Krypton:

The level0 was easy all I had to do is just convert the encrypt base64 string to ASCII and I got the password to the next level.

For Level1 I looked at the file krypton2 and the hints were clear that it's a Caesar Cipher, so I looked at all the shift value and found the password of the next level.

For this level, I was given a program encrypt which encrypts the file with the same key for the cipher of password krypton3, so I followed the step of creating a different directory and run the encrypt program on A-Z and I got the key for the cipher, and I was able to decrypt the password, if I did the same thing for the previous stage, I would still get the password.

```
krypton2@krypton:/tmp/tmp.i2MOkLWEkW$ cat keyfile.dat
cat: keyfile.dat: Permission denied
krypton2@krypton:/tmp/tmp.i2MOkLWEkW$ echo "ABCDEFGHIJKLMNOPQRSTUVWXYZ" > text
krypton2@krypton:/tmp/tmp.i2MOkLWEkW$ cat text
ABCDEFGHIJKLMNOPQRSTUVWXYZ
krypton2@krypton:/tmp/tmp.i2MOkLWEkW$ /krypton/krypton2/encrypt text
krypton2@krypton:/tmp/tmp.i2MOkLWEkW$ ls
ciphertext  keyfile.dat  text
krypton2@krypton:/tmp/tmp.i2MOkLWEkW$ cat ciphertext
MNOPQRSTUVWXYZABCDEFGHIJKLkrypton2@krypton:/tmp/tmp.i2MOkLWEkW$
```

```
MNOPQRSTUVWXYZABCDEFGHIJKLkrypton2@krypton:/tmp/tmp.i2MOkLWEkW$
krypton2@krypton:/tmp/tmp.i2MOkLWEkW$ cat /krypton/krypton2/
encrypt      keyfile.dat  krypton3      README
krypton2@krypton:/tmp/tmp.i2MOkLWEkW$ cat /krypton/krypton2/krypton3 | tr "[A-Z]" "[M-ZA-L]"
AYCQYPGQCYQW
krypton2@krypton:/tmp/tmp.i2MOkLWEkW$ cat /krypton/krypton2/krypton3 | tr "[M-ZA-L]" "[A-Z]"
CAESARISEASY
krypton2@krypton:/tmp/tmp.i2MOkLWEkW$
```

For Leve4, the hint was given to do frequency analysis, so I used an online tool for frequency analysis and the result I found for 1 character and a group of 3 characters was:

		% calculated	% expected
S	155×	12.06%	
C	107×	8.33%	
Q	106×	8.25%	
J	102×	7.94%	
U	100×	7.78%	
B	87×	6.77%	
G	81×	6.3%	
N	74×	5.76%	
D	69×	5.37%	
Z	57×	4.44%	
V	56×	4.36%	
W	47×	3.66%	
Y	42×	3.27%	
T	32×	2.49%	
M	29×	2.26%	
X	29×	2.26%	
L	27×	2.1%	
K	25×	1.95%	
A	20×	1.56%	
E	17×	1.32%	
F	11×	0.86%	
O	7×	0.54%	
H	2×	0.16%	
I	2×	0.16%	
R	1×	0.08%	

#N : 25 Σ = 1285.0 Σ = 100.03 #N : 25

```
Y VQLUK SNSYM AVCUD SWCGS WCJCB GUBXI QNLCG EHMQV CJLQG W
QZZM NQZLW MNCGE DCUVC XSJCT SQGWC GJKBB XDCUX BNTSN JDSQ
J NCZQV ZBVVS
 QEMSU YMAVC UDSWJ DSXCN UJXBV CBQZB VVSZJ SWSWC JCBGB XD
CUW NQTQJ CZKBN FUJDQ JCGZV MWSWQ VVAMJ JKBBX JDSYV QLUGB
KNSZB EGCUS
WQUUD QFSUY SQNSU
```

★ PLAINTEXT EXPECTED LANGUAGE English ▾

TARGET CHARACTERS FOR FREQUENCY ANALYSIS
◉ LETTERS (A-Z) ONLY
○ LETTERS (A-Z) AND DIGITS (0-9) ONLY
○ DIGITS (0-9) ONLY
○ ONLY THESE CHARACTERS: αβγδε
○ ALL CHARACTERS (INCLUDING PUNCTUATION AND SYMBOLS)
★ STANDARDIZE LETTERS (IGNORE UPPER-LOWER CASE AND DIACRITICS) ☑

ITEMS TO ANALYZE
◉ EACH CHARACTER SEPARATELY
○ BIGRAMS (COUPLES OF 2 CHARACTERS)
○ TRIGRAMS (SET OF 3 CHARACTERS)
○ N-GRAMS N= 4
★ (FOR NGRAMS)   ◉ BLOCKS ANALYSIS (ABCDEF => AB,CD,EF)
                 ○ SLIDING WINDOW/OVERLAPPING (ABCDEF => AB,BC,CD,DE,EF)
★ KEEP WORDS BORDERS (ABC_DE ≠ ABCDE) ☐

ANALYSE TO PERFORM
★ ◉ CALCULATE FREQUENCIES
   ○ COUNT APPEARANCES
   ○ LIST MISSING LETTERS/NGRAMS
   ○ COUNT LONGEST REPEATS OF ANY N-GRAMS
   ○ COUNT N-GRAMS WITH REPEATED CHARACTERS
   ○ SUGGEST AN ALPHABETIC TRANSCRIPTION OF N-GRAMS (STATISTICALLY)

JDS	8×	1.87%	
ZQS	5×	1.17%	
UQN	5×	1.17%	
CBG	4×	0.93%	
ZCY	4×	0.93%	
DSN	4×	0.93%	
DQU	4×	0.93%	
BKG	4×	0.93%	
YDS	4×	0.93%	
QGW	3×	0.7%	
JCB	3×	0.7%	
JSN	3×	0.7%	
JBX	3×	0.7%	
QJC	3×	0.7%	
SUQ	3×	0.7%	
VSJ	3×	0.7%	
WCJ	3×	0.7%	
YVQ	3×	0.7%	
DCU	3×	0.7%	
CGJ	3×	0.7%	
SNS	3×	0.7%	
SXC	2×	0.47%	
ASN	2×	0.47%	
BVV	2×	0.47%	
KCJ	2×	0.47%	
ALQ	2×	0.47%	
ZSW	2×	0.47%	
DSY	2×	0.47%	
ZBT	2×	0.47%	

```
JU TQGLB XDCU
Y VQLUK SNSYM AVCUD SWCGS WCJCB GUBXI QNLCG EHMQV CJLQG W
QZZM NQZLW MNCGE DCUVC XSJCT SQGWC GJKBB XDCUX BNTSN JDSQ
J NCZQV ZBVVS
 QEMSU YMAVC UDSWJ DSXCN UJXBV CBQZB VVSZJ SWSWC JCBGB XD
CUW NQTQJ CZKBN FUJDQ JCGZV MWSWQ VVAMJ JKBBX JDSYV QLUGB
KNSZB EGCUS
WQUUD QFSUY SQNSU
```

★ PLAINTEXT EXPECTED LANGUAGE English ▾

TARGET CHARACTERS FOR FREQUENCY ANALYSIS
◉ LETTERS (A-Z) ONLY
○ LETTERS (A-Z) AND DIGITS (0-9) ONLY
○ DIGITS (0-9) ONLY
○ ONLY THESE CHARACTERS: αβγδε
○ ALL CHARACTERS (INCLUDING PUNCTUATION AND SYMBOLS)
★ STANDARDIZE LETTERS (IGNORE UPPER-LOWER CASE AND DIACRITICS) ☑

ITEMS TO ANALYZE
○ EACH CHARACTER SEPARATELY
○ BIGRAMS (COUPLES OF 2 CHARACTERS)
◉ TRIGRAMS (SET OF 3 CHARACTERS)
○ N-GRAMS N= 4
★ (FOR NGRAMS)   ◉ BLOCKS ANALYSIS (ABCDEF => AB,CD,EF)
                 ○ SLIDING WINDOW/OVERLAPPING (ABCDEF => AB,BC,CD,DE,EF)
★ KEEP WORDS BORDERS (ABC_DE ≠ ABCDE) ☐

ANALYSE TO PERFORM
★ ◉ CALCULATE FREQUENCIES
   ○ COUNT APPEARANCES
   ○ LIST MISSING LETTERS/NGRAMS
   ○ COUNT LONGEST REPEATS OF ANY N-GRAMS
   ○ COUNT N-GRAMS WITH REPEATED CHARACTERS
   ○ SUGGEST AN ALPHABETIC TRANSCRIPTION OF N-GRAMS (STATISTICALLY)

[LAUNCH ANALYSIS]

After this, I just started to guess as the most common letter in English is E so I replace E with S and the most common word is THE so I replace THE with JDS. And for doing a bunch of guesses the string started to make sense and I got the password for the next level.

For Leve5, it was a poly-alphabetic cipher called Vigenère cipher so after doing some research on the cipher I got an idea on how to decrypt it, this video really helped full in understanding the cipher decryption trick (https://www.youtube.com/watch?v=LaWp_Kq0cKs), I wrote a python script which only reads the letter give as input from the key so I can do frequency analysis on it as it has been decrypted with the same key. Doing this I can figure out the letter used to decrypt, and repeating this I got the entire key.

```python
C: > Users > UZAIF SHAIKH > Documents > Python Scripts > vigenere.py > ...
1    with open("text1.txt","r") as file:
2        string1 = file.readlines()
3    length = int(input("len: "))
4    index = int(input("index: "))
5    letter = ""
6    for i in range(index,len(string1)):
7        c1 = string1[i].split()
8        for j in range(len(c1)):
9            letter += c1[j]
10
11   #print(letter)
12   for i in range(0,len(letter),length):
13       print(letter[i],end="")
```

Results

Occurency and Frequency Analysis
1-grams
% calculated | % expected

↑↓	↑↓	↑↓	↑↓
J	37×	15.29%	
S	24×	9.92%	
Y	22×	9.09%	
T	20×	8.26%	
F	18×	7.44%	
W	17×	7.02%	
M	16×	6.61%	
I	14×	5.79%	
N	12×	4.96%	
K	11×	4.55%	
X	9×	3.72%	
Z	7×	2.89%	
B	7×	2.89%	
Q	6×	2.48%	
L	5×	2.07%	
P	4×	1.65%	
D	3×	1.24%	
H	3×	1.24%	
R	3×	1.24%	
G	3×	1.24%	
A	1×	0.41%	

#N : 21 Σ = 242.00 Σ = 100.00 #N : 21

FREQUENCY ANALYSIS (ADVANCED)

★ TEXT TO ANALYZE

YIYWNQRLYTRHYDJTWZSLNNHTMJJYFNYIJJSLWNMFXBBKXIMJTBMIYJJNTYB
WKWWLFGWISJSZYSYPJNFJQFWTYWKJJMMNSYWKYSAYMTSQZJRFDMKXFJJPKF
STTTJMBMJDSQIJPFJTSJWJPJIKXISJFFYXMQMYIMZYFSJWJNTWGJYGZTMTY
SFFJTWJQBSFSJSJIJJNKWSXZYZKXMSSIFTXSSKTJTYWMYKLTISNJFITWIXN
BSJHJF

★ PLAINTEXT EXPECTED LANGUAGE   English

TARGET CHARACTERS FOR FREQUENCY ANALYSIS

◉ LETTERS (A-Z) ONLY
○ LETTERS (A-Z) AND DIGITS (0-9) ONLY
○ DIGITS (0-9) ONLY
○ ONLY THESE CHARACTERS: αβγδε
○ ALL CHARACTERS (INCLUDING PUNCTUATION AND SYMBOLS)
★ STANDARDIZE LETTERS (IGNORE UPPER-LOWER CASE AND DIACRITICS) ☑

ITEMS TO ANALYZE

◉ EACH CHARACTER SEPARATELY
○ BIGRAMS (COUPLES OF 2 CHARACTERS)
○ TRIGRAMS (SET OF 3 CHARACTERS)
○ N-GRAMS N= 4
★ (FOR NGRAMS)   ◉ BLOCKS ANALYSIS (ABCDEF => AB,CD,EF)
            ○ SLIDING WINDOW/OVERLAPPING (ABCDEF => AB,BC,CD,DE,EF)
★ KEEP WORDS BORDERS (ABC_DE ≠ ABCDE) ☐

ANALYSE TO PERFORM
★   ◉ CALCULATE FREQUENCIES

Vigenere 🖊 FREKEY
(Alphabet (26) ABCDEFGHIJKLMNOPQRSTUVWXYZ)

CLEARTEXT

Vigenere Cipher - dCode
Tag(s) : Poly-Alphabetic Cipher

Share

VIGENERE DECODER

★ VIGENERE CIPHERTEXT

HCIKVRJOX

PARAMETERS

★ PLAINTEXT LANGUAGE   English
★ ALPHABET   ABCDEFGHIJKLMNOPQRSTUVWXYZ

AUTOMATIC DECRYPTION

DECRYPTION METHOD

◉ KNOWING THE KEY/PASSWORD: FREKEY
○ KNOWING THE KEY-LENGTH/SIZE, NUMBER OF LETTERS: 3
○ KNOWING ONLY A PARTIAL KEY: KE?
○ KNOWING A PLAINTEXT WORD: CODE
○ COMMON-WORDS DICTIONARY ATTACK FOR KEY
○ VIGENERE CRYPTANALYSIS (KASISKI'S TEST)

DECRYPT

So for Level6, the key length was not given so I had to follow the video (https://www.youtube.com/watch?v=LaWp_Kq0cKs), about the Vigenère cipher and I wrote a python which counts the coincidence, and use the output from the python program I found 9 and its factor have a higher rate of co-incidence for this I need to ignore 1,2, and 3 as they are the factor for most of the number.

```
C: > Users > UZAIF SHAIKH > Documents > Python Scripts > 🐍 vigenere.py > ...
  1   import string
  2   with open("text1.txt","r") as file:
  3       string1 = file.readlines()
  4   #print(string1)
  5   letter = ""
  6   #print(string1[1])
  7   for i in range(len(string1)):
  8       #print(string1[i])
  9       c1 = string1[i].split()
 10       #print(c1)
 11       for j in range(len(c1)):
 12           letter += c1[j]
 13
 14   #print(letter)
 15   find = []
 16   for i in range(1,len(letter)):
 17       count = 0
 18       tmp = letter[i:len(letter)-i]
 19       for j in range(len(tmp)):
 20           if letter[i] == tmp[j]:
 21               count += 1
 22       if count != 0:
 23           find.append(count)
 24
 25   for i in set(find):
 26       print(i,": ",find.count(i))
 27
```

```
PS C:\Users\UZAIF SHAIKH\Documents\Python Scripts> python .\vigenere.py
1 :   14
2 :   10
3 :   15
4 :   16
5 :   6
6 :   13
7 :   10
8 :   17
9 :   15
10 :   18
11 :   15
12 :   12
13 :   10
14 :   14
15 :   12
16 :   13
17 :   14
18 :   10
19 :   15
20 :   14
21 .   14
```

And after guessing the key length I followed the same method from the previous level and got the key (this took me a lot of time achy) and the key was KEYLENGTH, and I used the online tool to get decrypt the cipher.

Results

Vigenere 🔑 KEYLENGHT

(Alphabet (26) ABCDEFGHIJKLMNOPQRSTUVWXYZ)

RANDOM

KILL THE GOBLIN!

HERO WARS

TRY

Vigenere Cipher - dCode

Tag(s) : Poly-Alphabetic Cipher

Share

VIGENERE DECODER

★ VIGENERE CIPHERTEXT

BELOSZ

PARAMETERS

★ PLAINTEXT LANGUAGE   English ▾
★ ALPHABET   ABCDEFGHIJKLMNOPQRSTUVWXYZ

AUTOMATIC DECRYPTION

DECRYPTION METHOD

◉ KNOWING THE KEY/PASSWORD: KEYLENGHT
○ KNOWING THE KEY-LENGTH/SIZE, NUMBER OF LETTERS: 3
○ KNOWING ONLY A PARTIAL KEY: KE?
○ KNOWING A PLAINTEXT WORD: CODE
○ COMMON-WORDS DICTIONARY ATTACK FOR KEY
○ VIGENERE CRYPTANALYSIS (KASISKI'S TEST)

DECRYPT

See also: **Beaufort Cipher** — **Caesar Cipher**

For the last level, I read the readme file and found out that that have been using XOR to encrypt the plaintext so to find how the key, there is a property of XOR that A ^ B = C and if A ^ C = B so using strings of A and to encrypt with give us the key and after getting the key I just ran the Vigenère cipher decrypt and got the password.

```
krypton6@krypton:/tmp/tmp.1DH00Fbncv$ python -c "print('A'*80)" > text
krypton6@krypton:/tmp/tmp.1DH00Fbncv$ cat text
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
krypton6@krypton:/tmp/tmp.1DH00Fbncv$ /krypton/krypton6/encrypt6 text key
krypton6@krypton:/tmp/tmp.1DH00Fbncv$ cat key
EICTDGYIYZKTHNSIRFXYCPFUEOCKRNEICTDGYIYZKTHNSIRFXYCPFUEOCKRNEICTDGYIYZKTHNSIRFXYkrypton6@krypton:/tmp/tmp.1DH00Fbncv$ |
```

KEY:

<mark>EICTDGYIYZKTHNSIRFXYCPFUEOCKRN</mark>

EICTDGYIYZKTHNSIRFXYCPFUEOCKRN

EICTDGYIYZKTHNSIRFXY

(Alphabet (26) ABCDEFGHIJKLMNOPQRSTUVWXYZ)

<mark>LFSRISNOTRANDOM</mark>

★ VIGENERE CIPHERTEXT

PNUKLYLWRQKGKBE

PARAMETERS

★ PLAINTEXT LANGUAGE English ⌄

★ ALPHABET ABCDEFGHIJKLMNOPQRSTUVWXYZ

AUTOMATIC DECRYPTION

DECRYPTION METHOD

◉ KNOWING THE KEY/PASSWORD: EICTDGYIYZKTHNSI

◯ KNOWING THE KEY-LENGTH/SIZE, NUMBER OF LETTERS: 3