Cryptology for IoT

Modules M4, M7, M9
Session of 27th April, 2022.

M4.3 Briefing to the session
M4.4 Introduction to the ciphers: Substitution , Transposition and mixed ciphers
M4.5 Methodology using Cryptool

Prof.: Guillermo Botella

Cryptology for IoT

Modules M4, M7, M9
Session of 27th April, 2022.

M4.3 Briefing to the session

M4.4 Introduction to the ciphers: Substitution , Transposition and mixed ciphers

M4.5 Methodology using Cryptool

Prof.: Guillermo Botella

M4.3 Briefing of today

- Starting with basic Cryptography and Cryptoanalysis
  - Slides and supplementary videos
- We go to the rooms. Practical Session I.
  - Assignments (They will be specified when we start).
  - Work in groups (Same than usual)
- First quiz at Socrative (around 25 questions)
  - Room number will be specified when we start
  - Individual work

Cryptology for IoT

Modules M4, M7, M9
Session of 27th April, 2022.

M4.3 Briefing to the session

M4.4 Introduction to the ciphers: Substitution , Transposition and mixed ciphers

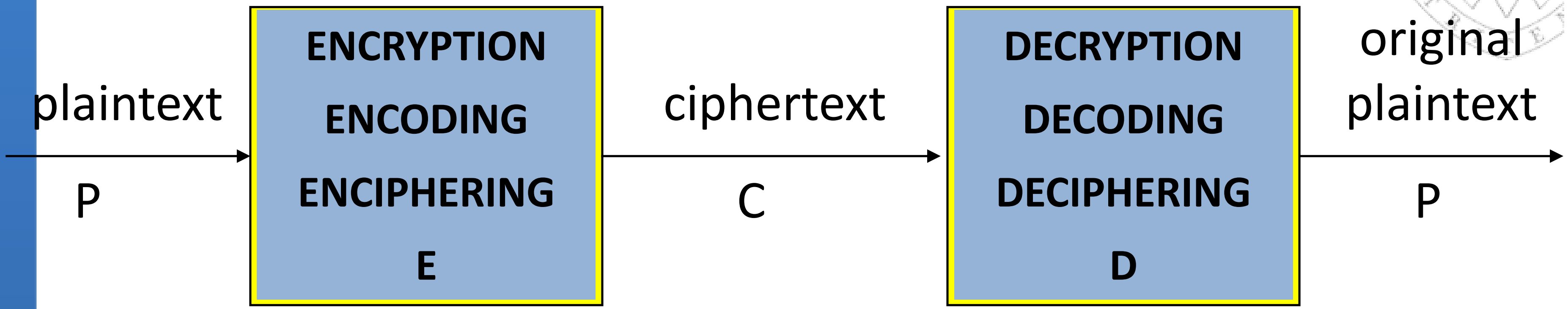
M4.5 Methodology using Cryptool

Prof.: Guillermo Botella

Organization of the M4.4

- Notation
- Ciphers (families)
- Substitution ciphers
  - Monoalphabetic Substitution
  - Polyalphabetic Substitution
- Transposition Ciphers
- Mixed Ciphers

Formal Notation



- C = E(P)
- P = D(C)

E – encryption rule/algorithm

D – decryption rule/algorithm

- We need a cryptosystem, where:
  - P = D(C)= D(E(P))
    - i.e., able to get the original message back

Representing Characters

- Letters (uppercase only) represented by numbers 0-25 (modulo 26).

A B C D ... X Y Z

0 1 2 3 ... 23 24 25

- Operations on letters:

A + 2 = C

X + 4 = B (circular!)

...

Basic Types of Ciphers

- Substitution ciphers
  - Letters of P replaced with other letters by E
- Transposition (permutation) ciphers
  - *Order* of letters in P rearranged by E
- Product ciphers
  - E "=" E_1 "+" E_2 "+" ... "+" E_n
    - Combine two or more ciphers to enhance the security of the cryptosystem

Substitution Ciphers

- **Substitution Ciphers:**
  - **Letters of P replaced with other letters by E**

The Caesar Cipher (1)

- $c_i=E(p_i)=p_i+3 \bmod 26$ (*26* letters in the English alphabet)

  Change each letter to the third letter following it (circularly)

  A → D, B → E, ... X → A, Y → B, Z → C

- Can represent as a permutation π: $\pi(i) = i+3 \bmod 26$

  $\pi(0)=3, \pi(1)=4, \ldots,$

  $\pi(23)=26 \bmod 26=0, \pi(24)=1, \pi(25)=2$

- Key = 3, or key = 'D' (because D represents 3)

The Caesar Cipher (2)

- Example
  - P (plaintext): HELLO WORLD
  - C (ciphertext): khoor zruog

- Caesar Cipher is a monoalphabetic substitution cipher (= simple substitution cipher)
  One key is used
  One letter substitutes the letter in P

Attacking a Substitution Cipher

- Exhaustive search
  - If the key space is small enough, try all possible keys until you find the right one
  - Cæsar cipher has 26 possible keys
    from A to Z OR: from 0 to 25
- Statistical analysis (attack)
  - Compare to so called 1-gram (unigram) model of English
    - 1-gram: It shows frequency of (single) characters in English
  - The longer the C, the more effective statistical analysis would be

1-grams (Unigrams) for English

a	0.080	h	0.060	n	0.070	t	0.090
b	0.015	i	0.065	o	0.080	u	0.030
c	0.030	j	0.005	p	0.020	v	0.010
d	0.040	k	0.005	q	0.002	w	0.015
e	0.130	l	0.035	r	0.065	x	0.005
f	0.020	m	0.030	s	0.060	y	0.020
g	0.015					z	0.002

Statistical Attack – Step 1

- Compute frequency f(c) of each letter c in ciphertext
- Example: c = ‘khoor zruog’
  - 10 characters: 3 * ‘o’, 2 * ‘r’, 1 * {k, h, z, u, g}
  - f(c):

    f(g)=0.1 f(h)=0.1 f(k)=0.1 f(o)=0.3 f(r)= 0.2

    f(u)=0.1 f(z)=0.1 $f(c_i)$ = 0 for any other c_i

- Apply 1-gram model of English
  - Frequency of (single) characters in English
  - 1-grams on previous slide

Statistical Analysis – Step 2

- phi $\varphi(i)$ - correlation of frequency of letters *in ciphertext* with frequency of corresponding letters *in English* —for key i
- For key i: $\varphi(i) = \Sigma_{0 \le c \le 25}\, f(c) * p(c - i)$
  - *c* representation of character (a-0, ..., z-25)
  - f(c) is frequency of letter c in ciphertext C
  - *p*(*x*) is frequency of character *x* in English
  - Intuition: sum of probabilities for words in P, if i were the key

c is a letter in ciphertext thus c-i is the letter in plaintext.

- Example: C = ‘khoor zruog’ (P = ‘HELLO WORLD’)

f(c): f(g)=0.1, f(h)=0.1, f(k)=0.1, f(o)=0.3, f(r)=0.2, f(u)=0.1, f(z)=0.1

c: g - 6, h - 7, k - 10, o - 14, r - 17, u - 20, z - 25

$$\varphi(i) = 0.1p(6 - i) + 0.1p(7 - i) + 0.1p(10 - i) + 0.3p(14 - i) + 0.2p(17 - i) + 0.1p(20 - i) + 0.1p(25 - i)$$

Statistical Attack – Step 2a (Calculations)

- Correlation φ(i) for 0≤ i ≤25

i	φ(*i*)	*i*	φ(*i*)	*i*	φ(*i*)	*i*	φ(*i*)
0	0.0482	7	0.0442	13	0.0520	19	0.0315
1	0.0364	8	0.0202	14	0.0535	20	0.0302
2	0.0410	9	0.0267	15	0.0226	21	0.0517
3	0.0575	10	0.0635	16	0.0322	22	0.0380
4	0.0252	11	0.0262	17	0.0392	23	0.0370
5	0.0190	12	0.0325	18	0.0299	24	0.0316
6	0.0660					25	0.0430

Statistical Attack – Step 3 (The Result)

- Most probable keys (largest $\varphi(i)$ values):
  - *$i = 6$, $\varphi(i) = 0.0660$*
    - *plaintext EBIIL TLOLA*
  - *$i = 10$, $\varphi(i) = 0.0635$*
    - *plaintext AXEEH PHKEW*
  - *$i = 3$, $\varphi(i) = 0.0575$*
    - *plaintext HELLO WORLD*
  - *$i = 14$, $\varphi(i) = 0.0535$*
    - *plaintext WTAAD LDGAS*
- Only English phrase is for i = 3
  - *That's the key (3 or 'D') – code broken*

Caesar's Problem

- Conclusion: Key is too short
  - 1-char key – monoalphabetic substitution
    - Can be found by exhaustive search
    - Statistical frequencies not concealed well by short key
      - They look too much like 'regular' English letters
- Solution: Make the key longer
  - n-char key ($n \geq 2$) – polyalphabetic substitution
    - Makes exhaustive search much more difficult
    - Statistical frequencies concealed much better
      - Makes cryptanalysis harder

Other Substitution Ciphers

n-char key:

- Polyalphabetic substitution ciphers
- Vigenere Tableaux cipher

Polyalphabetic Substitution - Examples

- Flatten (difuse) *somewhat* the frequency distribution of letters by combining high and low distributions

- Example – 2-key substitution:

	A	B	C	D	E	F	G	H	I	J	K	L	M
Key1:	a	d	g	j	m	p	s	v	y	b	e	h	k
Key2:	n	s	x	c	h	m	r	w	b	g	l	q	v

	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Key1:	n	q	t	w	z	c	f	i	l	o	r	u	x
Key2:	a	f	k	p	u	z	e	j	o	t	y	d	i

- **Question**:

  How Key1 and Key2 were defined?

Polyalphabetic Substitution - Examples

- ...
- Example:

	A	B	C	D	E	F	G	H	I	J	K	L	M
Key1:	a	d	g	j	m	p	s	v	y	b	e	h	k
Key2:	n	s	x	c	h	m	r	w	b	g	l	q	v

	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Key1:	n	q	t	w	z	c	f	i	l	o	r	u	x
Key2:	a	f	k	p	u	z	e	j	o	t	y	d	i

- Answer:

  Key1 – start with ‘a’, skip 2, take next,

  skip 2, take next letter, ... (circular)

  Key2 - start with ‘n’ (2nd half of alphabet), skip 4,

  take next, skip 4, take next, ... (circular)

Polyalphabetic Substitution - Examples

– Example:

	A	B	C	D	E	F	G	H	I	J	K	L	M
Key1:	a	d	g	j	m	p	s	v	y	b	e	h	k
Key2:	n	s	x	c	h	m	r	w	b	g	l	q	v

	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Key1:	n	q	t	w	z	c	f	i	l	o	r	u	x
Key2:	a	f	k	p	u	z	e	j	o	t	y	d	i

– **Plaintext: TOUGH STUFF**

– **Ciphertext: ffirv zfjpm**

use n (=2) keys in turn for consecutive P chars in P

- Note:
  - Different chars mapped into the same one: **T, O → f**
  - Same char mapped into different ones: **F → p, m**
  - '**f**' most frequent in C (0.30); in English: f(**f**) = 0.02 << f(**e**) = 0.13

Module 4: Cryptology for IoT

Vigenère Tableaux (1)

Note: Row A – shift 0 (a->a)
Row B – shift 1 (a->b)
Row C – shift 2 (a->c)
...
Row Z – shift 25 (a->z)

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	p
A	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	0
B	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	1
C	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	2
D	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	3
E	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	4
F	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	5
G	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	6
H	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	7
I	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	8
J	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	9
K	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	10
L	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	11
M	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	12
N	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	13
O	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	14
P	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	15
Q	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	16
R	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	17
S	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	18
T	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	19
U	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	20
V	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	21
W	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	22
X	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	23
Y	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	24
Z	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	25

Sec

Vigenère Tableaux (2)

- Example

Key:

EXODUS

Plaintext P:

YELLOW SUBMARINE FROM YELLOW RIVER

Extended keyword (re-applied to mimic words in P):

YELLOW SUBMARINE FROM YELLOW RIVER

EXODUS EXODUSEXO DUSE XODUSE XODUS

Ciphertext:

cbxoio wlppujmks ilgq vsofhb owyyj

Vigenère Tableaux (3)

- Example

  …

  Extended keyword (re-applied to mimic words in P):

  YELLOW SUBMARINE FROM YELLOW RIVER

  EXODUS EXODUSEXO DUSE XODUSE XODUS

  Ciphertext:

  cbzoio wlppujmks ilgq vsofhb owyyj

- Answer:

  c from P indexes row

  c from extended key indexes column

  e.g.: row Y and column e → 'c'

  row E and column x → 'b'

  row L and column o → 'z'

  …

Transposition Ciphers (1)

- Rearrange letters in plaintext to produce ciphertext
- Example 1a and 1b: Columnar transposition
  - Plaintext: **HELLO WORLD**
  - Transposition onto: (a) 3 columns:

    HEL
    LOW
    ORL
    DXX **XX** - padding

    (b) onto 2 columns:

    HE
    LL
    OW
    OR
    LD
  - Ciphertext (read column-by column):

    (a) **hlodeorxlwlx** (b) **hloolelwrd**
  - What is the key?
    - Number of columns: (a) key = 3 and (b) key = 2

Transposition Ciphers (2)

- Example 2: Rail-Fence Cipher
  - Plaintext: **HELLO WORLD**
  - Transposition into 2 rows (rails) column-by-column:

    HLOOL

    ELWRD
  - Ciphertext: **hloolelwrd** (Does it look familiar?)
  - What is the key?
    - Number of rails key = 2

Product Ciphers

- A.k.a. combination ciphers
- Built of multiple blocks, each is:
  - Substitution

or:

  - Transposition

- Example: two-block product cipher
  - $E_2(E_1(P, K_{E1}), K_{E2})$
- Product cipher might *not* necessarily be stronger than its individual components used separately!
  - Might not be even as strong as individual components

Identifying the type of a cipher

- Not always possible without further knowledge about the cipher's origin and background
  - – Voynich Manuscript – a book of the 15th century encrypted and written using an unknown alphabet
- To identify the type of the cipher we have seen to check out:
  - – Frequency test component: visualizes the letter distribution of a given text
  - – Friedman test component (kappa test)

Friedman Test (Kappa Test)

- The “**Frequency test” (a.k.a Kappa test)** component visualizes the letter distribution of a given texts
- It uses the index of coincidence, which measures the unevenness of the cipher letter frequencies to break the cipher
- The key length of a polyalphabetic cipher can be estimated by knowing two issues:
  - the probability (Kp) that any two randomly chosen source-language letters are the same (around 0.067 for English)
  - the probability (Kr) of a coincidence for a uniform random selection from the alphabet (1/26 = 0.0385 for English)

Friedman Test (Kappa Test)

- The key length can be estimated as the following:

$$\frac{\kappa_p - \kappa_r}{\kappa_o - \kappa_r}$$

- From the observed coincidence rate (*Ko*):

$$\kappa_o = \frac{\sum_{i=1}^{c} n_i(n_i - 1)}{N(N-1)}$$

Friedman Test (Kappa Test)

- Observed coincidence rate (*Ko*):

$$\kappa_o = \frac{\sum_{i=1}^{c} n_i(n_i - 1)}{N(N-1)}$$

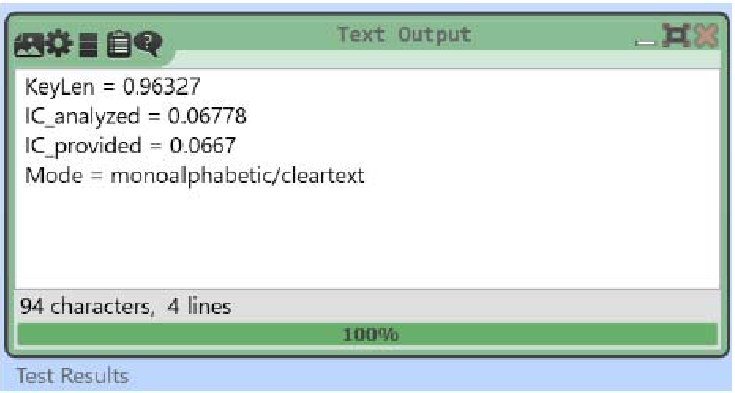
- ***c*** is the size of the alphabet (26 for English), ***N*** is the length of the text and n_i to n_c are the observed ciphertext letter frequencies, as integers

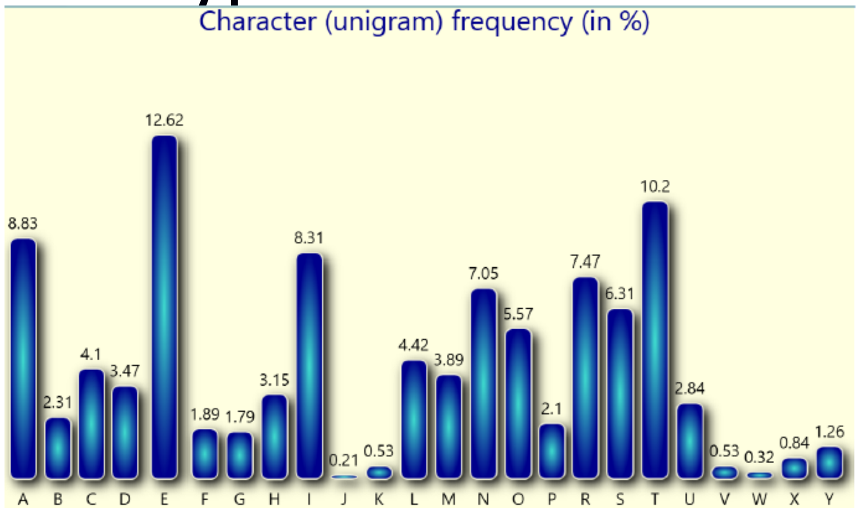
- That is, however, only an approximation; its accuracy increases with the length of the text

- It would, in practice, be necessary to try various key lengths that are close to the estimate

Friedman Test (Kappa Test)

- Computational tools to do (CT2):
- IC is the probability of two randomly drawn letters out of a text to be identical
- Useful to differentiate between plaintext (or transposed or monoalphabetic substituted text) and polyalphabetic encrypted texts





Friedman Test (Kappa Test)

- I.e example:
  - For English texts IC is 6.6%
  - For German texts IC is 7.8%

- Using **simple monoalphabetic encryption**, where a single letter is replaced by another letter, **does not change the IC of the text**

Friedman Test (Kappa Test)

- **Same applies to all transposition ciphers**, since these do not change the text frequencies

- **Homophone substitution** also aims at changing the letter distribution of a text to become the uniform distribution, but **here the IC is about 1/n** , where n is the amount of different symbols in the text

Friedman Test (Kappa Test)

- Thus, having an IC close to **6.6%** indicates that we have either a plaintext, a **monoalphabetic substituted text**, or a **transposed text**
- On the other hand, having an IC close to **3.8%** indicates that we have a **polyalphabetic encrypted text**
- Clearly, the IC is more accurate having long ciphertexts
- Identification of homophone ciphers can be done by counting the number of different used letters or symbols
- If the number is above the expected alphabet size, it is probably a **homophone substitution**

Criteria for "Good" Ciphers

- "Good" depends on intended application
  - Substitution
    - C hides chars of P
    - If > 1 key, C dissipates high frequency chars
  - Transposition
    - C scrambles text => hides n-grams for n > 1
  - Product ciphers
    - Can do all of the above
  - What is more important for your app?

    What facilities available to sender/receiver?
    - E.g., no supercomputer support on the battlefield

Criteria for "Good" Ciphers

- **Commercial Principles of Sound Encryption Systems**
  1. Sound mathematics
     - Proven vs. not broken so far
  2. Verified by expert analysis
     - Including outside experts
  3. Stood the test of time
     - Long-term success is not a guarantee
       - Still. Flows in many E's discovered soon after their release

- Examples of popular commercial encryption (We will see them at next M7 module):
  – DES / RSA / AES

DES = Data Encryption Standard
RSA = Rivest-Shamir-Adelman
AES = Advanced Encryption Standard (rel. new)

Cryptology for IoT

Modules M4, M7, M9
Session of 27th April, 2022.

M4.3 Briefing to the session
M4.4 Introduction to the ciphers: Substitution , Transposition and mixed ciphers
M4.5 Methodology using Cryptool

Prof.: Guillermo Botella

Step-by-step approach methodology for classical ciphers

- **First step →** Make the cipher processable for CT2, so we create a digital transcription of the ciphertext

- **Second step →** Identify the type of the cipher

- **Third step →** Try to break the cipher

Step-by-step approach methodology: i) Create a transcription

- There are two ways to create a transcription of a ciphertext for CT2
- The first method is to manually assign to each ciphertext symbol a letter by hand outside of CT2, e.g. with Windows Notepad.
  - The transcription is saved as a simple text file
  - This file can be loaded into CT2 by using the FileInput component and then be processed further

Step-by-step approach methodology: i) Create a transcription

- The second method to create a transcription uses the CT2 component "*Transcriptor*"
- With the transcriptor, a user can load a picture, e.g. a scan of a document

Step-by-step approach methodology: i) Create a transcription

- Finally, the transcriptor is able to output the complete transcription.
- It supports the user in two different ways:
  - It automatically guesses, which symbol the user just had marked by showing the most likely symbols
  - It can be set to semiautomatic mode. In semi-automatic mode, it automatically marks all other symbols that are similar to the one just marked by the user

Step-by-step approach methodology: ii) Identify the cipher

- Several analysis:

- **Text frequency analysis**

- **Friedman test analysis**

Step-by-step approach methodology: ii) Identify the cipher

- **Text frequency analysis.**

For that, CT2 contains a Frequency Test component

It can be configured to show:

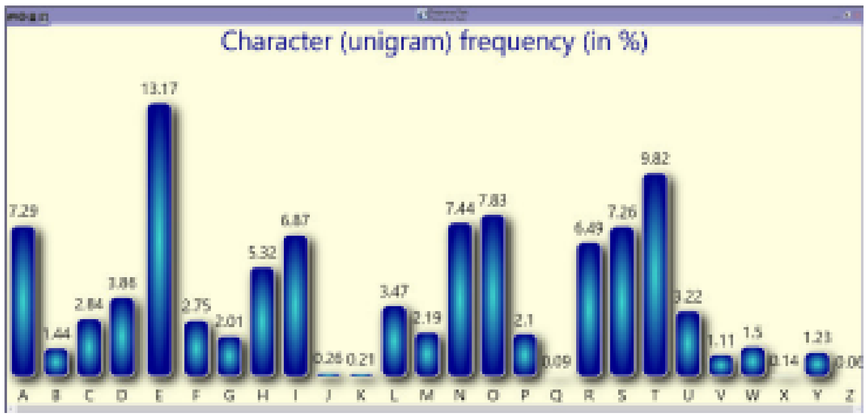
– unigram distribution

– bigram distribution

– etc.

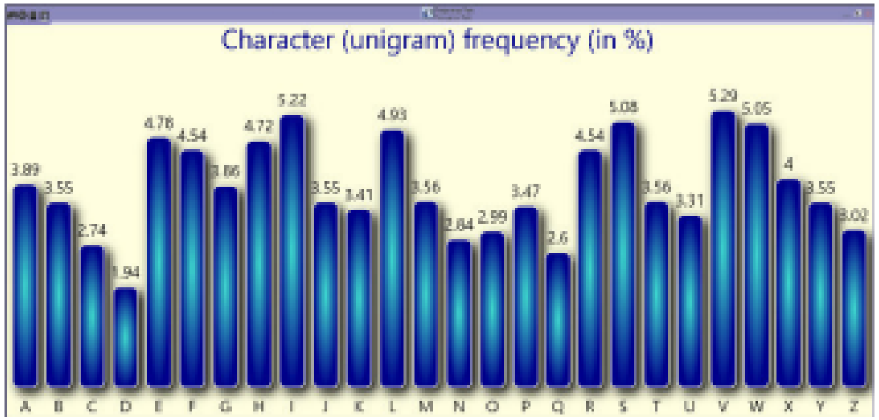
Step-by-step approach methodology: ii) Identify the cipher

- Example: Distribution of plain text. ("*The Declaration of Independence*" of the US)
- The text follows the letter distribution of the English., i.e. the ’E’ is the most frequent letter, the letters ’X’, ’Q’, and ’Z’ are very rare



Step-by-step approach methodology: ii) Identify the cipher

- Distribution of cipher text. Same text encrypted with Vigenere cipher
- Here, all letters are more or less equally distributed, showing the cryptanalyst that it is possibly a polyalphabetic substitution cipher

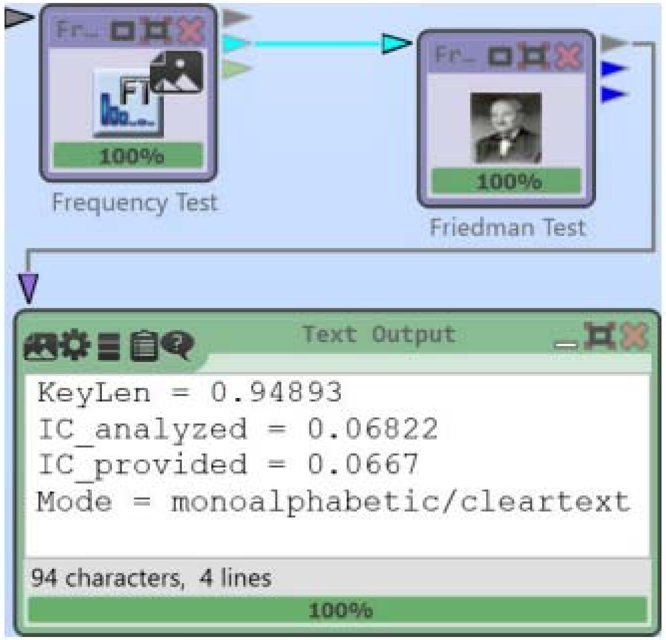


Step-by-step approach methodology: ii) Identify the cipher

- **Friedman test analysis.**

For that, CT2 contains a Friedman Test component

- With this test the key length (number of letters of a key word or phrase) of a polyalphabetic cipher can be calculated



Step-by-step approach methodology: ii) Identify the cipher

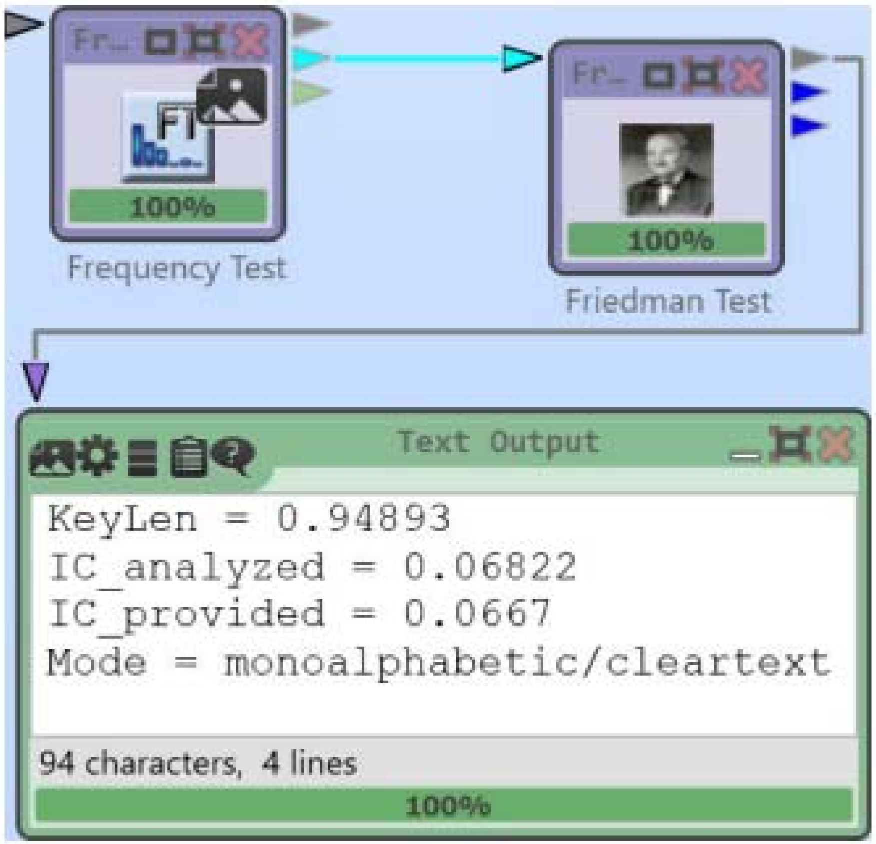
- **Friedman test analysis.**

For that, CT2 contains a Friedman Test component

- Same text (plain text)
- This text is possibly plaintext or a monoalphabetic substitution
- The ciphertext could be transposed since the transposition does not change the letter distribution

Step-by-step approach methodology: ii) Identify the cipher

- **Friedman test analysis.**

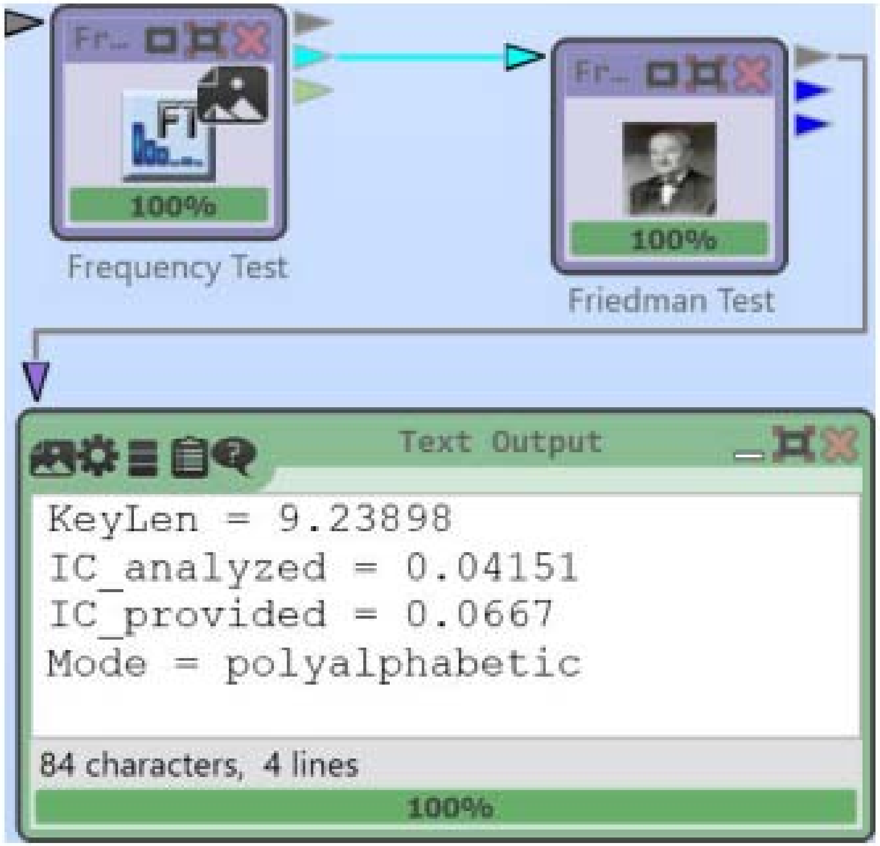


Step-by-step approach methodology: ii) Identify the cipher

- **<u>Friedman test analysis.</u>**
- Same text (ciphertext)
- Vigenere cipher
- It shows that the given text is possibly ciphertext and polyalphabetic
- The ciphertext could be transposed since the transposition does not change the letter distribution

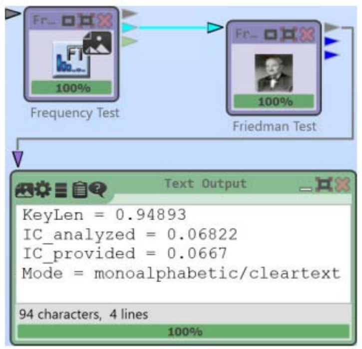
Step-by-step approach methodology: ii) Identify the cipher

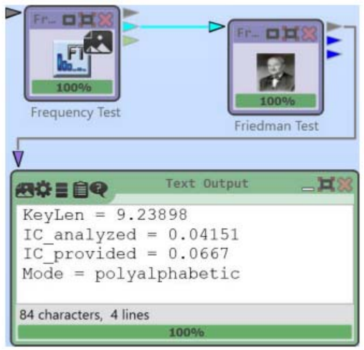
- **Friedman test analysis.**



Step-by-step approach methodology: ii) Identify the cipher

- **<u>Friedman test analysis.</u>**
- Additionally, it shows that the estimated 33 key length is about 9
- The component needs a provided IC (IC provided) which is used as a reference value for the analyzed IC (IC analyzed)





Step-by-step approach methodology: iii) Break the cipher

- **Break the cipher**

We use the help of different cryptanalysis components:

- Monoalphabetic substitution cipher
- Vigenere cipher
- Columnar transposition cipher

Step-by-step approach methodology: iii) Break the cipher

- **<u>Using the properly solver</u>**

Analysis

Start Time:	1/21/2018 8:28:57 PM	End Time:	1/21/2018 8:29:25 PM
Elapsed Time:	00:00:27	Keys/second:	10,119
		Current analyzed keylength:	9

Bestlist

#	Value	Key	Key Length	Text
1	8.91458977542657E	KEYWORD	7	THEDECLARATIONOFINDEPENDENCE
2	2.65449346106438E	DKEKEOKEY	9	ABYPOFEGXVFSVMKZCLOLOENNQXV
3	2.65599400712501E	DOKEDOEO	8	AISVPFKWSVFSCCUVDGHRPANNXYJ
4	2.65713062562153E	DKEKODKEY	9	ABYPEQEGXVFSVCVZCLOLOEDYQXVE
5	2.6594617226191E+	DKEKEODDO	9	ABYPOFLHHVFSVMKGDVOLOENNXYF
6	2.66189596613086E	DKEKEEOEO	9	ABYPOPAGHVFSVMUVCVOLOENXMK
7	2.66951117085202E	DKEKEO	6	ABYPOFLAROLICGUZCVOLOENNXRP
8	2.67115879404954E	DKEDRDKEY	9	ABYWBQEGXVFSCZVZCLOLOLAYQXV
9	2.6840571659966E+	DKKEO	5	ABSVEQEARKMMVMKGWZNHPEHXM
10	2.6865744018561E+	DOKDEO	6	AISWOFLHLVLICNOGCVOSILNNXYJEC

Step-by-step approach methodology: iii) Break the cipher

- **Using solvers**
- The solver automatically tested every key length between 5 and 20 using hill climbing
- Only about ten seconds are needed for the component to automatically break the cipher
- The decrypted text is automatically outputted by the component and can be displayed by an TextOutput component

Step-by-step approach methodology: iii) Break the cipher

- **<u>Using solvers</u>**
- All automatic cryptanalysis components have the same style of user interface
- Besides start and end-time, the elapsed time for the analysis is shown
- Furthermore, some components estimate the time for the remaining automatic analysis

Cryptology for IoT

Modules M4, M7, M9
Session of 27th April, 2022.

M4.3 Briefing to the session
M4.4 Introduction to the ciphers: Substitution , Transposition and mixed ciphers
M4.5 Methodology using Cryptool

Prof.: Guillermo Botella