

Krypton

Level 0 – 1

Step 1: Accessing the Krypton Wargame Portal

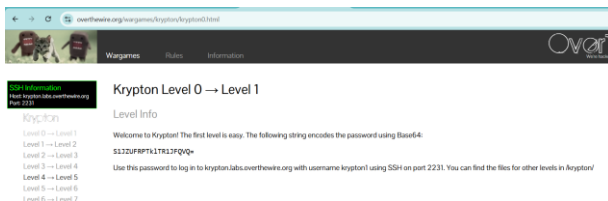
- **Tool Used:** Web Browser
- **Action Taken:**
Opened the URL: <https://overthewire.org/wargames/krypton/>

Explanation:

- This URL points to the official OverTheWire Krypton wargame portal.
- It provides:
 - Introduction to the Krypton series.
 - Instructions on how to start Level 0.
 - SSH login details including username and port.

Purpose:

- To understand the challenge and prepare for logging into Level 0.



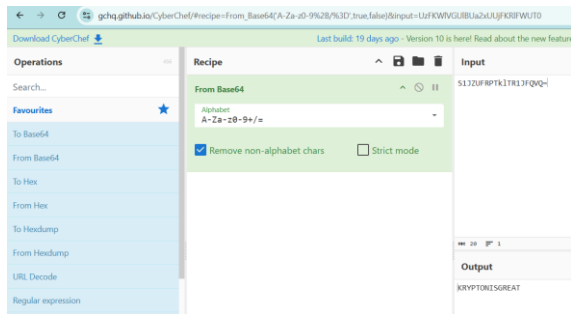
Step 2: Decoding the Password Using CyberChef

- **Tool Used:** CyberChef (Online Tool)
- **Action Taken:**
Used the "From Base64" operation in CyberChef.
- **Explanation:**
 - The Level 0 challenge presented a string that was Base64 encoded.
 - Base64 is a common encoding method that converts binary data into ASCII text.
 - In CyberChef:
 - Pasted the encoded string.
 - Selected the "From Base64" operation.

- Successfully decoded the string to reveal the **plaintext password** for Level 1.

Purpose:

- To obtain the login password for connecting to Krypton1.



Step 3: SSH Login into Krypton1 Using Kali Linux

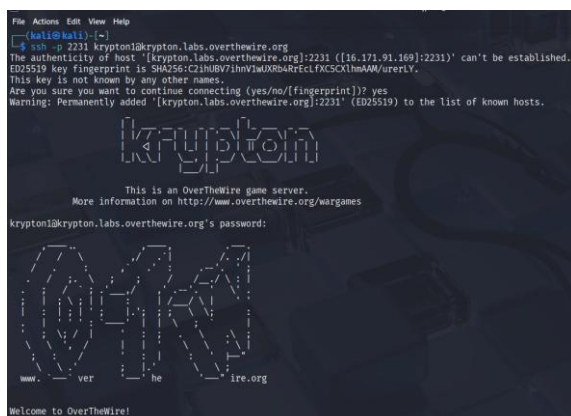
- **Tool Used:** Kali Linux Terminal (SSH Client)
- **Command Executed:** `ssh -p 2231 krypton1@krypton.labs.overthewire.org`

Explanation:

- Used the decoded password from CyberChef to log in.
- Connected to the remote server over port 2231 using SSH.
- Logged into the `krypton1` account to begin Level 1.

Purpose:

- To access the Krypton1 server and proceed with the next level of the wargame.



Level 1 → Level 2

Step 1: Accessing Level 1 Instructions in Browser

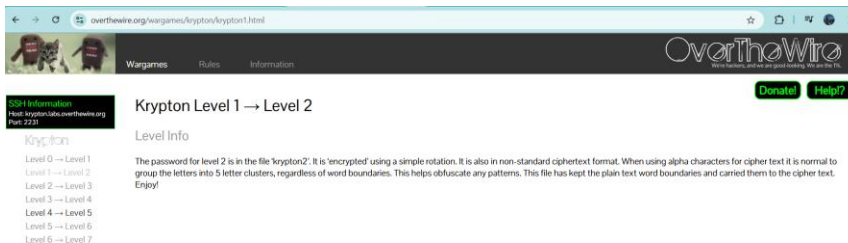
- **Tool Used:** Web Browser
- **Action Taken:**
Opened the URLs:
<https://overthewire.org/wargames/krypton/krypton1.html>

Explanation:

- These pages contained:
 - Specific guidance for solving **Level 1**.
 - Details about the type of encryption used in the challenge.
- The information hinted that the text was encoded using a **simple cipher**.

Purpose:

- To understand the context of the encryption method used in Level 1 and gather hints before proceeding.



Step 2: Exploring Files on the Krypton1 Server

- **Tool Used:** Kali Linux Terminal
- **Commands Executed:** `cd /krypton/krypton1`

```
: ls
```

```
: cat README
```

```
: cat krypton2
```

Explanation:

- Navigated to the `/krypton/krypton1` directory where challenge files were stored.
- Listed available files using `ls`.
- Read the `README` file, which provided hints about the encryption type.
- Displayed the contents of the `krypton2` file, which contained the **encrypted text** needed to solve the level.

Observation:

- The encryption used was identified as **ROT13** based on the README hints.

```
krypton1@bandit:~$ cd /krypton/krypton1
krypton1@bandit:/krypton/krypton1$ ls
krypton2 README
krypton1@bandit:/krypton/krypton1$ cat README
Welcome to Krypton!

This game is intended to give hands on experience with cryptography
and cryptanalysis. The levels progress from classic ciphers, to modern,
easy to harder.

Although there are excellent public tools, like cryptool, to perform
the simple analysis, we strongly encourage you to try and do these
without them for now. We will use them in later exercises.

** Please try these levels without cryptool first **

The first level is easy. The password for level 2 is in the file
'krypton2'. It is 'encrypted' using a simple rotation called ROT13.
It is also in non-standard ciphertext format. When using alpha characters for
cipher text it is normal to group the letters into 5 letter clusters,
regardless of word boundaries. This helps obfuscate any patterns.

This file has kept the plain text word boundaries and carried them to
the cipher text.

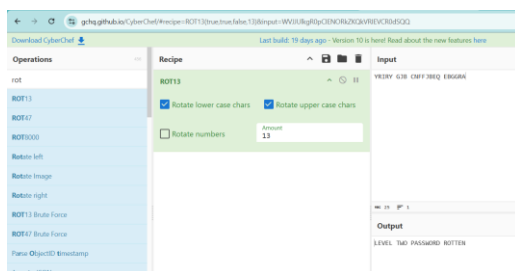
Enjoy!
krypton1@bandit:/krypton/krypton1$ cat krypton2
VIRIV GJB CNFFJBEQ EBGGRA
krypton1@bandit:/krypton/krypton1$ ^C
krypton1@bandit:/krypton/krypton1$
```

Step 3: Decoding the Encrypted Text Using CyberChef

- Tool Used:** CyberChef (Online Tool)
- Action Taken:**
Applied the "ROT13" operation on the encrypted text.
- Explanation:**
 - ROT13** (Rotate by 13 places) is a simple letter substitution cipher.
 - In CyberChef:
 - Pasted the ciphertext from krypton2.
 - Selected the "ROT13" operation.
 - Successfully decoded the text to reveal the password for **Krypton2**.

Purpose:

- To obtain the next level's password by decoding the ROT13 ciphered text.



Level 2 → Level 3

Step 1: Accessing Level 2 Instructions in Browser

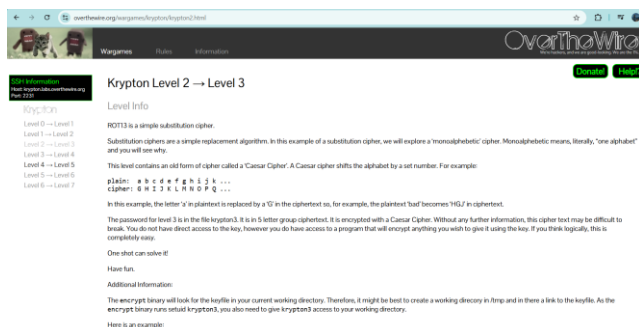
- **Tool Used:** Web Browser
- **Action Taken:**
Opened the URL =
<https://overthewire.org/wargames/krypton/krypton2.html>

Explanation:

- This page provided:
 - Hints related to the encryption method used in Level 2.
 - Instructions suggesting that some binary encryption mechanism might be involved.

Purpose:

- To understand the encryption challenge setup for Level 2.

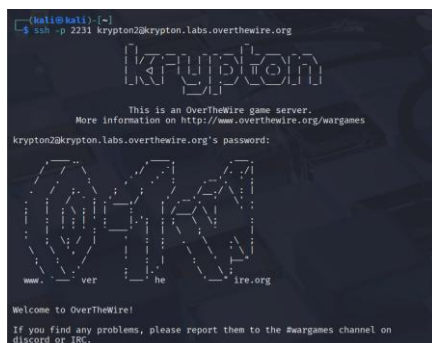


Step 2: SSH Login into Krypton2

- **Tool Used:** Kali Linux Terminal (SSH Client)
- **Command Executed:** `ssh -p 2231 krypton2@krypton.labs.overthewire.org`
- **Password Used:** ROTTEN (obtained from previous level)
- **Explanation:**
 - Connected to the Krypton2 server using the decoded password.
 - SSH used port 2231 as specified.

Purpose:

- To gain access to the environment for solving Level 2.



Step 3: Navigating to Challenge Directory and Reading Files

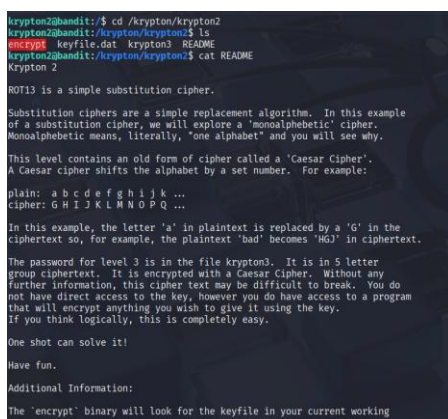
- **Tool Used:** Kali Linux Terminal
- **Commands Executed=** cd /krypton/krypton2

= ls

= cat README

Explanation:

- Changed into the /krypton/krypton2 directory.
- Listed all files to find challenge-related materials.
- Read the README file to understand the task.
- Observed that there was a binary called encrypt and a data file called keyfile.dat.



Step 4: Creating a Temporary Directory and Simulating Encryption

- **Tool Used:** Kali Linux Terminal
- **Commands Executed:**

```
= mktemp -d
= cd /tmp/tmp.Wf20nCpCDQ
= ln -s /krypton/krypton2/keyfile.dat
= ls
= chmod 777 .
= /krypton/krypton2/encrypt /etc/issue
= ls
= cat krypton3
```

Explanation:

- Created a temporary working directory using mktemp.
- Linked the keyfile.dat file into the temp directory using ln -s.
- Changed permissions to allow full access using chmod 777.
- Used the encrypt binary to encrypt /etc/issue.
- After encryption, found a file containing the ciphertext (krypton3).

Purpose:

- To simulate the encryption environment and obtain data for decryption.

```
krypton2@melinda:~$ mktemp -d
/tmp/tmp.Wf20nCpCDQ
krypton2@melinda:~$ cd /tmp/tmp.Wf20nCpCDQ
krypton2@melinda:/tmp/tmp.Wf20nCpCDQ$ ln -s /krypton/krypton2/keyfile.dat
krypton2@melinda:/tmp/tmp.Wf20nCpCDQ$ ls
keyfile.dat
krypton2@melinda:/tmp/tmp.Wf20nCpCDQ$ chmod 777 .
krypton2@melinda:/tmp/tmp.Wf20nCpCDQ$ /krypton/krypton2/encrypt /etc/issue
krypton2@melinda:/tmp/tmp.Wf20nCpCDQ$ ls
ciphertext  keyfile.dat
krypton2@bandit:/krypton/krypton2$ cat krypton3
ONQJEMDUEQMEK
krypton2@bandit:/krypton/krypton2$
```

Step 5: Getting the Ciphertext Again

- **Tool Used:** Kali Linux Terminal
- **Commands Executed:**

```
= mktemp -d
= cd /tmp/tmp.309IVIOQps
= ln -s /krypton/krypton2/keyfile.dat
= ls
= chmod 777 .
= /krypton/krypton2/encrypt /etc/issue
```

= ls

= cat ciphertext

Explanation:

- Repeated the encryption simulation in a new temporary directory.
- Retrieved fresh ciphertext output for further analysis.
- Prepared the ciphertext needed for online decryption tools.

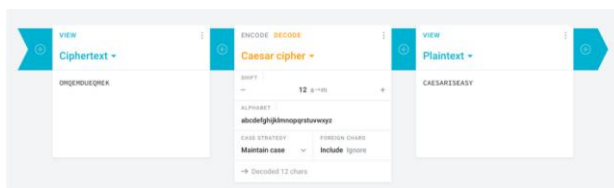
```
krypton2@bandit:~$ mkdir -p /tmp/tmp.3091V10Qps
krypton2@bandit:~$ cd /tmp/tmp.3091V10Qps
krypton2@bandit:/tmp/tmp.3091V10Qps$ ln -s /krypton/krypton2/keyfile.dat
krypton2@bandit:/tmp/tmp.3091V10Qps$ ls
keyfile.dat
krypton2@bandit:/tmp/tmp.3091V10Qps$ chmod 777 .
krypton2@bandit:/tmp/tmp.3091V10Qps$ /krypton/krypton2/encrypt /etc/issue
krypton2@bandit:/tmp/tmp.3091V10Qps$ ls
ciphertext keyfile.dat
krypton2@bandit:/tmp/tmp.3091V10Qps$ cat ciphertext
GNGZFGXFEZKkrypton2@bandit:/tmp/tmp.3091V10Qps$
```

Step 6: Decrypting Ciphertext Using Cryptii

- **Tool Used:** Web Browser + Cryptii (Online Cipher Tool)
- **Action Taken:**
 - Pasted the ciphertext into
 - <https://cryptii.com/pipes/caesar-cipher>
- **Explanation:**
 - Used the **Caesar Cipher Decryption Tool** at Cryptii.
 - Adjusted the settings to try all possible rotations.
 - Successfully found the correct decryption that revealed the password for Krypton3.

Purpose:

- To decode the ciphertext and retrieve the password needed to move to the next level.



Level 3 → Level 4

Step 1: Accessing Level 3 Instructions in Browser

- **Tool Used:** Web Browser
- **Action Taken:**

Opened the URL:

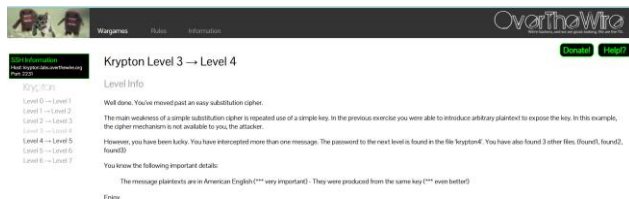
<https://overthewire.org/wargames/krypton/krypton3.html>

Explanation:

- The webpage provided:
 - Hints that the encryption might involve a substitution cipher.
 - Suggested that frequency analysis would be useful to solve the level.

Purpose:

- To understand the goal and strategy needed for Krypton Level 3.



Step 2: SSH Login into Krypton3

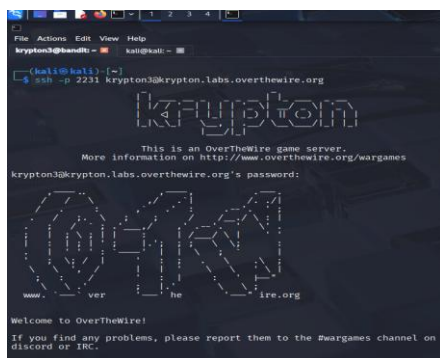
- **Tool Used:** Kali Linux Terminal (SSH Client)
- **Command Executed:** `ssh -p 2231 krypton3@krypton.labs.overthewire.org`

Explanation:

- Logged into the Krypton3 user account using the password obtained from the previous level.
- Connected over SSH on port 2231.

Purpose:

- To access the environment where Level 3 challenge files were located.



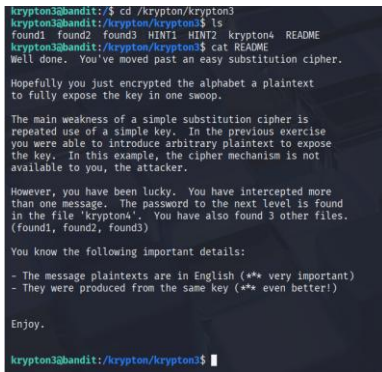
Step 3: Navigating to Challenge Directory and Listing Files

- **Tool Used:** Kali Linux Terminal
- **Commands Executed:**

```
= cd /krypton/krypton3
= ls
= cat README
```

Explanation:

- Changed into the /krypton/krypton3 directory.
- Listed the files and found important ones including README.
- Reading the README hinted that some classic cryptographic technique (frequency analysis) would help.



```
krypton3@bandit:/$ cd /krypton/krypton3
krypton3@bandit:/krypton/krypton3$ ls
found1 found2 found3 HINT1 HINT2 krypton4 README
krypton3@bandit:/krypton/krypton3$ cat README
Well done. You've moved past an easy substitution cipher.

Hopefully you just encrypted the alphabet a plaintext
to fully expose the key in one swoop.

The main weakness of a simple substitution cipher is
repeated use of a simple key. In the previous exercise
you were able to introduce arbitrary plaintext to expose
the key. In this example, the cipher mechanism is not
available to you, the attacker.

However, you have been lucky. You have intercepted more
than one message. The password to the next level is found
in the file 'krypton4'. You have also found 3 other files.
(found1, found2, found3)

You know the following important details:
- The message plaintexts are in English (** very important)
- They were produced from the same key (** even better!)

Enjoy.

krypton3@bandit:/krypton/krypton3$
```

Step 4: Exploring Important Files

- **Tool Used:** Kali Linux Terminal
- **Commands Executed:**

```
= cat found1
```

```
= cat found2
```

Explanation:

- Displayed the contents of found1 and found2.
- These files contained encrypted text, likely using a **monoalphabetic substitution cipher**.

Purpose:

- To collect the ciphertext needed for decryption.

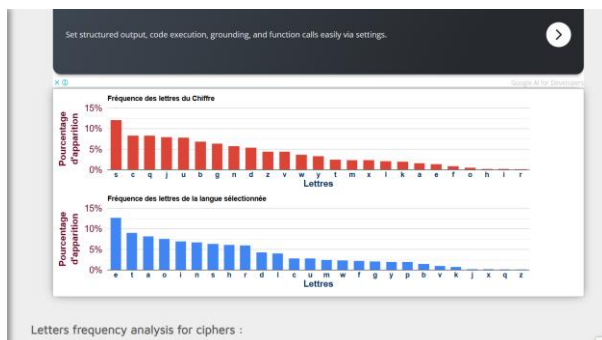
[illegible]

Step 5: Performing Frequency Analysis

- **Tool Used:** Manual Frequency Analysis / Decryption Tools
- **Action Taken:**
 - Observed letter frequencies in found2.
 - Identified the most commonly appearing letters and guessed their substitutions based on typical English language frequency (e.g., E, T, A, O are common in English).

Purpose:

- To begin cracking the substitution cipher systematically.



Step 6: Viewing the Final Ciphared Password

- **Tool Used:** Kali Linux Terminal
- **Command Executed:** cat krypton4

Explanation:

- Displayed the contents of `krypton4`.
- The file contained another encrypted text related to the password needed for the next level.

Purpose:

- To gather additional cipher material for full decryption.

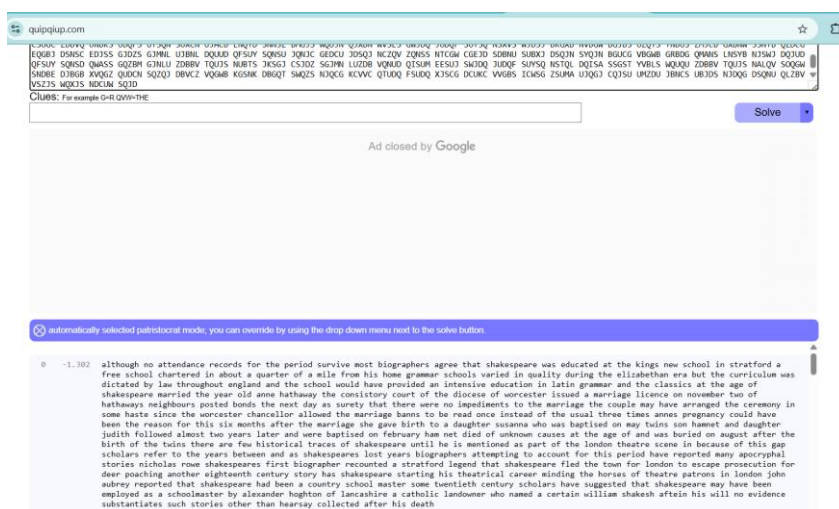
```
krypton3@bandit:/krypton/krypton3$ cat krypton4
KSVVW BGSJD SVSIS VXBMN YOUUK BNWCU ANMJS krypton3@bandit:/krypton/krypton3$
```

Step 7: Decrypting the Ciphertext Using Quipqiup

- **Tool Used:** Web Browser + Quipqiup (Online Substitution Cipher Solver)
- **Action Taken:**
 - Opened: <https://quipqiup.com/>
 - Pasted the content from found2.
 - Used Quipqiup to automatically solve the monoalphabetic substitution cipher.

Purpose:

- To obtain the partially decrypted text, including important keyword mappings.

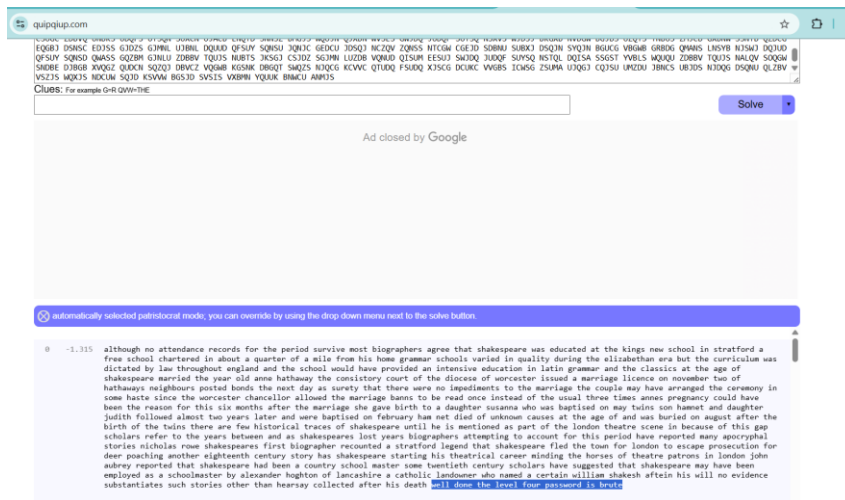


Step 8: Final Decryption to Obtain the Password

- **Action Taken:**
 - Appended and decrypted the final krypton4 file contents.
 - Carefully adjusted the decryption mapping to fully reveal the password.

Purpose:

- To extract the password required to move to Krypton Level 4.



Level 4 → Level 5

Step 1: Accessing Level 4 Instructions in Browser

- **Tool Used:** Web Browser
- **Action Taken:**

Opened the URL:

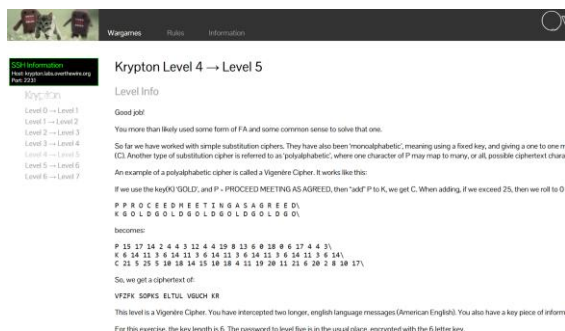
<https://overthewire.org/wargames/krypton/krypton4.html>

Explanation:

- The page provided hints suggesting that the encryption method involved the **Vigenère cipher**, a more complex polyalphabetic cipher.
- Resources like Wikipedia were recommended to better understand how Vigenère ciphers work.

Purpose:

- To prepare for decrypting the text by understanding the cipher mechanism.



Step 2: SSH Login into Krypton4

- **Tool Used:** Kali Linux Terminal (SSH Client)
- **Command Executed:** `ssh -p 2231 krypton4@krypton.labs.overthewire.org`

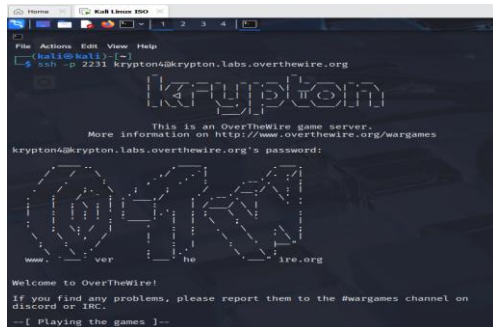
- **Password Used:** BRUTE

Explanation:

- Logged into the Krypton4 user account using the password obtained from the previous level.
- Accessed over SSH port 2231.

Purpose:

- To access Level 4 challenge files and attempt decryption.



Step 3: Identifying Encryption Type from README

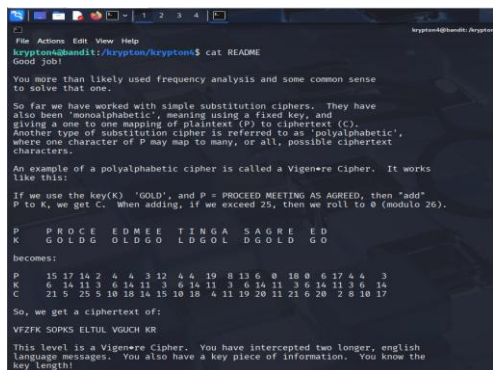
- **Tool Used:** Kali Linux Terminal
- **Command Executed:** cat README

Explanation:

- Read the README file inside the challenge directory.
- Confirmed that the challenge involved a **Vigenère cipher**.

Purpose:

- To confirm the encryption method before attempting decryption.



Step 4: Exploring Directory Files

- **Tool Used:** Kali Linux Terminal
- **Commands Executed:**

```
= ls
= cd /krypton/krypton4
= ls
= cat found1
= cat krypton5
```

Explanation:

- Navigated into the /krypton/krypton4 directory.
- Listed available files including found1 and krypton5.
- Displayed contents:
 - found1 contained ciphertext encrypted using the Vigenère cipher.
 - krypton5 likely held additional encoded information for the next level.

```
krypton@bandit:~$ ls
krypton@bandit:~$ cd /krypton/krypton4
krypton@bandit:~/krypton4$ ls
found1 found2 hint krypton5 README
krypton@bandit:~/krypton4$ cat found1
YTC5 3E1B K0VY K1EM 1EAPR 30V0 0V0L 0K1B 5SLX R1QV1 1D0GT W0R1C RVV0P BUZK1 YL1ZP DLCD1 1NGJ3 U0R1P TF0GL 0W0R 1E2RV NMV5P J0LCL R00N3 NM0XK F0L5P 3G0VV ER1TT 0
Q0RM W0M0N 1TX0G J13VY T5Y0L Q0ZTT 05P1T J0B1Y V5S1E L1K0L R03G5 W0R1C C1T5C V0D1C L0R03 M0F1B 3P0V0 NM0ZK G0W0F U0P0B X0S0L C0Z0U 1B0ZK M0M0K L00CK K0C0C 1U5B5 0M0P0 1P
J0N K5P1R M0R0R V0M0R M0K0E P1Z0K 50V1T 0E0J3 RY0L0 J0Z0U K0L0M R1F0D 1YV0E 1L0M0 L0R1Y V0D0R K2R0L 0N0R0 V0M0R 0L0L3 V0VFF K0S0P 0V10M W0V1Y M0C0L 0V10R 5000P 0R0L1 5P0
CC M2R0D M0V1Y M055K V150V V1G0L P1Z10N P1M0V R0N0H 1550E 1M0L1 P5E03 0V10M 0P01C 10V1E Z0LTK M0M0Y V1G0V W0M0S U0V0G V150R 1050J 1B0M0 V0V1R M0X0M B0M0D V050M W0M0Y 10V1
N 1M0M0 K0L0K VYV0X 5R05V M0C0V W0L0V 1511B M0V1L 0T11C 5050X EYV0C C0L0M K0L0F J05E0 8R0D0 W0F0M C5D0F Y0K0M V0L0M K0L0V C0K0E X0F01 M0P1N 50M0F Z0M0A P0M0R 0K0L1B G0F0H
M050F M0P1T M0V1Y T0L0K 0T0L0 V1V1T 05M0V J0M0R V1E1R W0M0R 0V0M0 R050T M0L0F 1P050 V05M0 1Y0E0 0P0R0 0L0P0 M0K0E 1L050 V0M0R Y0F01 Y0U0E V0L0X V0M0Y 0510J J05E0 P0505
11115 11115 P0M0X K0E0K U500K 1Y1B1 Y500R 1P1M0 0L0FZ W010F 0H1Y0 VYV0Y M0U5B 5M0M0 0M0X0 5R01M V1E1R VYV05P 1111C M0M0L K0M0K 5M0LY Y1Z0E FT1LY R5F0D 5P11M 0V0M0 W0M0F 1
500B M0M0N 1L05K K0M0Y 0100L KZ0P3 0L51C VY10M Z0R0K W0M0H 05M0F M051M M0M0N 1C50K 0V10B 1C51D V0P0E M050F J0M0R J0M0S P1K0L M0M0N C0F0V 00V1Q 050M0 M0K0S M051P K0F1P 0L
050 1C0F3 P051Y 510V0 P1M0Y V115H M051B 5E0P3 1E0M0 M051P M0V10 L0M0V 0V05P V0M0X 05R0M J110N 1YV0E 0R0LX B0M0K L0M0K K0M0R M0K0S 550M0 W050M V051Z R0L0F M0M0N T0M0L 0L
R0C L0M0L V050K L0M0K R0M0D T10R1 P1M0B 1L1V0 010V1 5P0X0 1000Q M0M0N M0Y0D F0K1Y 0R0LY 0M51P K0L1L 1K505 Y500R 1J0F0 G1P0K M01B0 0H0V0 1F0X1 M0E0R 1E0M0 C0L1N 1VYV0 C10T
U M0L0L C0F0M 05P0R M0F0J G0M0H 1550C 50M0K U0R0G krypton@bandit:~/krypton4$ cat krypton5
HCK1V R30Xkrypton@bandit:~/krypton4$
```

Step 5: Decrypting Vigenère Cipher Using Dcode

- **Tool Used:** Web Browser + Dcode (Vigenère Cipher Solver)
- **Action Taken:**
 - Opened: <https://www.dcode.fr/vigenere-cipher>
 - Pasted the ciphertext from found1.
 - Edited settings to guess a key length of **6 characters** (based on hints).
 - Clicked "DECRYPT" to automatically attempt decoding.

Purpose:

- To recover the plaintext and identify the hidden password.



Step 6: Getting Possible Key Candidates

- **Tool Used:** Dcode Vigenère Cipher Tool
- **Explanation:**
 - After setting the key length correctly, the tool output two possible 6-character keys.
 - These candidate keys were tested to reveal readable English text.

Purpose:

- To narrow down the correct decryption key used for encrypting the challenge text.

THEGI RLTHA TWILL BEEAS YREPL IEDTH EMANF ORWHE NSHEK NOWSY OUARE INTHE COUNT RYOFT HEWIN KIESS HEWIL LFIND YOUAN DMAKE YOUAL LHERS LAVES PERHA PSNOT SAIDT HESCA RECRO WFORW EMEAN TODES TROYH EROHT HATIS DIFFE RENTS AIDTH EGUAR DIANO FTHEG ATESN OONEH ASEVE RDEST ROYED HERBE FORES OINAT URALL YTHOU GHTSH EWOUU DMAKE SLAVE SOFYO UASSH EHASO FTHER ESTBU TTAKE CAREF ORSHE ISWIC KEDAN OFTER CEAND MAYNO TALLO WYOUT ODEST ROYHE RKEEP TOTHE WESTW HERET HESUN SETSA NDYOU CANNO TFAIL TOFIN DHERT HEYTH ANKED HIMAN DBADE HINGO ODBYE ANDTU RNEDT OWARD THEWE STWAL KINGO VERFI ELDSO FSOFT GRASS DOTTE DHERE ANDTH EREWI THDAI SIESA NDBUT TERCU PSDOR OTHYS TILLW ORETH EPRET TYSIL KDRES SSHEH ADPUT	FREAEY	SKIDT HECCA RECBO WFORG EMEAN DODES TBOYH ERYHT HATSS DIFFO RENTS KIDTH EQUAR DIKNO FTHOG ATESX OONEH KSEVE RNEST ROIED HERLE FOREC OINAT ERALL YDHOU GHDSH EWOL E DMAKO SLAVE COFYO UKSSH EHKSO FTHOR ESTBE TTAKE MAREF OBSHE ISGIC KEDKN DFIEB CEAND WAYNO TKLLO WYVUT ODECT ROYHO RKEEP DOTHE WOSTW HEBET HESEN SETSK NDYOU MANNO TPAIL TOPIN DHEBT HEYTR ANKED RIMAN DLADE HIWGO ODBIE ANDTE RNEDT YWARD TREWE STGAL KINQO VERFS ELDSO PSOFT GBASS DODTE DHEBE ANDTR EREWI DHDAI SSES A NDLOT TERMU PSDOB OTHYS DILLW OBETH
---	--------	---

Step 7: Final Password Retrieval Using Fool's Online Decryptor

- **Tool Used:** Web Browser + Fool's Vigenère Decryptor
- **Action Taken:**
 - Opened: <https://f00l.de/hacking/vigenere.php>
 - Pasted the encrypted text.

- Used one of the identified keys to successfully decrypt and reveal the password for **Krypton5**.

Purpose:

- To fully decrypt the text and acquire the password needed for advancing to Level 5.

VIGENERE - ONLINE VIGENERE ANALYSIS AND CRACKING

[- home](#) - [backlist tools](#) -

breaking a vigenere cipher. First step will be calculation or guessing the key length your text has been encrypted with. Then we have to crack the key the key cannot be cracked correctly, you may try to use some known plain text attacks. In the end your text will be properly decrypted.

Ciphertext HCTIV R30R	Analyze cipher text to calculate key length. <input type="button" value="analyze"/>
Key length: <input type="text" value="0"/>	Try to crack key based on key length. <input type="button" value="crack"/>
Key: <input type="text" value="key"/>	Decrypt cipher text using key. <input type="button" value="decrypt"/>
Clear text	

Level 5 → Level 6

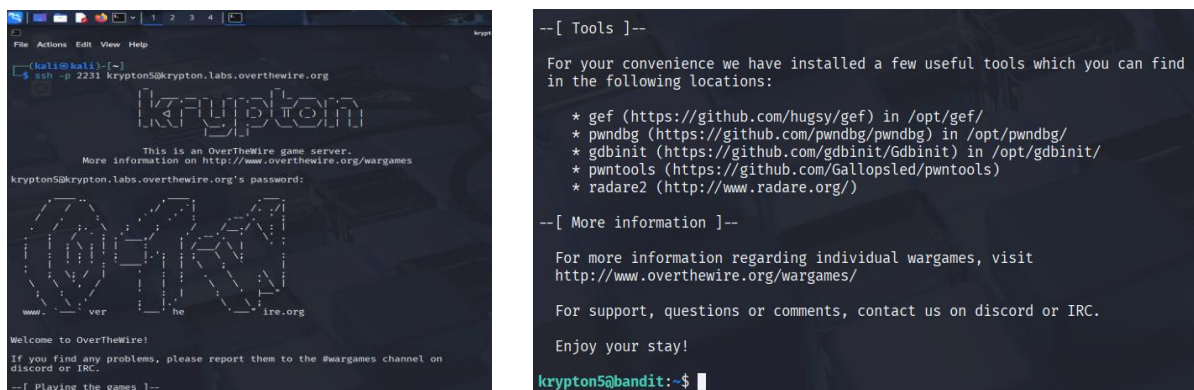
Step 1: Accessing Level 5 Instructions in Browser

- **Tool Used: Web Browser**
- **Action Taken:**
 - **Opened the URL:**
<https://overthewire.org/wargames/krypton/krypton5.html>
- **Explanation:**
 - The webpage gave hints that the encryption is a simple substitution cipher — most likely a Caesar Cipher.
- **Purpose:**
 - To understand the type of encryption used and prepare for solving the challenge.

Step 2: SSH Login into Krypton5

- **Tool Used: Kali Linux Terminal (SSH Client)**

- **Command Executed:** `ssh -p 2231 krypton5@krypton.labs.overthewire.org`
- **Password Used:**
CLEARTEXT (from previous level)
- **Explanation:**
 - Connected to the Krypton5 server on port 2231 using SSH.
- **Purpose:**
 - To access Krypton5 and retrieve the encrypted files.



The left screenshot shows a terminal window with the command `ssh -p 2231 krypton5@krypton.labs.overthewire.org` being executed. The output displays the 'KRYPTON' logo, a welcome message, and a list of installed tools. The right screenshot shows the 'Tools' section of the welcome message, listing tools like gef, pwndbg, gdbinit, pwntools, and radare2 with their respective GitHub links.

```
--[ Tools ]--

For your convenience we have installed a few useful tools which you can find
in the following locations:

* gef (https://github.com/hugsy/gef) in /opt/gef/
* pwndbg (https://github.com/pwndbg/pwndbg) in /opt/pwndbg/
* gdbinit (https://github.com/gdbinit/Gdbinit) in /opt/gdbinit/
* pwntools (https://github.com/Gallopsled/pwntools)
* radare2 (http://www.radare.org/)

--[ More information ]--

For more information regarding individual wargames, visit
http://www.overthewire.org/wargames/

For support, questions or comments, contact us on discord or IRC.

Enjoy your stay!

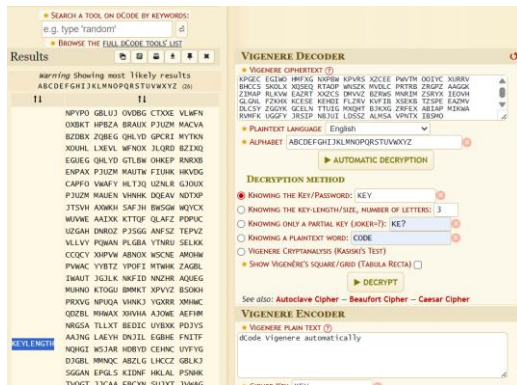
krypton5@bandit:~$
```

Step 3: Navigating to Challenge Directory and Reading Files

- **Tool Used:** Kali Linux Terminal
- **Commands Executed:**

```
= cd /krypton/krypton5
= ls
= cat README
```
- **Explanation:**
 - Moved into the `/krypton/krypton5` directory.
 - Listed the contents.
 - Read the README, which gave more hints about the encryption approach.
- **Purpose:**
 - To gather background information about the encryption mechanism.

- **Purpose:**
 - To retrieve the password file for moving forward.



Step 6: Accessing and Reading the Final Password File

- **Tool Used:** Kali Linux Terminal
- **Commands Executed:**

```
ls
cat krypton6
```

- **Explanation:**
 - Listed all files again inside the /krypton/krypton5 directory.
 - Found a file named krypton6.
 - Displayed the contents of krypton6, which contained either plaintext or further encrypted text.
- **Purpose:**
 - To retrieve the text (likely a password) required for proceeding to the next level, Krypton6.

```
krypton5@bandit:/krypton/krypton5$ ls
found1 found2 found3 krypton6 README
krypton5@bandit:/krypton/krypton5$ cat krypton6
BELOS Zkrypton5@bandit:/krypton/krypton5$
```

Step 7: Final Decryption to Obtain the Password

- **Tool Used:** Web Browser + Fool's Online Vigenère Decryptor
- **Action Taken:**
 - Opened the URL: <https://f00l.de/hacking/vigenere.php>.
 - Pasted the ciphertext (from krypton6) into the decryption box.

- Used the decryptor to decode the text if it was still encrypted with Vigenère cipher.
- **Explanation:**
 - If the content was encrypted, the Fool's online tool helped fully decrypt the hidden password.
 - Otherwise, it was just verified through the tool.
- **Purpose:**
 - To finally reveal the **correct password** needed to log into **Krypton6** and move forward.

The screenshot shows a web interface titled "VIGENERE - ONLINE VIGENERE ANALYSIS AND CRACKING". It includes a small introductory text about the tool's purpose. The interface has three main sections:

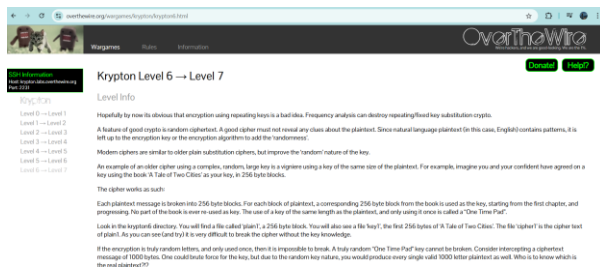
- Top section:** A text input field labeled "Ciphertext" with the value "WUJCN 7" entered. To the right is a button labeled "analyze".
- Middle section:** A text input field labeled "Key length" with the value "5" entered. To the right is a button labeled "check".
- Bottom section:** A text input field labeled "Key" with the value "keylength" entered. To the right is a button labeled "decrypt".

 There are also some smaller, less legible text elements and a "decrypt" button at the bottom right.

Level 6 → Level 7

Step 1: Accessing Level 6 Instructions in Browser

- **Tool Used: Web Browser**
- **Action Taken:**
 - **Opened the URL:**
<https://overthewire.org/wargames/krypton/krypton6.html>
- **Explanation:**
 - The page provided an overview of the Level 6 challenge.
 - It hinted that multiple key files and bitwise operations might be involved.
- **Purpose:**
 - To understand the challenge structure before starting.



Step 2: SSH Login into Krypton6

- **Tool Used: Kali Linux Terminal (SSH Client)**
- **Command Executed: `ssh -p 2231 krypton6@krypton.labs.overthewire.org`**
- **Password Used: RANDOM (obtained from previous level)**
- **Explanation:**
 - Logged into the Krypton6 server via SSH over port 2231.
- **Purpose:**
 - To access the server environment and available files for Level 6.



Step 3: Navigating to the Krypton6 Directory and Reading README

- **Tool Used: Kali Linux Terminal**
- **Commands Executed:**
 - `= cd /krypton/krypton6`
 - `= ls`
 - `= cat README`
- **Explanation:**
 - Changed directory to /krypton/krypton6.

- Listed available files.
- Read the README, which explained that bitwise XOR operations and keyfiles were involved.
- Purpose:
 - To understand the technical hints for solving the level.

```
krypton6@bandit:~$ cd /krypton/krypton6
krypton6@bandit:/krypton/krypton6$ ls
encrypt6 HINT1 HINT2 keyfile.dat krypton7 onetime README
krypton6@bandit:/krypton/krypton6$ cat README

Hopefully by now its obvious that encryption using repeating keys
is a bad idea. Frequency analysis can destroy repeating/fixed key
substitution crypto.

A feature of good crypto is random ciphertext. A good cipher must
not reveal any clues about the plaintext. Since natural language
plaintext (in this case, English) contains patterns, it is left up
to the encryption key or the encryption algorithm to add the
'randomness'.

Modern ciphers are similar to older plain substitution
ciphers, but improve the 'random' nature of the key.

An example of an older cipher using a complex, random, large key
is a vigenere using a key of the same size of the plaintext. For
example, imagine you and your confident have agreed on a key using
the book 'A Tale of Two Cities' as your key, in 256 byte blocks.

The cipher works as such:

Each plaintext message is broken into 256 byte blocks. For each
block of plaintext, a corresponding 256 byte block from the book
is used as the key, starting from the first chapter, and progressing.
No part of the book is ever re-used as key. The use of a key of the
same length as the plaintext, and only using it once is called a 'One Time Pad'.

Look in the krypton6/onetime directory. You will find a file called 'plain1', a 256
byte block. You will also see a file 'key1', the first 256 bytes of
'A Tale of Two Cities'. The file 'cipher1' is the cipher text of
plain1. As you can see (and try) it is very difficult to break
the cipher without the key knowledge.

Its time to employ a stream cipher. A stream cipher attempts to create
an on-the-fly 'random' keystream to encrypt the incoming plaintext one
byte at a time. Typically, the 'random' key byte is xor'd with the
plaintext to produce the ciphertext. If the random keystream can be
replicated at the receiving end, then a further xor will produce the
plaintext once again.

From this example forward, we will be working with bytes, not ASCII
text, so a hex editor/dumper like hexdump is a necessity. Now is the
right time to start to learn to use tools like cryptool.

In this example, the keyfile is in your directory, however it is
not readable by you. The binary 'encrypt6' is also available.
It will read the keyfile and encrypt any message you desire, using
the key AND a 'random' number. You get to perform a 'known ciphertext'
attack by introducing plaintext of your choice. The challenge here is
not simple, but the 'random' number generator is weak.

As stated, it is now that we suggest you begin to use public tools, like cryptool,
to help in your analysis. You will most likely need a hint to get going.
See 'HINT1' if you need a kickstart.

If you have further difficulty, there is a hint in 'HINT2'.

The password for level 7 (krypton7) is encrypted with 'encrypt6'.

Good Luck!
```

Step 4: Exploring Additional Hints

- Tool Used: Kali Linux Terminal
- Commands Executed:
 - = cat HINT1
 - = cat HINT2
- Explanation:
 - Read HINT1 and HINT2 files.
 - These hints described details about how to use the keyfiles and the encryption process.
- Purpose:

- To get further insights necessary to build the correct decryption strategy.

```
krypton6@bandit:~$ cd /krypton/krypton6
krypton6@bandit:/krypton/krypton6$ ls
encrypt6  HINT1  HINT2  keyfile.dat  krypton7  onetime  README
krypton6@bandit:/krypton/krypton6$ cat HINT1
The 'random' generator has a limited number of bits, and is periodic.
Entropy analysis and a good look at the bytes in a hex editor will hel
There is a pattern!
krypton6@bandit:/krypton/krypton6$ cat HINT2
8 bit LFSR
```

Step 5: Investigating Krypton7 Encrypted File

- **Tool Used: Kali Linux Terminal**
- **Commands Executed:**
 - = `cat krypton7`
 - = `mkdir /tmp/lev7`
- **Explanation:**
 - Displayed the encrypted krypton7 file.
 - Created a temporary directory /tmp/lev7 for safe experimentation.
- **Purpose:**
 - To prepare a workspace for linking and analyzing the encryption.

```
krypton6@bandit:/krypton/krypton6$ cat krypton7
PNUKLYLWRQKGKBE
krypton6@bandit:/krypton/krypton6$ mkdir /tmp/lev7
```

Step 6: Setting Up Temporary Workspace with Linked Files

- **Tool Used: Kali Linux Terminal**
- **Commands Executed:**
 - = `cd /tmp/lev7`
 - = `ln -s /krypton/krypton6/keyfile00.dat`
 - = `ln -s /krypton/krypton6/krypton7`
- **Explanation:**
 - Linked keyfile00.dat and krypton7 into the /tmp/lev7 directory using symbolic links.
- **Purpose:**
 - To have the necessary files in a writable and safe temporary workspace.


```
krypton6@bandit:/krypton/krypton6$ cd /tmp/lev7
krypton6@bandit:/tmp/lev7$ ln -s /krypton/krypton6/keyfile00.dat
krypton6@bandit:/tmp/lev7$ ln -s /krypton/krypton6/krypton7
```

Step 7: Linking Additional Keyfile and Setting Permissions

- **Tool Used: Kali Linux Terminal**
- **Commands Executed:**
 - = ls
 - = ln -s /krypton/krypton6/keyfile.dat
 - = chmod 777 .
- **Explanation:**
 - Listed files to confirm links.
 - Linked keyfile.dat and gave full permissions (read/write/execute) to the directory.
- **Purpose:**
 - To avoid any permission-related issues while processing the files.

```
krypton6@bandit:/tmp/lev7$ ls
a.txt  cipher.txt  cipher.txt  k  keyfile00.dat  keyfile.dat  krypton6  krypton7  main.py
krypton6@bandit:/tmp/lev7$ ln -s /krypton/krypton6/keyfile.dat
ln: failed to create symbolic link './keyfile.dat': File exists
krypton6@bandit:/tmp/lev7$ chmod 777 .
```

Step 8: Encrypting Sample File and Studying the Ciphertext

- **Tool Used: Kali Linux Terminal**
- **Commands Executed:**
 - = cat a.txt
 - = /krypton/krypton6/encrypt6 a.txt cipher.txt
 - = cat cipher.txt
 - = man xxd
- **Explanation:**
 - Created a plaintext a.txt.
 - Used the encrypt6 binary to encrypt it into cipher.txt.
 - Read the manual for xxd, a tool for visualizing binary/hex data.
- **Purpose:**
 - To simulate how encryption worked and understand how plaintext was being transformed.

```

krypton@bandit:/tmp/lev5$ cat a.txt
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
krypton@bandit:/tmp/lev5$ /krypton/krypton6/encrypt6 a.txt cipher.txt
krypton@bandit:/tmp/lev5$ cat cipher.txt
UICIDGVYZKTHNSIRFYCPUEOCKRNEICTDGVYZKTHNSIRFYCPUEOCKRNEICTDGVYZKTHNSIRFYCPUEOCKRNEkrypton@bandit:/tmp/lev5$
krypton@bandit:/tmp/lev5$
krypton@bandit:/tmp/lev5$
krypton@bandit:/tmp/lev5$
krypton@bandit:/tmp/lev5$ man xxd

```

Step 9: Binary-Level Analysis Using Xxd

- **Tool Used: Kali Linux Terminal**
- **Commands Executed:**
 - `= xxd -b a.txt`
 - `= xxd -b cipher.txt`
- **Explanation:**
 - Viewed the contents of both a.txt and cipher.txt in binary format (bit-by-bit).
- **Purpose:**
 - To analyze and reverse-engineer the encryption at the bit level.

```

File Actions Edit View Help
xxd(1)                                     General Commands Manual
NAME
xxd - make a hex dump or do the reverse.

SYNOPSIS
xxd -h[elp]
xxd [options] [infile [outfile]]
xxd -i[ever] [options] [infile [outfile]]

DESCRIPTION
xxd creates a hex dump of a given file or standard input. It can also convert a hex dump back to its original binary form. Like uuencode(1) and wudecode(1) it allows the transmission of binary data in a 'mail-safe' ASCII representation, but has the advantage of decoding to standard output. Moreover, it can be used to perform binary file patching.

OPTIONS
If no infile is given, standard input is read. If infile is specified as a '-' character, then input is taken from standard input. If no outfile is given (or a '-' character is in its place), results are sent to standard output.
Note that a "lazy" parser is used which does not check for more than the first option letter, unless the option is followed by a parameter. Spaces between a single option letter and its parameter are optional. Parameters to options can be specified in decimal, hexadecimal or octal notation. Thus -c8, -c 8, -c 0x10 and -c 010 are all equivalent.
-a | -autobskip
Toggle autobskip: A single '*' replaces NUL-lines. Default off.
-b | -bitsize
Switch to bits (binary digits) dump, rather than hex dump. This option writes octets as eight digits "1"s and "0"s instead of a normal hexadecimal dump. Each line is preceded by a line number in hexadecimal and followed by an ASCII (or EBCDIC) representation. The command line switches -p, -i do not work with this mode.
-c cols | -cols cols
Format: cols octets per line. Default 16 (-i: 32, -ps: 30, -b: 8). Max 256. No maximum for -ps. With -ps, 0 results in one long line of output.
-e | -escape
Capitalize variable names in C include file style, when using -i.

Manual page xxd(1) line 1 (press h for help or q to quit)

```

Step 10: Reviewing Ciphertext Again

- **Tool Used: Kali Linux Terminal**
- **Command Executed:**
 - `= cat cipher.txt`
- **Explanation:**
 - Displayed the contents of the cipher.txt again to manually cross-check changes after encryption.
- **Purpose:**
 - To prepare correct input for the decryption script.


```
kali@kali:~$ ssh -p 2231 krypton7@krypton.labs.overthewire.org
krypton7@krypton.labs.overthewire.org's password:
Welcome to OverTheWire!
If you find any problems, please report them to the #wargames channel on discord or IRC.
--[ Playing the games ]--
This machine might hold several wargames.
If you are playing "somegame", then:
```

Step 13: Exploring Krypton7 Directory and Reading README

- **Tool Used:** Kali Linux Terminal
- **Commands Executed:**

= cd /krypton/krypton7

= ls

= cat README

- **Explanation:**
 - Entered the /krypton/krypton7 directory.
 - Listed files and read the README for instructions about the new Level 7 challenge.
- **Purpose:**
 - To understand the next challenge setup after entering Level 7.

```
* gef (https://github.com/hugsy/gef) in /opt/gef/
* pwndbg (https://github.com/pwndbg/pwndbg) in /opt/pwndbg/
* gdbinit (https://github.com/gdbinit/gdbinit) in /opt/gdbinit/
* pwntools (https://github.com/Gallopsled/pwntools)
* radare2 (http://www.radare.org/)

--[ More information ]--

For more information regarding individual wargames, visit
http://www.overthewire.org/wargames/

For support, questions or comments, contact us on discord or IRC.

Enjoy your stay!

krypton7@bandit:~$ cd /krypton/krypton7
krypton7@bandit:/krypton/krypton7$ ls
README
krypton7@bandit:/krypton/krypton7$ cat README
Congratulations on beating Krypton!
krypton7@bandit:/krypton/krypton7$
```

