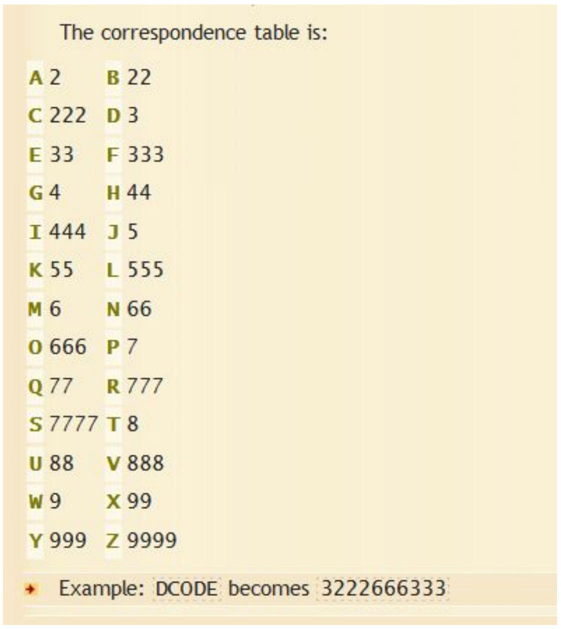

encryption 2 22 222 3 33 333

All Maps News Images Shopping More Settings Tools

About 2,360,000 results (0.80 seconds)

Mobile Phone Tap Code Cipher (SMS Multitap Mode) - Decoder ...
https://www.dcode.fr/multitap-abc-cipher
Tool to decrypt/**encrypt** ext with SMS phone Tap mode. Multi-tap (or ABC ... A, **2**, B, **22**. C, **222**, D, **3**. E, **33**, F, **333**. G, 4, H, 44. I, 444, J, 5. K, 55, L, 555. M, 6, N, 66.

I got back to the same site, but this time with a decoder and a table:



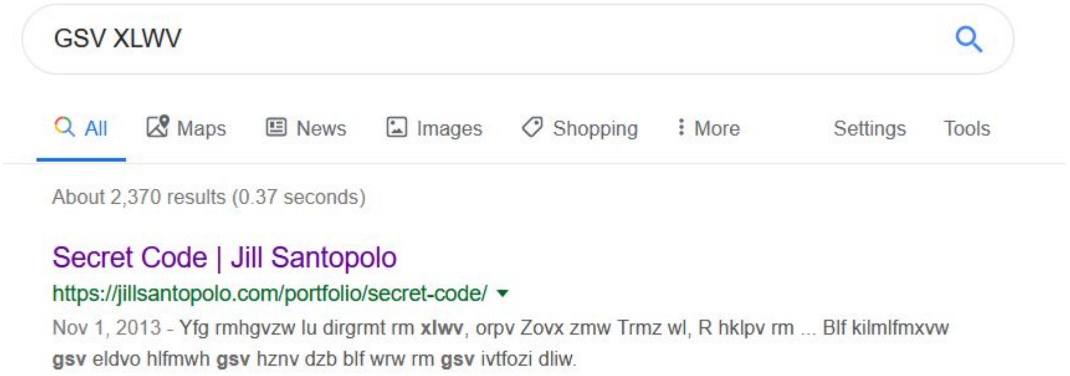
The correspondence table is:

A	2	B	22
C	222	D	3
E	33	F	333
G	4	H	44
I	444	J	5
K	55	L	555
M	6	N	66
O	666	P	7
Q	77	R	777
S	7777	T	8
U	88	V	888
W	9	X	99
Y	999	Z	9999

Example: DCODE becomes 3222666333

After some work, we have:
"IF YOU ARE READING THE CIPHER YOU ARE OKAY. YOUR SHARE OF THE HEIST IS IN YOUR HOUSE THE KEY TO THE LOCK IS BELOW GO TO PARIS : GSV XLWV GL GSV HZU OLXP TLIVGRIVNVMGUFMW!!"

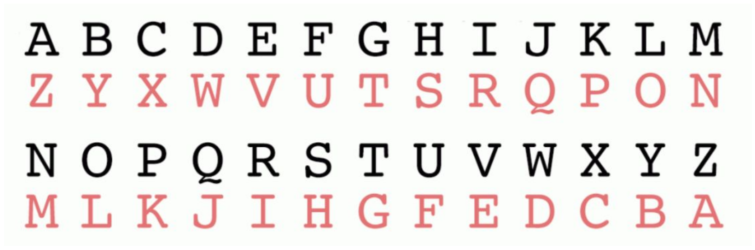
Submitting this didn't work. Some googling on the formatting led me to realize that the nonsensical part isn't a password but more code. I'll google part of it:



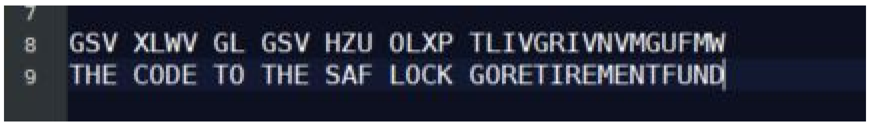
GSV XLWV

About 2,370 results (0.37 seconds)

Secret Code | Jill Santopolo
https://jillsantopolo.com/portfolio/secret-code/
Nov 1, 2013 - Yfg rmhgvzw lu dirgrmt rm **xlwv**, orpv Zovx zmw Trmz wl, R hklpv rm ... Blf kilmlfmxvw **gsv** eldvo hlfmwh **gsv** hznv dzb blf wrw rm **gsv** ivtfozi dliw.



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

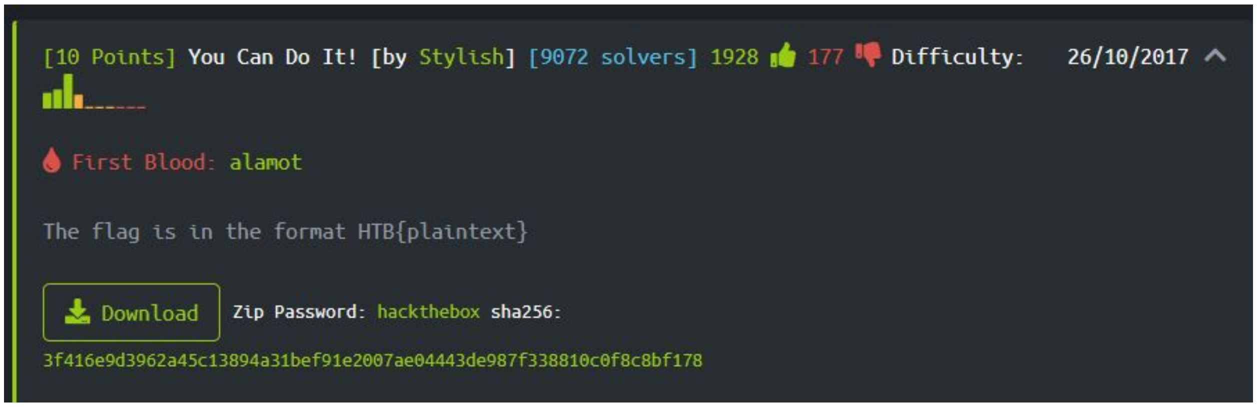


```
GSV XLWV GL GSV HZU OLXP TLIVGRIVNVMGUFMW
THE CODE TO THE SAF LOCK GORETIREMENTFUND
```

The correct submission format was HTB{GORETIREMENTFUND!!}. This took a lot of trial and error.

Let’s do one more easy one:

YOU CAN DO IT


[10 Points] You Can Do It! [by Stylish] [9072 solvers] 1928 177 Difficulty: 26/10/2017

First Blood: alamot

The flag is in the format HTB{plaintext}

Download Zip Password: hackthebox sha256:
3f416e9d3962a45c13894a31bef91e2007ae04443de987f338810c0f8c8bf178

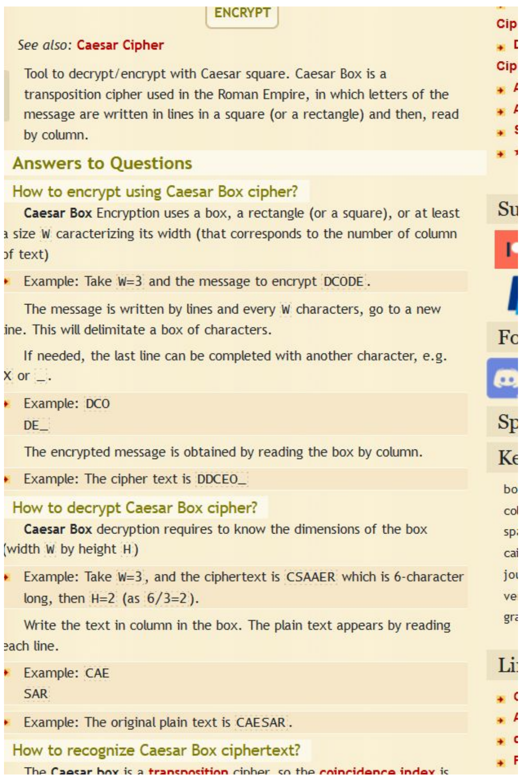
Opening the file gave us the following code: YHAOANUTDSYOEOIEUTTC!

Is this a word scramble? I can make out some words: YOU, DONUT, TO, HAD, CAN

The title is “you can do it”, so I started crossing out letters on a sheet of paper and saw this:

~~Y~~HA~~O~~AN~~U~~TDSYOEOIEUTTC! -- every third letter spells “you”

From here I just spent a lot of time googling until I found this:


ENCRYPT

See also: Caesar Cipher

Tool to decrypt/encrypt with Caesar square. Caesar Box is a transposition cipher used in the Roman Empire, in which letters of the message are written in lines in a square (or a rectangle) and then, read by column.

Answers to Questions

How to encrypt using Caesar Box cipher?

Caesar Box Encryption uses a box, a rectangle (or a square), or at least a size W caracterizing its width (that corresponds to the number of column of text)

- Example: Take W=3 and the message to encrypt DCODE.

The message is written by lines and every W characters, go to a new line. This will delimitate a box of characters.

If needed, the last line can be completed with another character, e.g. X or _.

- Example: DCO
  DE_

The encrypted message is obtained by reading the box by column.

- Example: The cipher text is DDCEO_

How to decrypt Caesar Box cipher?

Caesar Box decryption requires to know the dimensions of the box (width W by height H)

- Example: Take W=3, and the ciphertext is CSAAER which is 6-character long, then H=2 (as 6/3=2).

Write the text in column in the box. The plain text appears by reading each line.

- Example: CAE
  SAR

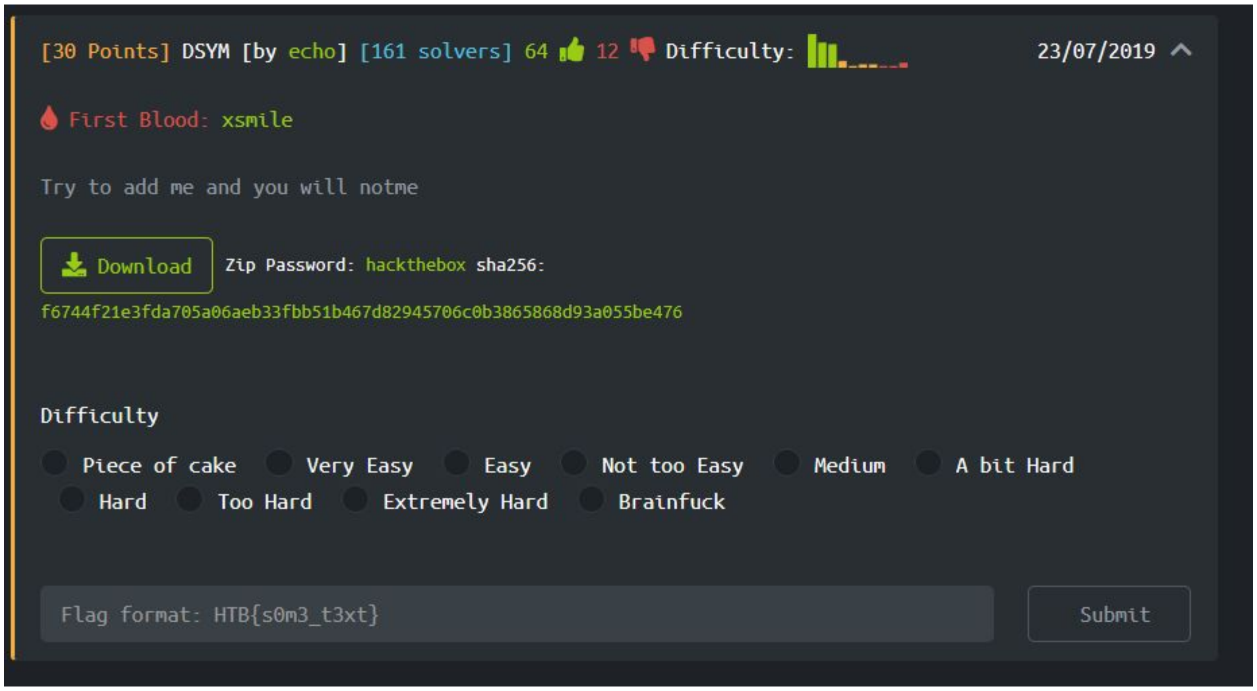
- Example: The original plain text is CAESAR.

How to recognize Caesar Box ciphertext?

The **Caesar box** is a transposition cipher, so the coincidence index is

Continuing to fiddle with this gave the code: YOU SEE THAT YOU CAN DO IT. Didn’t even need a computer…

DSYM



[30 Points] DSYM [by echo] [161 solvers] 64 12 Difficulty: 23/07/2019

First Blood: xsmile

Try to add me and you will notme

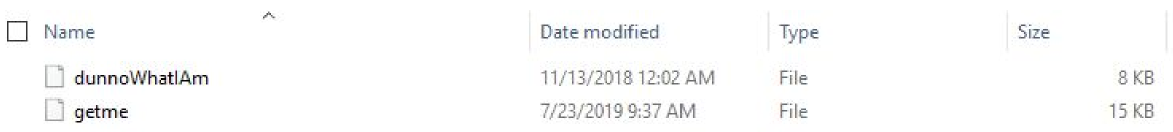
Download Zip Password: hackthebox sha256: f6744f21e3fda705a06aeb33fbb51b467d82945706c0b3865868d93a055be476

Difficulty

Piece of cake · Very Easy · Easy · Not too Easy · Medium · A bit Hard · Hard · Too Hard · Extremely Hard · Brainfuck

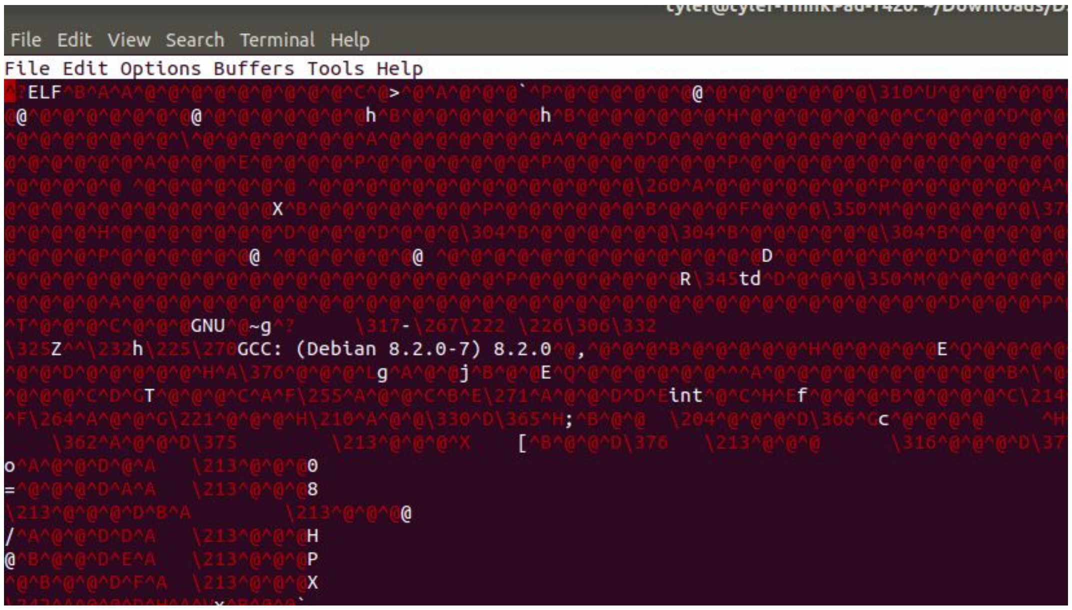
Flag format: HTB{s0m3_t3xt} · Submit

Opening this folder reveals two files:



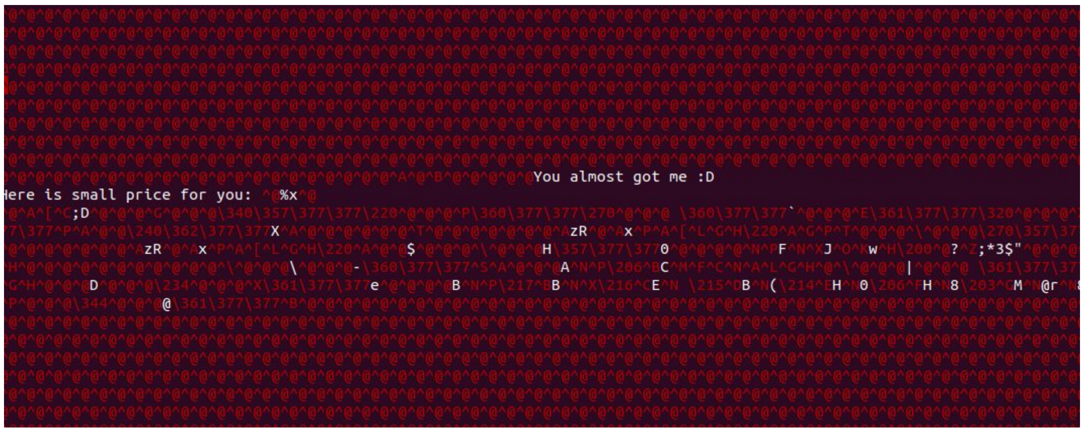
Name	Date modified	Type	Size
dunnoWhatIAm	11/13/2018 12:02 AM	File	8 KB
getme	7/23/2019 9:37 AM	File	15 KB

Let’s take a look:

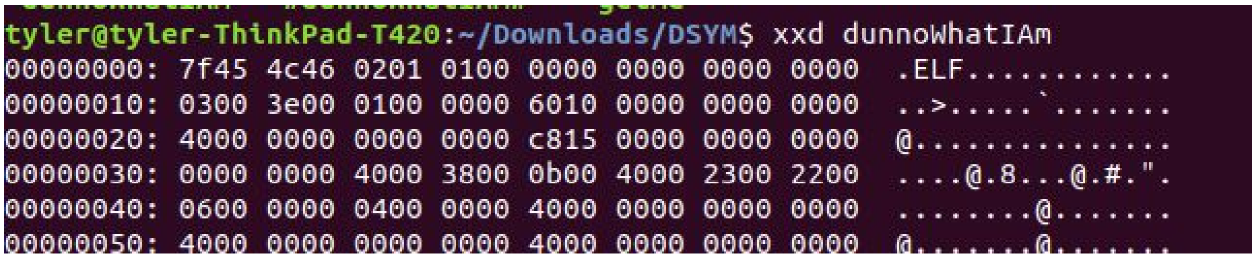


File Edit View Search Terminal Help
File Edit Options Buffers Tools Help

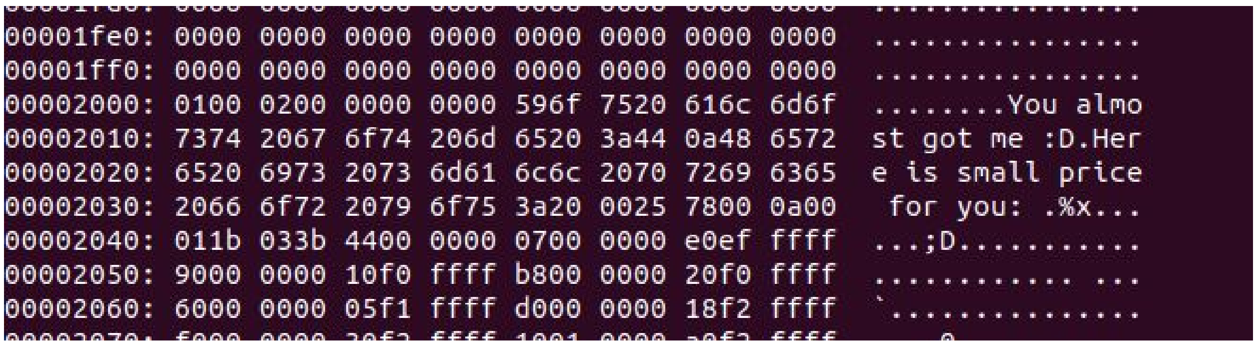
```
^?ELF^B^A^A^@^@^@^@^@^@^@^@^@^C^@>^@^A^@^@^@`^P^@^@^@^@^@^@@^@^@^@^@^@^@^@\310^U^@^@^@^@^@^
@@^@^@^@^@^@^@^@@^@^@^@^@^@^@^@h^B^@^@^@^@^@^@h^B^@^@^@^@^@^@^H^@^@^@^@^@^@^@^C^@^@^@^D^@^@^
^@^@^@^@^@^@^@\^@^@^@^@^@^@^@^A^@^@^@^@^@^@^A^@^@^D^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^
@^@^@^@^@^A^@^@^@^E^@^@^@^@^P^@^@^@^@^@^@^P^@^@^@^@^@^@^P^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@ ^@^@^@^@^@^@^@ ^@^@^@^@^@^@^@^@^@^@^@^@^@^@\260^A^@^@^@^@^@^@^P^@^@^@^@^@^A^
@^@^@^@^@^@^@^@^@^@^@^@X^B^@^@^@^@^@^@^P^@^@^@^@^@^@^B^@^@^@^F^@^@^@\350^M^@^@^@^@^@^@\37
@^@^@^H^@^@^@^@^@^@^@^D^@^@^@^D^@^@^@\304^B^@^@^@^@^@^@\304^B^@^@^@^@^@^@\304^B^@^@^@^@^
@^@^@^@^P^@^@^@^@^@^@@ ^@^@^@^@^@^@@ ^@^@^@^@^@^@^@^@^@^@^@^@^@D^@^@^@^@^@^@^D^@^@^@^@^
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^P^@^@^@^@^@^@R\345td^D^@^@^@\350^M^@^@^@^@^@^@
^@^@^@^@^A^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^D^@^@^@^P^
^T^@^@^@^C^@^@^@GNU^@~g^?      \317-\267\222 \226\306\332
\325Z^^\232h\225\270GCC: (Debian 8.2.0-7) 8.2.0^@,^@^@^@^B^@^@^@^@^@^H^@^@^@^@^@E^Q^@^@^@^@
^@^@^D^@^@^@^@^@^H^A\376^@^@^@^Lg^A^@^@j^B^@^@E^Q^@^@^@^@^@^@^^^A^@^@^@^@^@^@^@^@^@^B^\^@
^@^@^@^C^D^GT^@^@^@^C^A^F\255^A^@^@^C^B^E\271^A^@^@^D^D^Eint^@^C^H^Ef^@^@^@^B^@^@^@^@^C\214
^F\264^A^@^@^G\221^@^@^@^H\210^A^@^@\330^D\365^H;^B^@^@  \204^@^@^@^D\366^Gc^@^@^@^@      ^H
    \362^A^@^@^D\375          \213^@^@^@^X     [^B^@^@^D\376    \213^@^@^@          \316^@^@^@^D\37
o^A^@^@^D^@^A   \213^@^@^@0
=^@^@^@^D^A^A   \213^@^@^@8
\213^@^@^@^D^B^A          \213^@^@^@@
/^A^@^@^D^D^A   \213^@^@^@H
@^B^@^@^D^E^A   \213^@^@^@P
^@^B^@^@^D^F^A  \213^@^@^@X
```



```
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^A^@^B^@^@^@^@You almost got me :D
Here is small price for you: ^@%x^@
^@^A^[^C;D^@^@^@^G^@^@^@\340\357\377\377\220^@^@^@^P\360\377\377\270^@^@^@ \360\377\377`^@^@^@^E\361\377\377\320^@^@^@^
7\377^P^A^@^@\240\362\377\377X^A^@^@^@^@^@^@^T^@^@^@^@^@^@^AzR^@^Ax^P^A^[^L^G^H\220^A^G^P^T^@^@^@^\^@^@^@\270\357\37
^@^@^@^@^@^@^AzR^@^Ax^P^A^[^L^G^H\220^A^@^@$^@^@^@^\^@^@^@H\357\377\3770^@^@^@^@^N^PF^N^XJ^O^Kw^H\200^@?^Z;*3$"^@^@^@
^H^@^@^@^@^@^@^@^@^@^\^@^@^@\^@^@^@-\360\377\377^S^A^@^@^@A^N^P\206^BC^M^F^C^N^A^L^G^H^@^\^@^@^@|^@^@^@ \361\377\37
^G^H^@^@^@D^@^@^@\234^@^@^@^X\361\377\377e^@^@^@^@B^N^P\217^BB^N^X\216^CE^N \215^DB^N(\214^EH^N0\206^FH^N8\203^GM^N@r^N8
^P^@^@^@\344^@^@^@@\361\377\377^B^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@^@
```

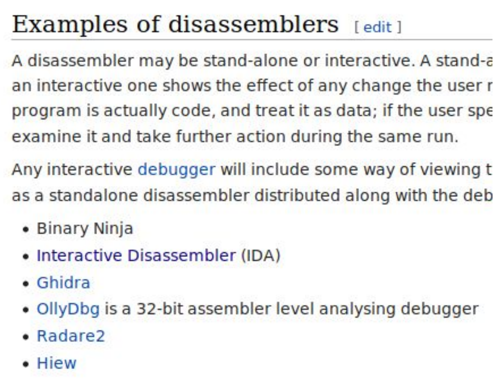


```
tyler@tyler-ThinkPad-T420:~/Downloads/DSYM$ xxd dunnoWhatIAm
00000000: 7f45 4c46 0201 0100 0000 0000 0000 0000  .ELF............
00000010: 0300 3e00 0100 0000 6010 0000 0000 0000  ..>.....`.......
00000020: 4000 0000 0000 0000 c815 0000 0000 0000  @...............
00000030: 0000 0000 4000 3800 0b00 4000 2300 2200  ....@.8...@.#.".
00000040: 0600 0000 0400 0000 4000 0000 0000 0000  ........@.......
00000050: 4000 0000 0000 0000 4000 0000 0000 0000  @.......@.......
```



```
00001fe0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00001ff0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00002000: 0100 0200 0000 0000 596f 7520 616c 6d6f  ........You almo
00002010: 7374 2067 6f74 206d 6520 3a44 0a48 6572  st got me :D.Her
00002020: 6520 6973 2073 6d61 6c6c 2070 7269 6365  e is small price
00002030: 2066 6f72 2079 6f75 3a20 0025 7800 0a00   for you: .%x...
00002040: 011b 033b 4400 0000 0700 0000 e0ef ffff  ...;D...........
00002050: 9000 0000 10f0 ffff b800 0000 20f0 ffff  ............ ...
00002060: 6000 0000 05f1 ffff d000 0000 18f2 ffff  `...............
```

So these are not immediately readable, let’s do some googling. ELF files are executables that run on linux systems. I’d love to be able to turn this machine code back into a readable format.



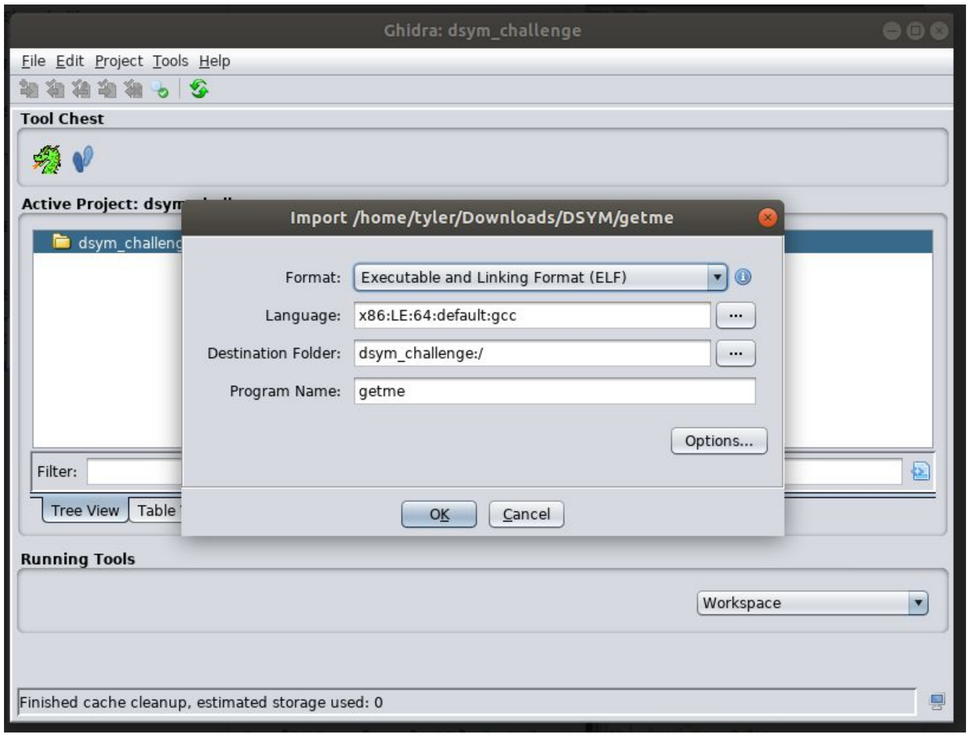
Examples of disassemblers [edit]

A disassembler may be stand-alone or interactive. A stand-a
an interactive one shows the effect of any change the user r
program is actually code, and treat it as data; if the user spe
examine it and take further action during the same run.

Any interactive debugger will include some way of viewing t
as a standalone disassembler distributed along with the deb

- Binary Ninja
- Interactive Disassembler (IDA)
- Ghidra
- OllyDbg is a 32-bit assembler level analysing debugger
- Radare2
- Hiew

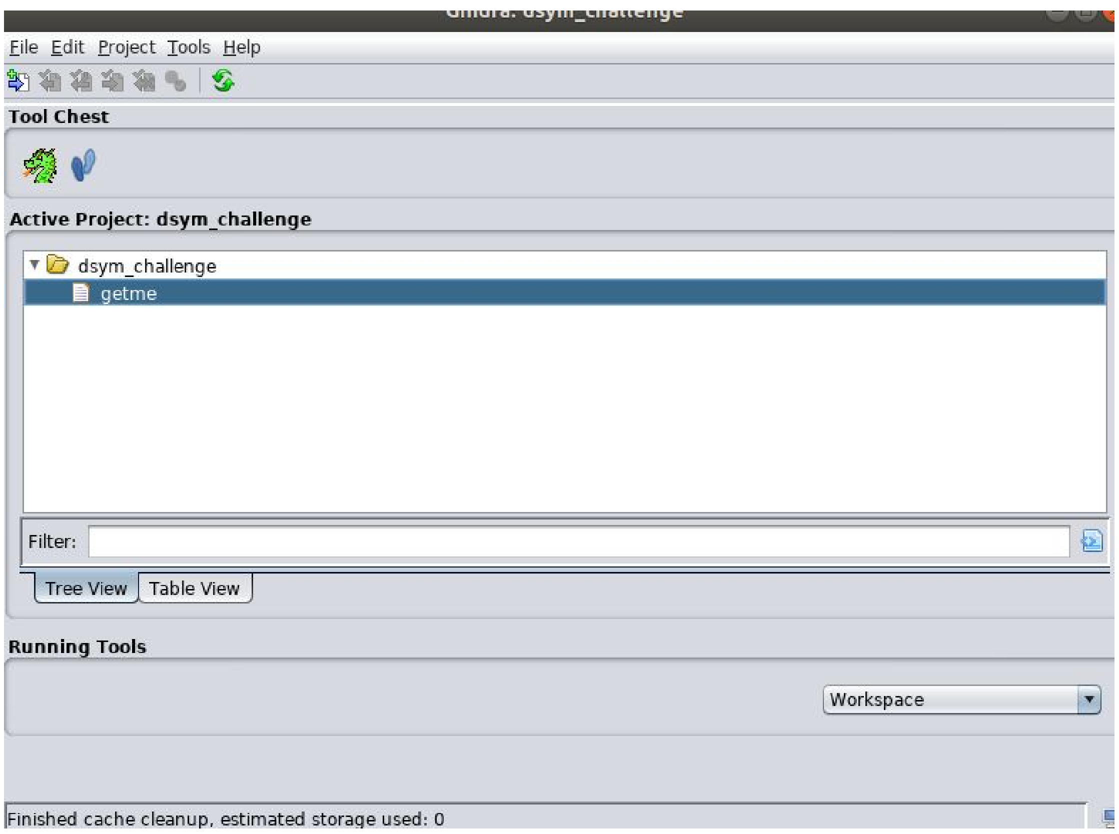
So I’m going to look around at these programs. Eventually the easiest one to install going down the list was ghidra. Let’s start with ‘getme’:



Import /home/tyler/Downloads/DSYM/getme

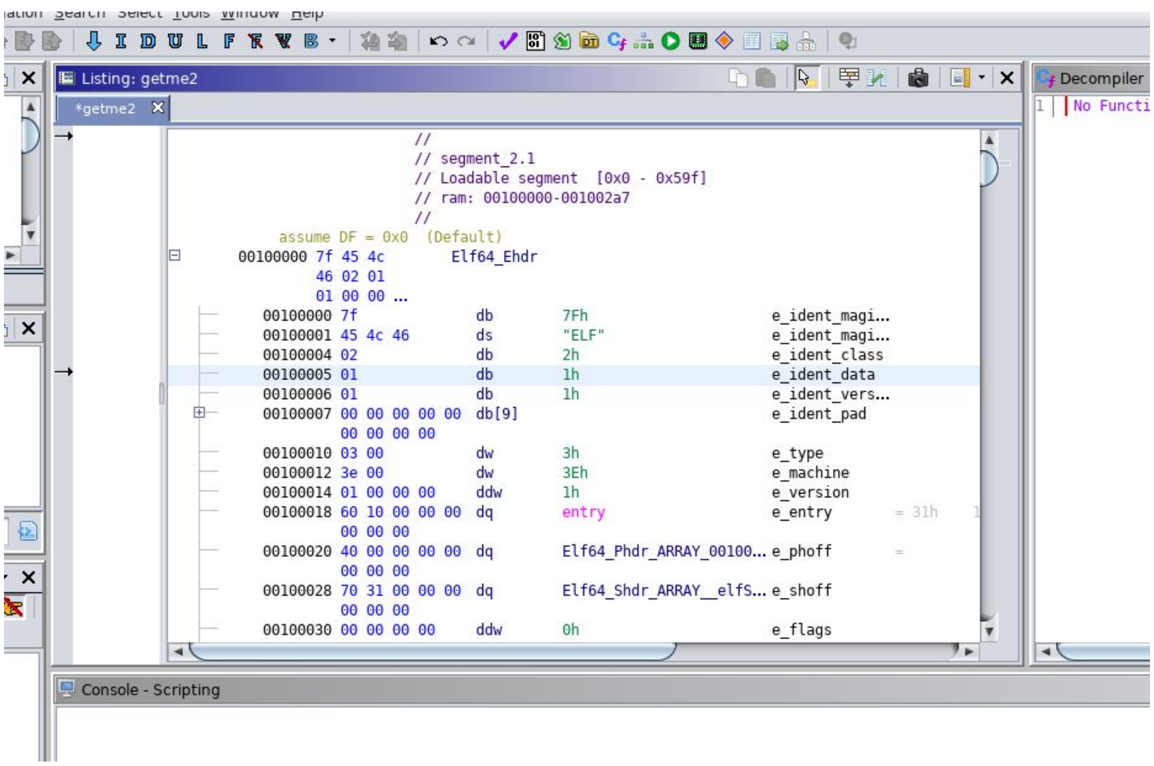
Format: Executable and Linking Format (ELF)
Language: x86:LE:64:default:gcc
Destination Folder: dsym_challenge:/
Program Name: getme

Finished cache cleanup, estimated storage used: 0



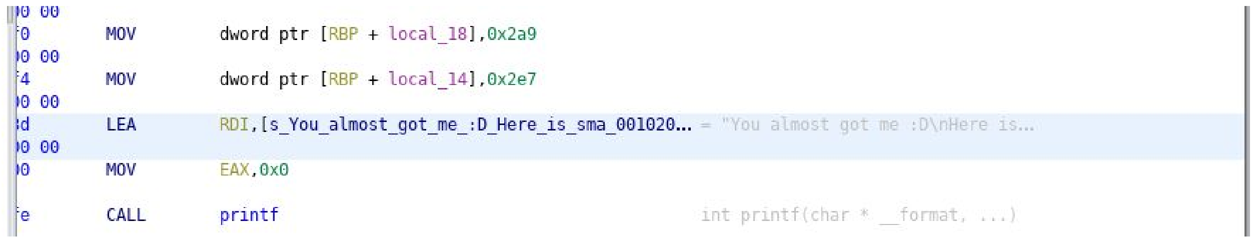
Finished cache cleanup, estimated storage used: 0

This isn't the most intuitive interface, but after awhile I clicked the dragon icon:



```
                        //
                        // segment_2.1
                        // Loadable segment  [0x0 - 0x59f]
                        // ram: 00100000-001002a7
                        //
        assume DF = 0x0  (Default)
    00100000 7f 45 4c        Elf64_Ehdr
             46 02 01
             01 00 00 ...
      00100000 7f              db        7Fh                     e_ident_magi...
      00100001 45 4c 46        ds        "ELF"                   e_ident_magi...
      00100004 02              db        2h                      e_ident_class
      00100005 01              db        1h                      e_ident_data
      00100006 01              db        1h                      e_ident_vers...
      00100007 00 00 00 00 00  db[9]                             e_ident_pad
               00 00 00 00
      00100010 03 00           dw        3h                      e_type
      00100012 3e 00           dw        3Eh                     e_machine
      00100014 01 00 00 00     ddw       1h                      e_version
      00100018 60 10 00 00 00  dq        entry                   e_entry      = 31h
               00 00 00
      00100020 40 00 00 00 00  dq        Elf64_Phdr_ARRAY_00100... e_phoff    =
               00 00 00
      00100028 70 31 00 00 00  dq        Elf64_Shdr_ARRAY__elfS... e_shoff
               00 00 00
      00100030 00 00 00 00     ddw       0h                      e_flags
```

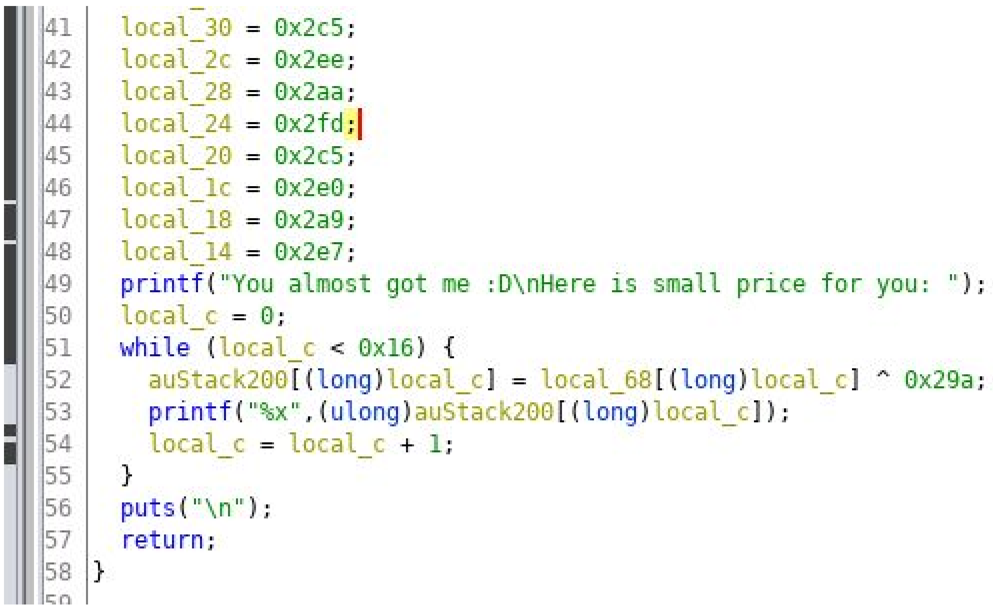
This is similar to what I got by fooling around in the terminal, but with a very important addition:



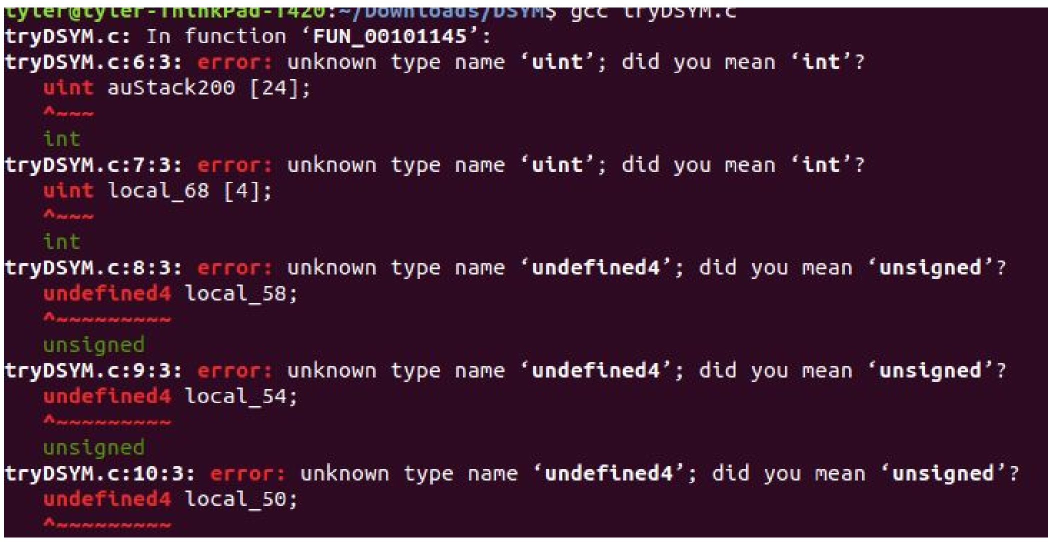
```
0 00
0        MOV        dword ptr [RBP + local_18],0x2a9
0 00
4        MOV        dword ptr [RBP + local_14],0x2e7
0 00
d        LEA        RDI,[s_You_almost_got_me_:D_Here_is_sma_001020… = "You almost got me :D\nHere is…
0 00
0        MOV        EAX,0x0

e        CALL       printf                            int printf(char * __format, ...)
```

The code surrounding the suspicious string we saw in the script when decompiled looks like C code. Let’s run it!

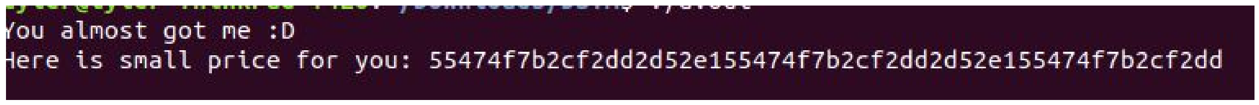


```
  local_30 = 0x2c5;
  local_2c = 0x2ee;
  local_28 = 0x2aa;
  local_24 = 0x2fd;
  local_20 = 0x2c5;
  local_1c = 0x2e0;
  local_18 = 0x2a9;
  local_14 = 0x2e7;
  printf("You almost got me :D\nHere is small price for you: ");
  local_c = 0;
  while (local_c < 0x16) {
    auStack200[(long)local_c] = local_68[(long)local_c] ^ 0x29a;
    printf("%x",(ulong)auStack200[(long)local_c]);
    local_c = local_c + 1;
  }
  puts("\n");
  return;
}
```



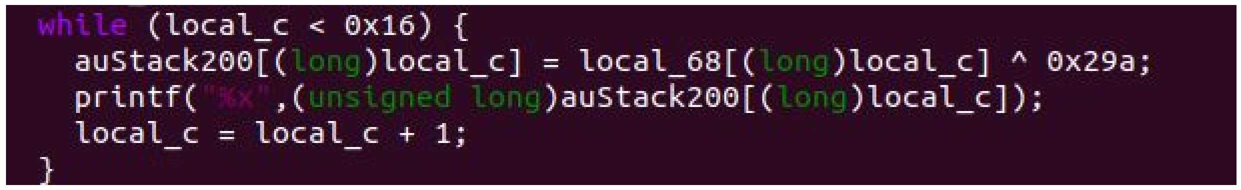
```
tyler@tyler-ThinkPad-T420:~/Downloads/DSYM$ gcc tryDSYM.c
tryDSYM.c: In function ‘FUN_00101145’:
tryDSYM.c:6:3: error: unknown type name ‘uint’; did you mean ‘int’?
   uint auStack200 [24];
   ^~~~
   int
tryDSYM.c:7:3: error: unknown type name ‘uint’; did you mean ‘int’?
   uint local_68 [4];
   ^~~~
   int
tryDSYM.c:8:3: error: unknown type name ‘undefined4’; did you mean ‘unsigned’?
   undefined4 local_58;
   ^~~~~~~~~~
   unsigned
tryDSYM.c:9:3: error: unknown type name ‘undefined4’; did you mean ‘unsigned’?
   undefined4 local_54;
   ^~~~~~~~~~
   unsigned
tryDSYM.c:10:3: error: unknown type name ‘undefined4’; did you mean ‘unsigned’?
   undefined4 local_50;
   ^~~~~~~~~~
```

Not good enough, but this can't be a dead end, so let's try to fix the code. I'll change uint to unsigned int and undefined4 to unsigned:



```
You almost got me :D
Here is small price for you: 55474f7b2cf2dd2d52e155474f7b2cf2dd2d52e155474f7b2cf2dd
```

This didn't work as the submission. It's hexadecimal? No matter how I try to convert this hex string to decimal or text, it doesn't make a lot of sense. Does the code require more attention?

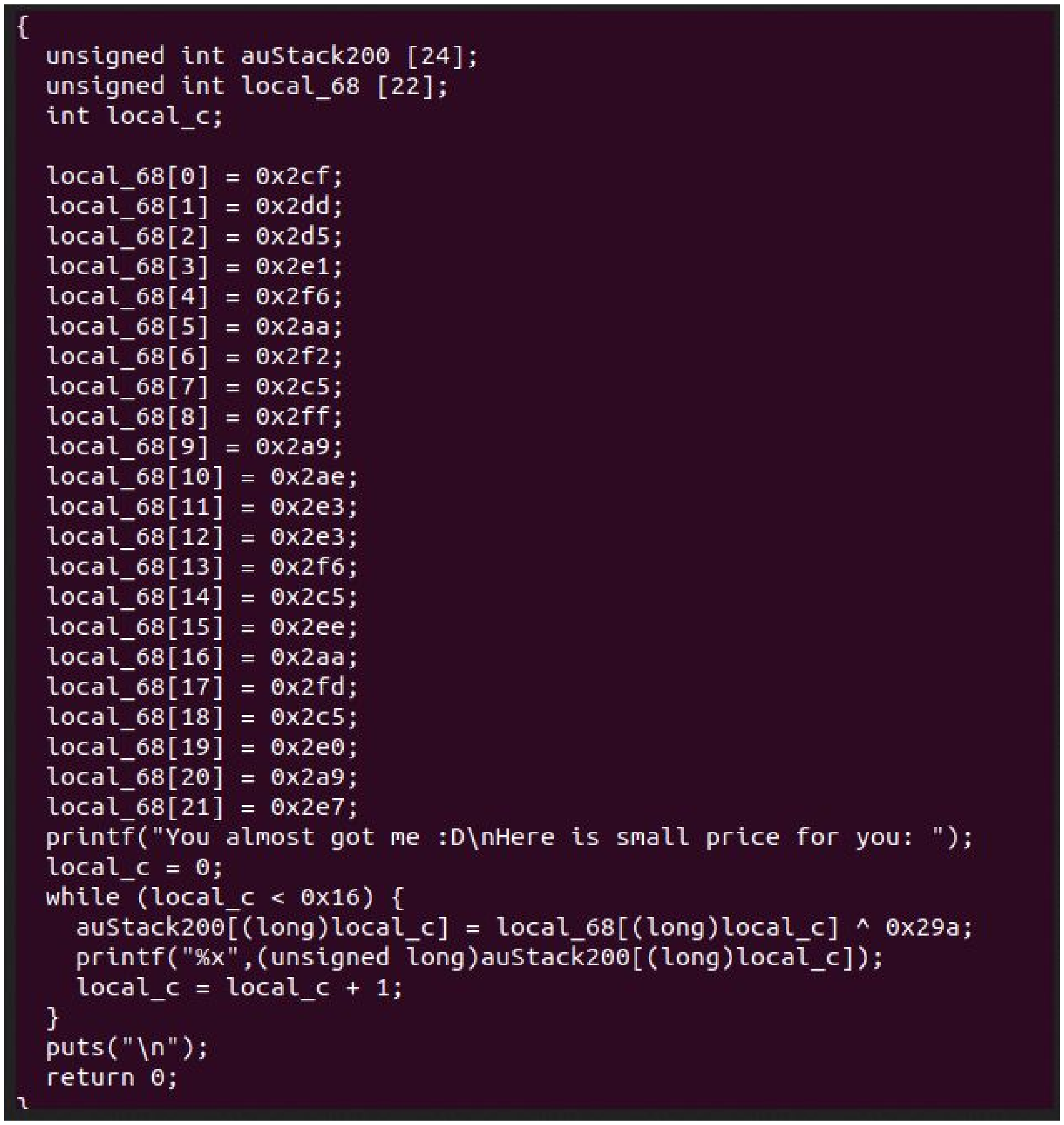


```
  while (local_c < 0x16) {
    auStack200[(long)local_c] = local_68[(long)local_c] ^ 0x29a;
    printf("%x",(unsigned long)auStack200[(long)local_c]);
    local_c = local_c + 1;
  }
```

Staring at the code, I see:

- local_c is an iterator
- the while loop cycles 22 times
- local_c is incremented at the end of each loop
- each loop uses the local_68[] array to modify the auStack200[] array
- the auStack200[] array has 24 elements
- local_68[] has 4 elements

4 elements isn't enough for this loop. But if I count up the local_XX variables, there are 22 total. Let's modify the code again:



```
{
  unsigned int auStack200 [24];
  unsigned int local_68 [22];
  int local_c;

  local_68[0] = 0x2cf;
  local_68[1] = 0x2dd;
  local_68[2] = 0x2d5;
  local_68[3] = 0x2e1;
  local_68[4] = 0x2f6;
  local_68[5] = 0x2aa;
  local_68[6] = 0x2f2;
  local_68[7] = 0x2c5;
  local_68[8] = 0x2ff;
  local_68[9] = 0x2a9;
  local_68[10] = 0x2ae;
  local_68[11] = 0x2e3;
  local_68[12] = 0x2e3;
  local_68[13] = 0x2f6;
  local_68[14] = 0x2c5;
  local_68[15] = 0x2ee;
  local_68[16] = 0x2aa;
  local_68[17] = 0x2fd;
  local_68[18] = 0x2c5;
  local_68[19] = 0x2e0;
  local_68[20] = 0x2a9;
  local_68[21] = 0x2e7;
  printf("You almost got me :D\nHere is small price for you: ");
  local_c = 0;
  while (local_c < 0x16) {
    auStack200[(long)local_c] = local_68[(long)local_c] ^ 0x29a;
    printf("%x",(unsigned long)auStack200[(long)local_c]);
    local_c = local_c + 1;
  }
  puts("\n");
  return 0;
}
```

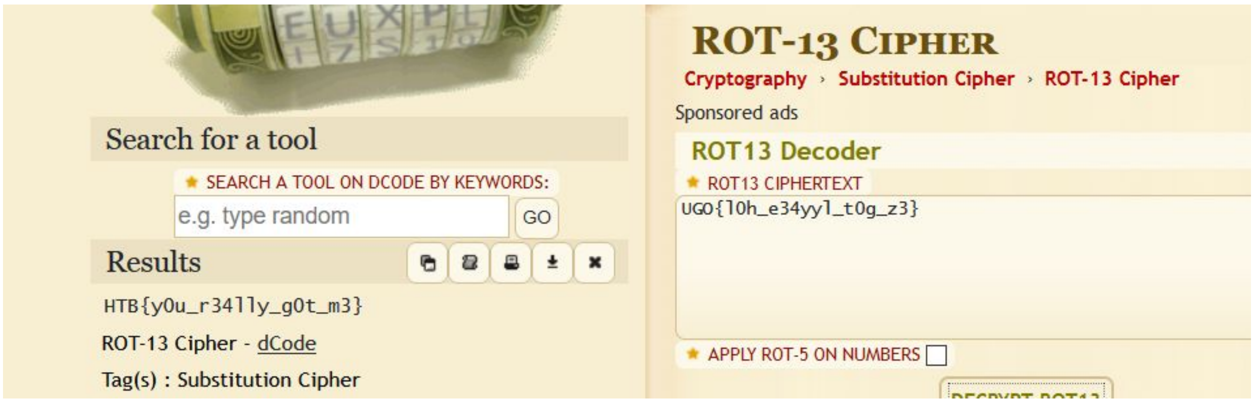
Now the output is slightly different:

You almost got me :D
Here is small price for you: 55474f7b6c30685f65333479796c5f7430675f7a337d

HTB{55474f7b6c30685f65333479796c5f7430675f7a337d} doesn't work.

Using an online converter, I translate this string to UGO{l0h_e34yyl_t0g_z3}.

This looks really similar to the HTB{xxxxx} format. Playing around with it and looking for common ciphers, I notice that U is 13 chars in the alphabet from H, G is 13 chars from T, and O is 13 chars from B. Googling this, I came across a cipher called ROT13. Using an online converter:



```
UGO{l0h_e34yyl_t0g_z3}
```

```
HTB{y0u_r34lly_g0t_m3}
```

ROT-13 Cipher - dCode

Tag(s) : Substitution Cipher

And it’s good!