SMS Ex01

**Basic cryptanalysis in practice**

# 1.1) Basic cryptanalysis in practice

## The deciphered text and the corresponding key

Deciphered Text:

relational databases, by design, focus on preventing redundancy within the data pool. this aspect becomes especially important in terms of watermarking, since redundancy is one of the main prerequisites in order to embed additional information which implicates a notable increase in the data pool’s entropic properties. utilizing a highly entropic carrier medium therefore dramatically minimizes the potential possibilities of hiding additional information without seriously interfering with the structural, quantitative and/or qualitative consistency thereof. as a result, embedding additional information in a low redundancy carrier signal using traditional steganographic techniques either increases the size of the original information (since redundant information we could make use of for this purpose is rare), or may heavily distort the same due to indispensable changes which have to be made.

Corresponding key

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

## How to crack the cipher

1. Read the downloaded cipher letter by letter in order to retrive the letter frequency of the ciphered text

{'D': 89, 'Q': 83, 'F': 66, 'Z': 66, 'E': 62, 'X': 53, 'U': 52, 'C': 45, 'M': 32, 'J': 28, 'W': 28, 'O': 26, 'Y': 22, 'S': 21, 'G': 18, 'L': 15, 'A': 14, 'I': 11, 'R': 9, 'T': 8, 'V': 5, 'N': 4, 'H': 3, 'B': 2}

1. Search for patterns which could define the language. In this case there were several 1 letter words which indicates the english language
2. Check those 1 letter words and match the cipher letter with the rest of the text in order to determine if it is either an ‘A’ or a ‘I’. Use the letter frequency.
3. Search for 2 letter words and look at similarities between them
4. Search for 3 letter words and look for reoccurring letter patterns especially ‘th’ and ‘ing’
5. The rest is trial and error

## Source Code

*import* io  
  
  
*def* read\_file(file\_name: *str*):  
 num\_of\_chars = {}  
 file = io.open(file\_name, mode="r", encoding="utf-8")  
  
 chiffre: *str* = file.read()  
 file = io.open("Chiffre.txt", mode="r", encoding="utf-8")  
  
 *while* 1:  
 char = file.read(1)  
 *if not* char:  
 *break  
  
 if ord*(char) > 123 *or ord*(char) < 65:  
 *continue  
 elif* char *not in* num\_of\_chars.keys():  
 num\_of\_chars[char] = 1  
 *else*:  
 num\_of\_chars[char] = num\_of\_chars[char] + 1  
  
 file.close()  
  
 *print*(chiffre)  
 *print*(*dict*(*sorted*(num\_of\_chars.items(), key=*lambda* item: item[1], reverse=*True*)))  
  
 plain = chiffre  
  
 plain = plain.replace('W', 'h')  
 plain = plain.replace('T', 'w')  
 plain = plain.replace('Z', 't')  
 plain = plain.replace('E', 'n')  
 plain = plain.replace('Q', 'e')  
 plain = plain.replace('R', 'b')  
 plain = plain.replace('U', 'r')  
 plain = plain.replace('S', 'u')  
 plain = plain.replace('C', 's')  
 plain = plain.replace('Y', 'm')  
 plain = plain.replace('I', 'y')  
 plain = plain.replace('D', 'i')  
 plain = plain.replace('M', 'd')  
 plain = plain.replace('O', 'c')  
 plain = plain.replace('X', 'o')  
 plain = plain.replace('F', 'a')  
 plain = plain.replace('L', 'f')  
 plain = plain.replace('J', 'l')  
 plain = plain.replace('A', 'g')  
 plain = plain.replace('G', 'p')  
 plain = plain.replace('B', 'k')  
 plain = plain.replace('N', 'q')  
 plain = plain.replace('H', 'z')  
 plain = plain.replace('V', 'v')  
  
 *print*(plain)  
  
  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 filename = "Chiffre.txt"  
 read\_file(filename)

# 1.2) Playfair cipher

Siehe source code in .zip

# 1.3) Geocaching

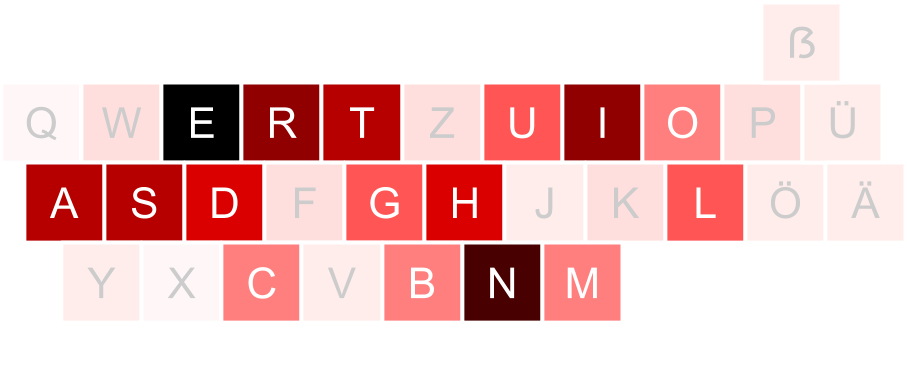
ROT13:

|  |  |
| --- | --- |
| Ciphered | Plain |
| uggc://pelcgbby-bayvar.qr | http://cryptool-online.de |
| Cnexra vfg orv qra Urnqrexbbeqvangra tüafgvt. | Parken ist bei den Headerkoordinaten günstig. |
| Ovggr frvq orvz Ybttra ibefvpugvt. | Bitte seid beim Loggen vorsichtig. |

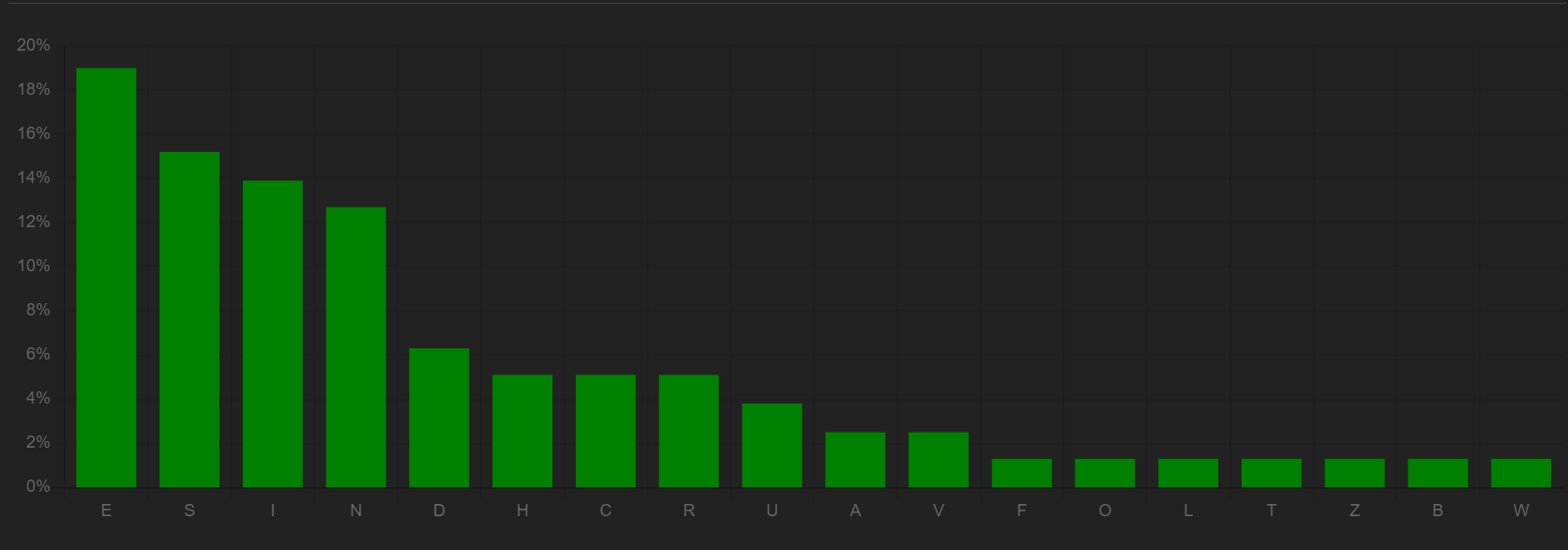
Coordinates ciphered:

faovheendiiensidslsiictiirineneveeescishneshseeazendnuuuisrrishescdcbrwsenndnse

German letter distribution



Letter distribution of Coordinates



As the coordinate’s distribution of letters is similar to the letter distribution of the german language in general it is most likely that no substitution cipher was use but instead a shuffling cipher.

As of this the railfence cipher deciphered the coordinates:

FINDE DAS SCHLOSS BEI VIER ACHT ZWEI EINS DREI NEUN NEUN UND EINS VIER DREI EINS SECHS EINS SECHS

Finde das Schloss bei 48213991 und 1431616

# 1.4) Cracking one-time pads

Start by using the information with which sentence the second plain text starts.

We can try to create a mapping on that and search for similarities in both texts.

Otherwise if we know that both plain texts were ciphered with the same one time pad (especially if only letters were used), we can XOR those texts and search for similarities.

1st text:

The source code for android is open source and it is developed in private by Google, with the source code released publicly when a new version of android is released. Google publishes most of the code including network and telephony stacks under the non-copyleft apache license version two, which allows modification and redistribution. The license does not grant rights to the android trademark, so device manufactures and wireless carriers have to license it

2nd text:

SELinux is a mandatory access control mechanism for the Linux kernel, implemented as a Linux security module. The Linux Security Modules (LSM) framework allows third-party access control mechanisms to be linked into the kernel and to modify the default DAC implementation. LSM is implemented as a series of security function hooks , upcalls, and related data structures that are integrated into the various modules of the Linux kernel responsible for access control.