M183 Applikationssicherheit Implementieren
14

By Jürg Nietlispach

Recap # 13

?

Recap # 13

Data Integrity

- Physical & Logical Domains
- How to achieve it (physical & logical)
    - Constraints for Entity Integrity & Referential Integrity
    - Encryption
        - Block/Stream Ciphers
        - Transposition / Substitution Ciphers
        - Monoalphabetic Substitution
    - Checksums, Error Correction etc.

Data Encryption - Overview

- Block Ciphers vs Stream Ciphers
- Substitution Ciphers vs Transposition Ciphers
- Monoalphabetic Substitution
- **Polyalphabetic Substitution**
- **Symmetric Key Systems vs. Public Key Systems**

Encryption - Polyalphabetic Substitution

Idea

One single Character of an alphabet can be mapped to different Characters of the same alphabet.

Example

Vigenère Cipher: using a Key K for encryption of a plaintext

Plaintext: A T T A C K A T D A W N
Key: L e m o n L e m o n L e
Ciphertext: L X F O P V E F R N H R

Letter A is Mapped to L, O, E and N

Decryption - Polyalphabetic Substitution

Example

Vigenère Cipher: using a Key K for decryption of a ciphertext

Ciphertext: L X F O P V E F R N H R
Key: L e m o n L e m o n L e
Plaintext: A T T A C K A T D A W N

Cryptoanalysis - Polyalphabetic Substitution

Idea

The Fact that there are certain mappings of letters or words, that stay the same (a.k.a Repetitions, see Kasiski & Babbage)

Example 1

Vigenère Cipher: using a Key K for encryption of a plaintext

Plaintext: A T T A C K A T D A W N
Key: L e m o n L e m o n L e
Ciphertext: L X F O P V E F R N H R

Cryptoanalysis - Polyalphabetic Substitution

Example 2

Vigenère Cipher: using a Key K for encryption of a plaintext

```
Key:          ABCDABCDABCDABCDABCDABCDABCD
Plaintext:    CRYPTOISSHORTFORCRYPTOGRAPHY
Ciphertext:   CSASTPKVSIQUTGQUCSASTPIUAQJB
```

Key-Length may be: 16 or every factor of 16: 8, 4, 2, 1

Cryptoanalysis - Polyalphabetic Substitution

When the Key Length L is known -> display Ciphertext in Columns of L

L1	L2	L3	L4	L5	L6
C	V	J	T	N	A
F	E	N	M	C	D
M	K	B	X	F	S
T	K	L3	H	G	S
O	J	W	H	O	F

Perform Frequency analysis within each Column and calculate the shift to the regular frequency of the language. L1 corresponds to the letter of the shifted position!

See also: https://www.dcode.fr/vigenere-cipher

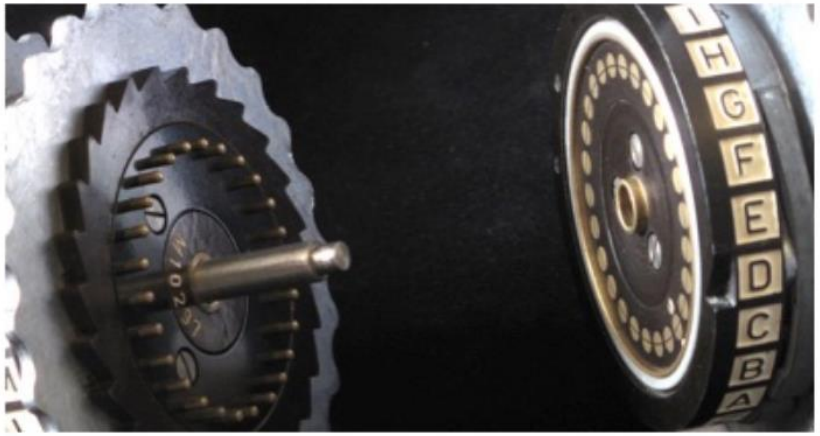
Lab – Polyalphabetic Substitution

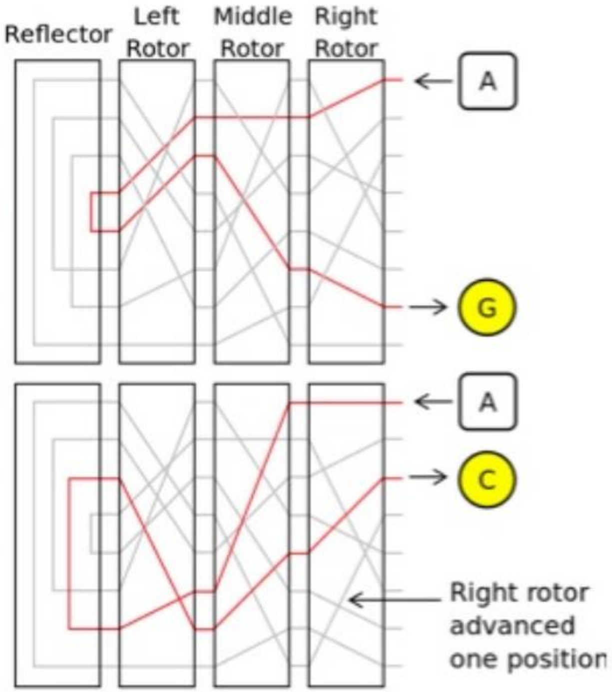
1. Plaintext Analysis Engine (Letter Frequency)
2. Plaintext Encryption Engine (Polyalphabetic with Key given as Character String)
3. Ciphertext Analysis Engine (Letter Frequency, see also https://www.dcode.fr/vigenere-cipher)
4. Ciphertext Decryption Engine (using the given Key)

Encryption «Rolling» Substitution

So far: the polyalphabetic substitution worked using a fixed key that is repeated.
Now: the key defines only the starting point and changes («rolled») during encryption.

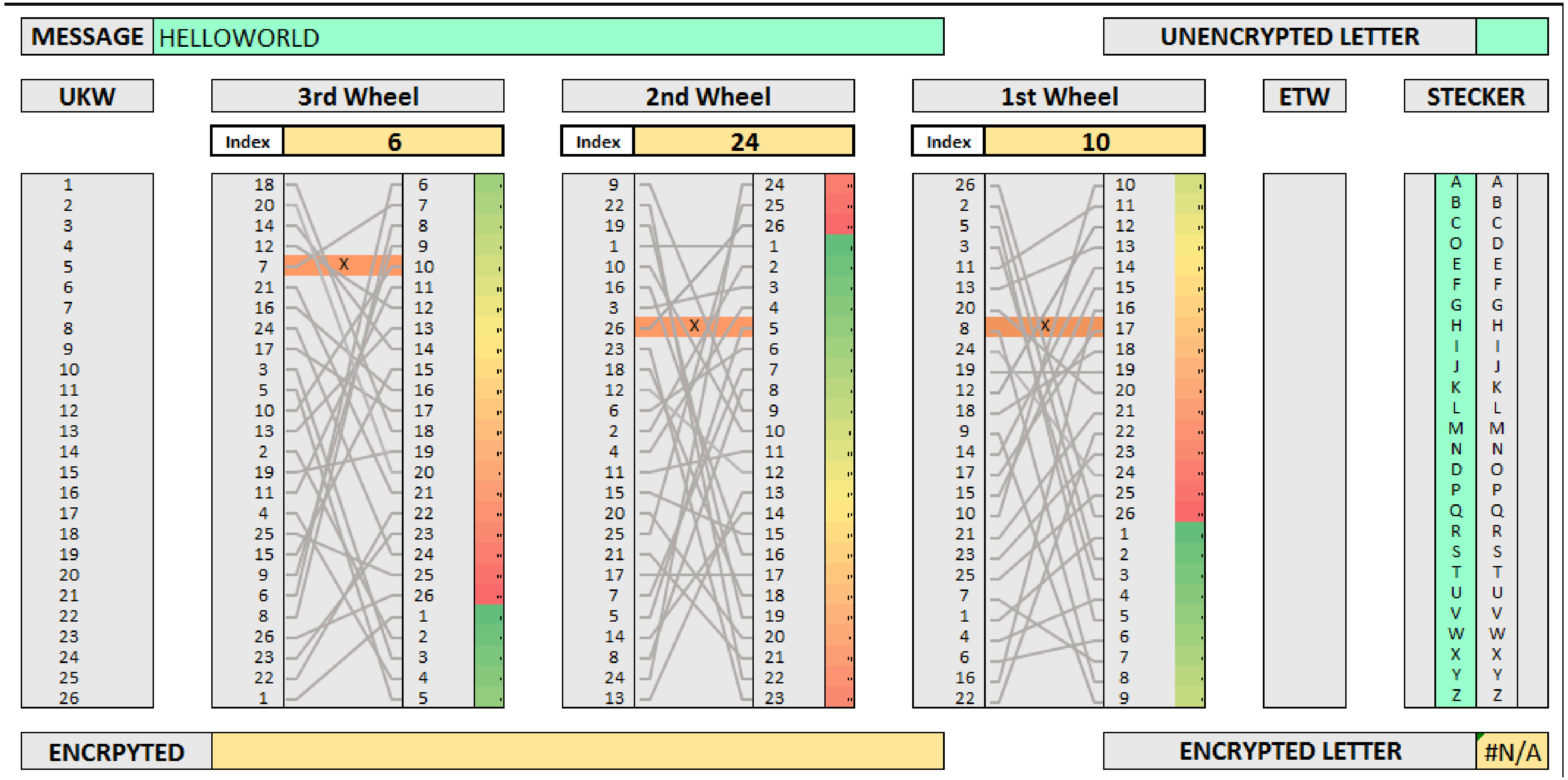
Example: Enigma.



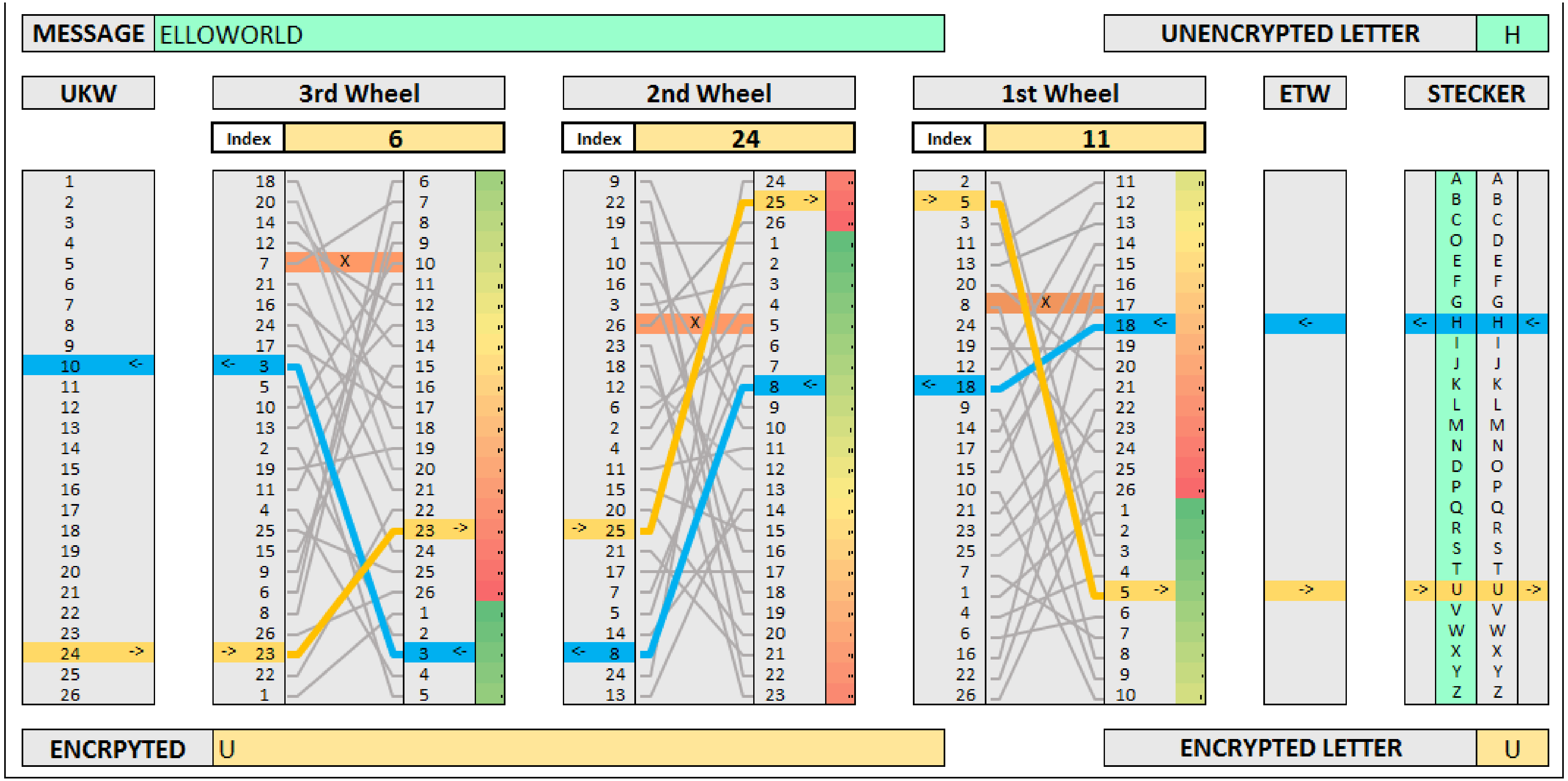


https://www.youtube.com/watch?v=dq27B1-6F5k

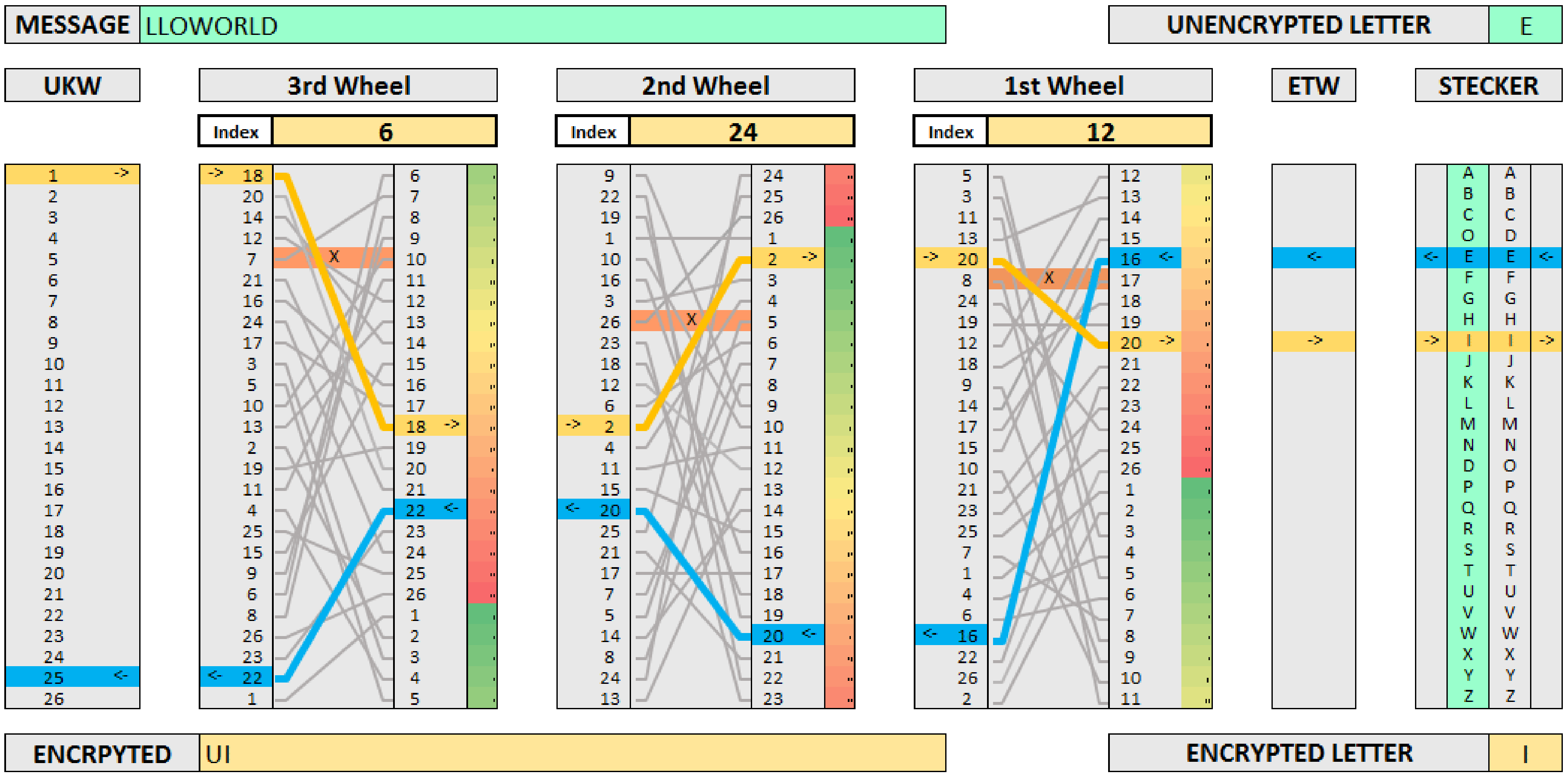
Encryption «Rolling» Substitution 1



Encryption «Rolling» Substitution 2



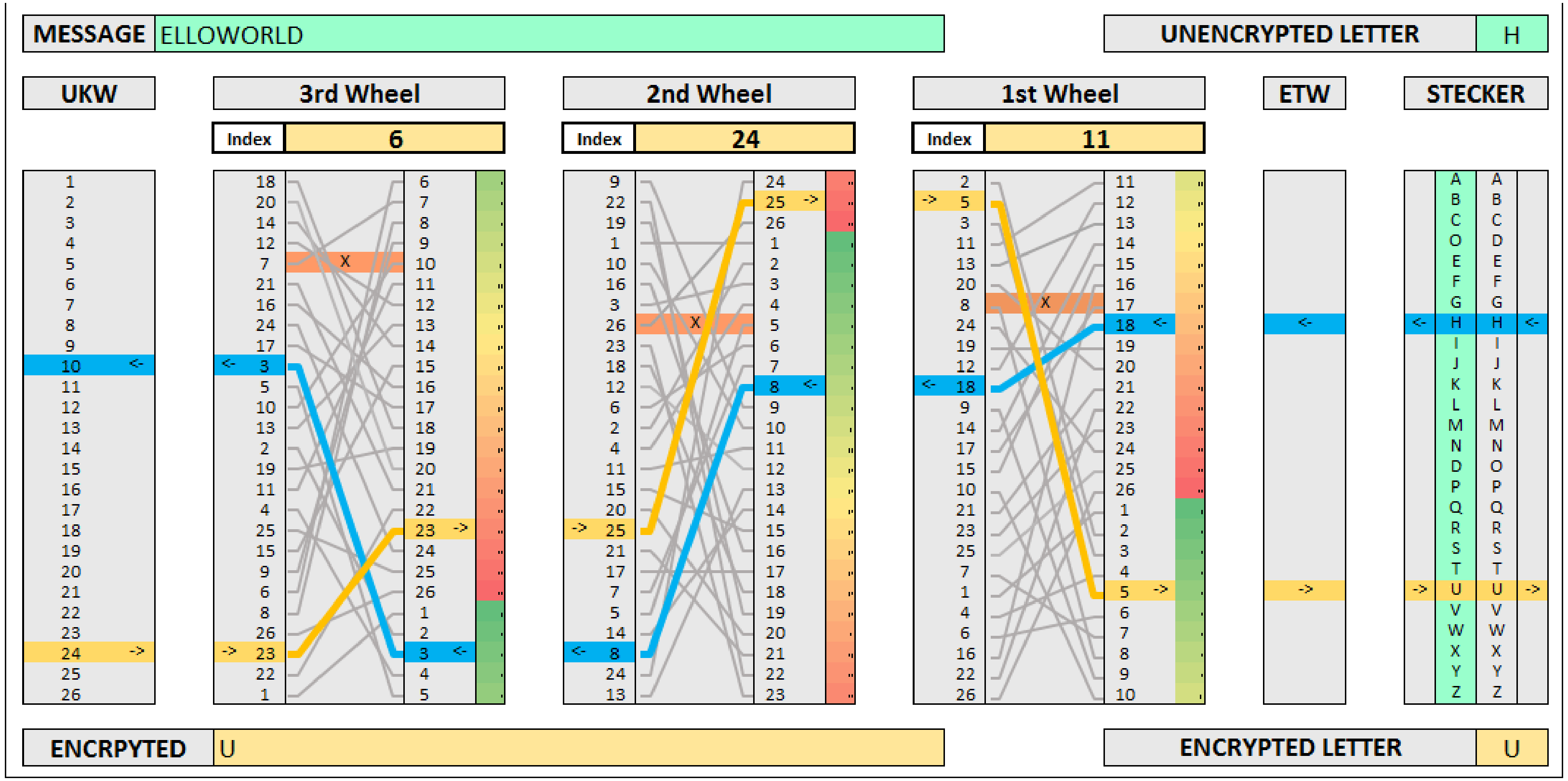
Encryption «Rolling» Substitution 3 etc.



Decryption «Rolling» Substitution

Turn the Enigma in its original state / configuration and enter the ciphertext.

Example: The Power flows in «opposite direction» from U -> H.



Cryptoanalysis «Rolling» Substitution

https://en.wikipedia.org/wiki/Cryptanalysis_of_the_Enigma

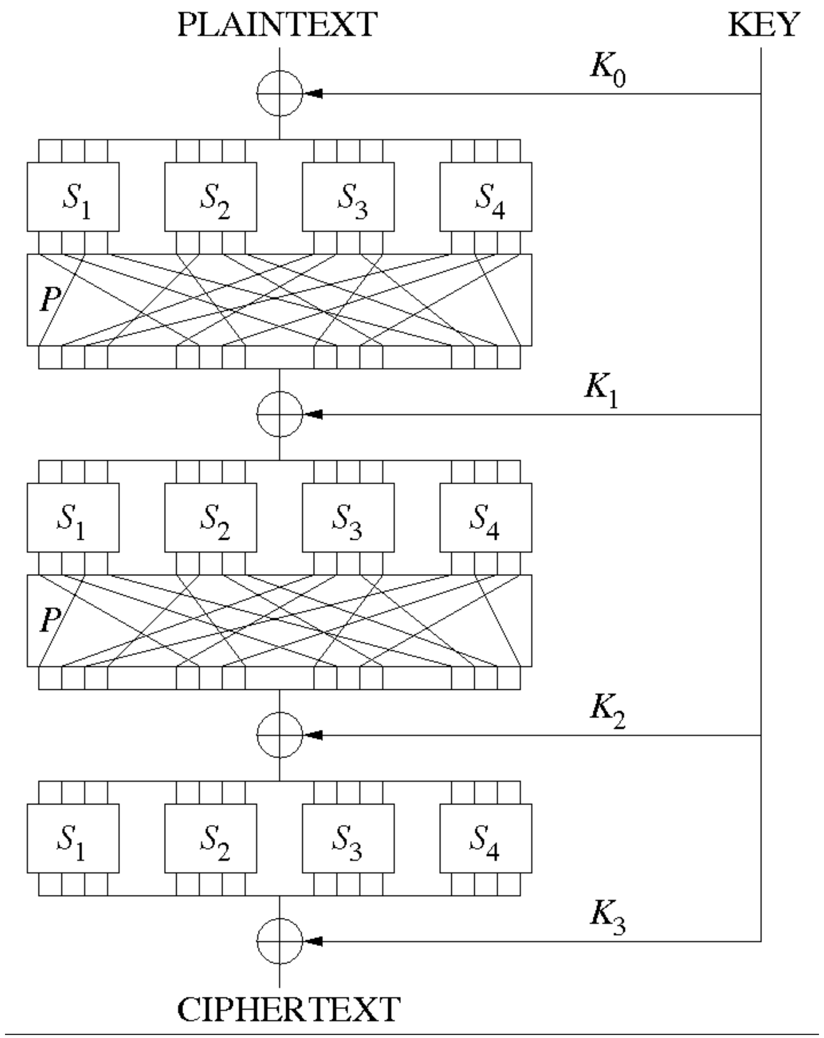
Problem of Cicles / Loops (A -> B -> C -> **A -> B**) due to weak keys (**weak key** is a key, which, used with a specific cipher, makes the cipher behave in some undesirable way)

Encryption using Block Ciphers

So Far: Key defines only the starting position and «changes» over time
But: Weak Key Problem

(One Part of the) Solution: Use Block Ciphers

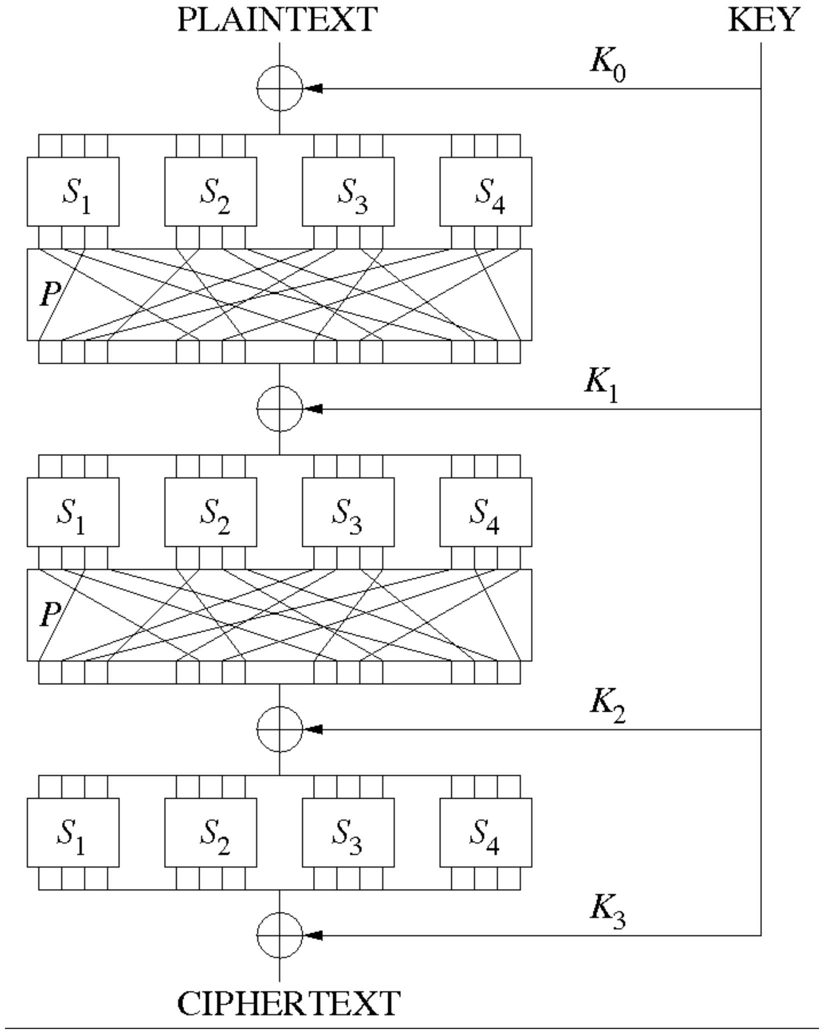
Substitution - permutation networks



Such a network takes a
- block of the plaintext and
- the key as inputs, and
- applies several alternating "rounds" or "layers" of substitution boxes (S-boxes) and permutation boxes (P-boxes)
- to produce the ciphertext block

Examples:
- DES (fixed key length, not recommended),
- AES, Blowfish (both, variable key lenghts)

Properties of Substitution - permutation networks



Must meet **confusion** (correlation key <-> cipertext) and **diffusion** (how redundancy of plaintext transfers to the ciphertext) properties (Shannon)

Diffusion: for a randomly chosen input block, if one flips the *i*-th bit, then the probability that the *j*-th output bit will change is approximately a half, for any *i* and *j*!

The reason for **confusion** is exactly the same as for diffusion ☺

«One Time Pad»

In cryptography, the **one-time pad** (**OTP**) is an encryption technique that cannot be cracked, but requires the use of a one-time pre-shared key the **same size as, or longer than, the message** being sent. In this technique, a plaintext is paired with a random secret key (also referred to as *a one-time pad*)

Encryption

```
      H       E       L       L       O  message
   7 (H)   4 (E)  11 (L)  11 (L)  14 (O) message
+ 23 (X)  12 (M)   2 (C)  10 (K)  11 (L) key
= 30      16      13      21      25     message + key
=  4 (E)  16 (Q)  13 (N)  21 (V)  25 (Z) (message + key) mod 26
      E       Q       N       V       Z  → ciphertext
```

Decryption

```
      E       Q       N       V       Z  ciphertext
   4 (E)  16 (Q)  13 (N)  21 (V)  25 (Z) ciphertext
-  23 (X)  12 (M)   2 (C)  10 (K)  11 (L) key
= -19      4      11      11      14     ciphertext – key
=   7 (H)   4 (E)  11 (L)  11 (L)  14 (O) ciphertext – key (mod 26)
      H       E       L       L       O  → message
```

«One Time Pad»

Attempts of Cryptoanalysis

- Is the key really random?
- Can the key really be exchanged secretly

Lab – One Time Pad

1. Plaintext Analysis Tool (Letter Frequency)
2. Encryption Engine for a Plaintext
3. Key is dynamically generated and displayed
4. Ciphertext is generated and displayed
5. Decryption Engine using the key

Properties of Symmetric Key Systems

1. The encryption and decryption keys are the same.
2. Communicating parties must have the same key before they can achieve secure communication.

How is the Key exchanged securely?

- Second Communication Channel?
- Trusted Courier?
- **Asymmetric / Public Key System**

Asymmetric / Public Key Systems

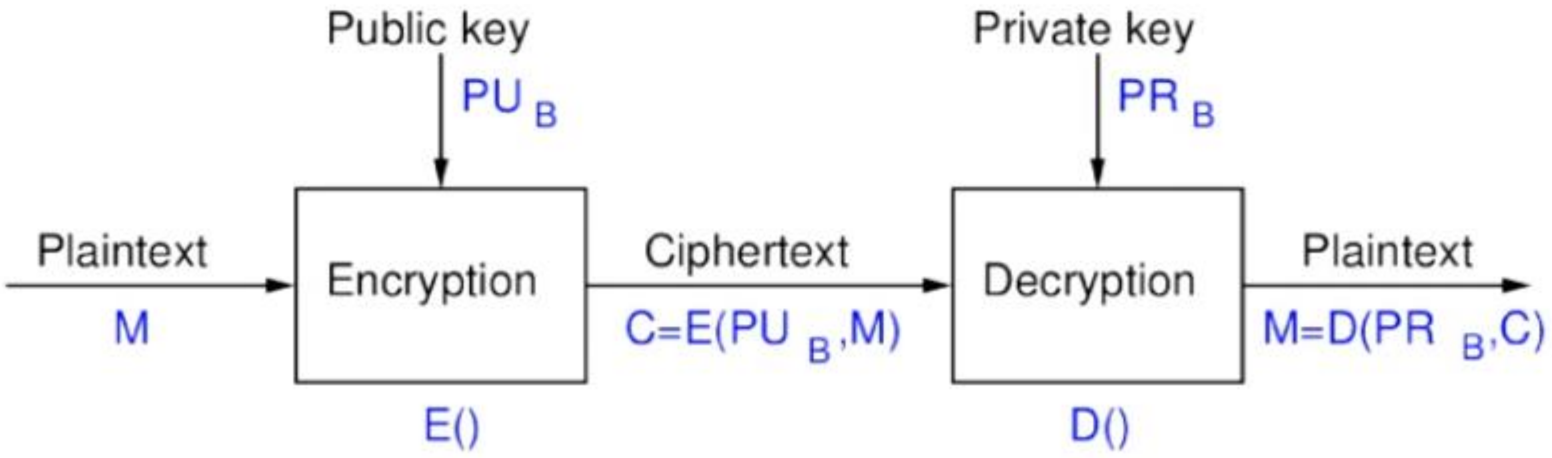
Benefit: no (secure) key exchange necessary!

Idea: Use separate keys for encryption and decryption, one key is publicly accessible!

Examples

- Diffie-Hellmann-Key Exchange, RSA, Digital Certificates

How Public Key Systems Work



1. Plaintext is encrypted with the public Key of the receiver
2. Ciphertext is decrypted with the private Key of the receiver

But: public key for encryption is available for everybody
-> how can we guarantee, that nobody else can decipher the message using the public key?

Why Public Key Systems Work?

Ciphertext (at position i in character code) is calculated as follows:

```
c_i = plaintext_i ^ e mod n
```

E = Public Key
N = Part 2 of the Public Key (N = p * q, two secret & large prime numbers)

For Plaintext (at position i in character code), with known private key d

```
p_i = c_i ^ d mod n
```

For Plaintext (at position i), <u>without</u> private key d, reverse this operation…

-> log_e(c_i) = plaintext_i mod n (-> Discrete Logarithm Problem, very hard to calculate)
-> Or try to factor n (-> Prime Factorization Problem, very hard to calculate)

Public Key System Examples

Self-Study:

https://en.wikipedia.org/wiki/Diffie%E2%80%93Hellman_key_exchange
https://www.youtube.com/watch?v=M7kEpw1tn50

Properties of Public Key Systems

1. Can be used for encrypting data with the public key

2. Can be used for sender verification / authentication
   - In case the a message was encrypted with the private key of the sender – the only way to decrypt the message is by using the public key of the sender!

3. Can be used for both (encrypt it with the private key and the public key)!
   - We can ensure, that the message is encrypted and sender is verified!

Problem: Calculations compared to Symmetric Variants 1000x slower!

More Info: https://www.youtube.com/watch?v=GSIDS_lvRv4

Diffie-Hellmann Key Exchange

The **Diffie–Hellman** key exchange method **allows two parties** that have **no prior knowledge** of each other to jointly establish a shared secret key over an insecure channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher.

Diffie-Hellmann Key Exchange Example

1. Alice and Bob agree to use a modulus p = 23 and base g = 5 (which is a primitive root modulo 23).
2. Alice chooses a secret integer $\mathbf{a}$ = 4, then sends Bob $A = g^{a} \bmod p$
   - $A = 5^{4} \bmod 23 = 4$
3. Bob chooses a secret integer $\mathbf{b}$ = 3, then sends Alice $B = g^{b} \bmod p$
   - $B = 5^{3} \bmod 23 = 10$
4. Alice computes $\mathbf{s} = B^{a} \bmod p$
   - $\mathbf{s} = 10^{4} \bmod 23 = 18$
5. Bob computes $\mathbf{s} = A^{b} \bmod p$
   - $\mathbf{s} = 4^{3} \bmod 23 = 18$
6. Alice and Bob now share a secret (the number 18).

RSA (Rivest-Shamir-Adleman)

Ist ein asymmetrisches kryptographisches Verfahren, das sowohl zum Verschlüsseln als auch zum digitalen Signieren verwendet werden kann.

RSA – Example Setup

1. Wir wählen $p = 11$ und $q = 13$ für die beiden Primzahlen.
2. Der RSA-Modul ist $N = p \cdot q = 143$.
3. Die eulersche φ-Funktion nimmt damit den Wert $\varphi(N) = \varphi(143) = (p-1)(q-1) = 120$ an.
4. Die Zahl e muss zu 120 teilerfremd sein. Wir wählen $e = 23$. Damit bilden $e = 23$ und $N = 143$ den öffentlichen Schlüssel.
5. Berechnung der Inversen zu e bezüglich $\mod \varphi(N)$:
   Es gilt: $e \cdot d + k \cdot \varphi(N) = 1 = ggT(e, \varphi(N))$
   bzw. im konkreten Beispiel: $23 \cdot d + k \cdot 120 = 1 = ggT(23, 120)$. Mit dem erweiterten euklidischen Algorithmus berechnet man nun die Faktoren $d = 47$ und $k = -9$, so dass die Gleichung aus dem Beispiel wie folgt aussieht:
   $23 \cdot 47 + (-9) \cdot 120 = 1$
   d ist der geheime Entschlüsselungsexponent, während k nicht weiter benötigt wird.

RSA – Example Encryption

P_i = 7 (Plaintext Character in Character Code)
N = 143 (Public Key Part 1)
E = 23 (Public Key Part 2)

=> C_i = 7^23 mod 143 => 2

Applications

Often used: Hybrid Variants due to speed! I.E. D-H- Key Exchange of a Symmetric Key (AES)

File Security

- SFTP, NTFS

Traffic Security

- TLS / SSL
- PGP

Database Security

- AES