Cryptanalysis

Lecture 2: The adversary joins the twentieth century

John Manferdelli
jmanfer@microsoft.com
JohnManferdelli@hotmail.com

Dramatis persona

Users

- Alice (party A)
- Bob (party B)
- Trent (trusted authority)
- Peggy and Victor (authentication participants)

Users Agents

- Cryptographic designer
- Personnel Security
- Security Guards
- Security Analysts

Adversaries

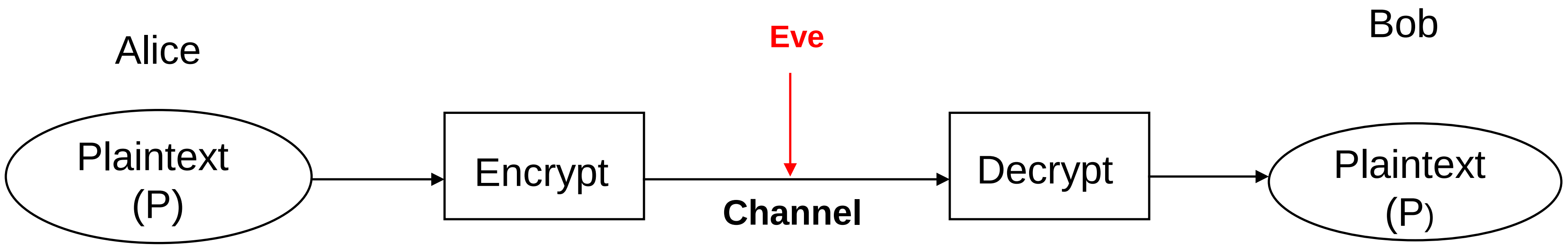
- Eve (passive eavesdropper)
- Mallory (active interceptor)
- Fred (forger)
- Daffy (disruptor)
- Mother Nature
- Users (Yes Brutus, the fault lies in us, not the stars)

Adversaries Agents

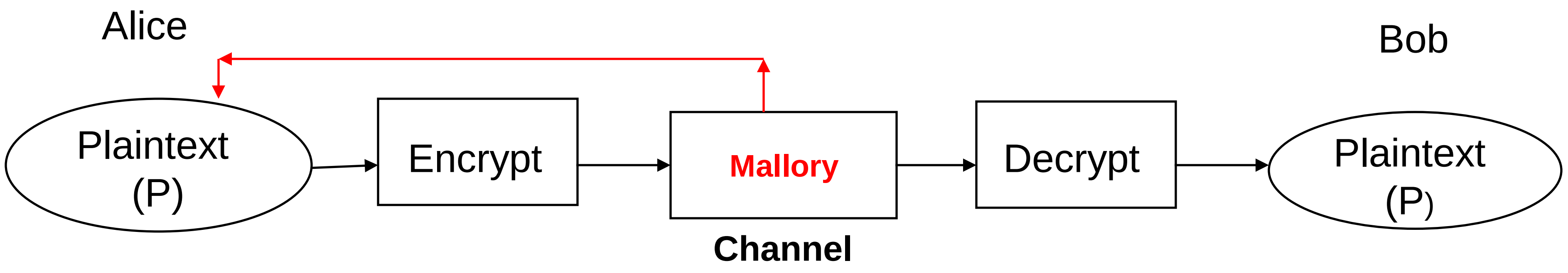
- Dopey (dim attacker)
- Einstein (smart attacker --- you)
- Rockefeller (rich attacker)
- Klaus (inside spy)

Adversaries and their discontents

Wiretap Adversary (Eve)



Man in the Middle Adversary (Mallory)



Claude Shannon

Information Theory Motivation

- How much information is in a binary string?
- Game: I have a value between 0 and 2^n-1 (inclusive), find it by asking the minimum number of yes/no questions.
    - Write the number as $[b_{n-1}b_{n-2}\ldots b_0]_2$.
    - Questions: Is b_{n-1} 1?, Is b_{n-2} 1? , … , Is b_0 1?
- So, what is the amount of information in a number between 0 and 2^n-1?
    - Answer: n bits
    - The same question: Let X be a probability distribution taking on values between 0 and 2^n-1 with equal probability. What is the information content of a observation?
    - There is a mathematical function that measures the information in an observation from a probability distribution. It's denoted H(X).
- $H(X)= \sum_i -p_i \lg(p_i)$

What is the form of H(X)?

- If H is continuous and satisfies:
  - $H(1/n, \ldots, 1/n) < H(1/(n+1), \ldots, 1/(n+1))$
  - $H(p_1,p_2,\ldots,p_j,\ldots,p_n)=H(p_1,p_2,\ldots, qp_j, (1-q)p_j,\ldots,p_n)$
  - $H(p_1,p_2,\ldots,p_j,\ldots,p_n)= 1$ if $p_j= 1/n$ for all j

  then $H(p)= \square_{i=1}^{n} -p_i \lg(p_i)$.

- $H(p_1,p_2,\ldots,p_j,\ldots,p_n)$ is maximized if $p_j= 1/n$ for all j

Information Theory

- The “definition” of H(X) has two desireable properties:
  - Doubling the storage (the bits your familiar with) doubles the information content
  - H(1/2, 1/3, 1/6)= H(1/2, 1/2) + ½ H(2/3,1/3)
- It was originally developed to study how efficiently one can reliably transmit information over “noisy” channel.
- Applied by Shannon to Cryptography (BTSJ, 1949)
- Thus information learned about Y by observing X is

  I(Y,X)= H(Y)-H(Y|X).
- Used to estimate requirements for cryptanalysis of a cipher.

Sample key distributions

- Studying key search
  - Distribution A: 2 bit key each key equally likely
  - Distribution B: 4 bit key each key equally likely
  - Distribution C: n bit key each key equally likely
  - Distribution A': 2 bit key selected from distribution (1/2, 1/6, 1/6, 1/6)
  - Distribution B': 4 bit key selected from distribution (1/2, 1/30, 1/30, …, 1/30)
  - Distribution C': n bit key selected from distribution (1/2, ½ 1/(2^n-1),…, ½ 1/(2^n-1))

H for the key distributions

- Distribution A: H(X)= $\frac{1}{4}$ lg(4) + $\frac{1}{4}$ lg(4) + $\frac{1}{4}$ lg(4) +1/4 lg(4) = 2 bits
- Distribution B: H(X)= 16 x (1/16 lg(16))= 4 bits
- Distribution C: H(X)= 2^n x (1/2^n) lg(2^n) = n bits
- Distribution A': H(X) = $\frac{1}{2}$ lg(2) + 3 x(1/6 lg(6))= 1.79 bits
- Distribution B': H(X) = $\frac{1}{2}$ lg(2) + 15 x(1/30 lg(30))= 2.95 bits
- Distribution C': H(X) = $\frac{1}{2}$ lg(2) + 1/2 2^n-1 x(1/(2^n-1) lg(2^n-1)) $\approx$ n/2+1 bits

Some Theorems

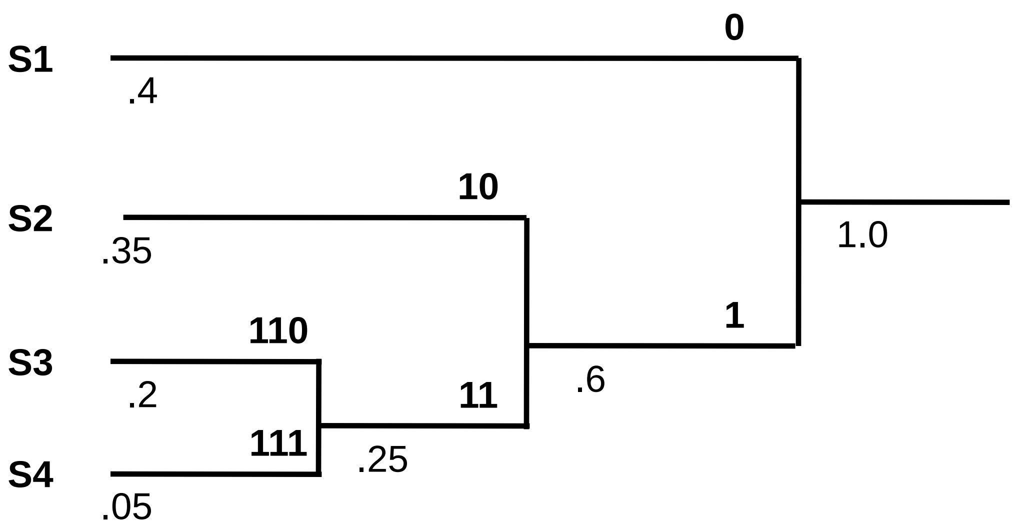
- Bayes: P(X=x|Y=y) P(Y=y)= P(Y=y|X=x) P(X=x)= P(X=x, Y=y)
- X and Y are independent iff P(X=x, Y=y)= P(X=x)P(Y=y)

- H(X,Y)= H(Y)+H(X|Y)
- H(X,Y) $\le$ H(X)+H(Y)
- H(Y|X) $\le$ H(Y) with equality iff X and Y are independent.

- If X is a random variable representing an experiment in selecting one of N items from a set, S, H(X) $\le$ lg(N) with equality iff every selection is equally likely (Selecting a key has highest entropy off each key is equally likely).

Huffman Coding

- Uniquely readable
- Average length, L, satisfies
  - $H(X) \leq L < H(X)+1$



Morse Code

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

H(X)= -(.4lg(.4)) + .35 lg(.35) + .2 lg(.2) + .05 lg(.05))
H(X)= 1.74, [H(X)]= 2. [y] means the ceiling function, the smallest integer greater than or equal to y.

Long term equivocation

- H_E= Lim $_{n \rightarrow \infty} \square_{(x[1],\ldots,x[n])}$ (1/n)Pr(X=(x[1],…,x[n])) lg(Pr(X=(x[1],…,x[n])))
- For random stream of letters
  - H_R= $\square_i$(1/26)lg(26)=4.7004
- For English
  - H_E = 1.2-1.5 (so English is about 75% redundant)
  - There are approximately T(n)= 2^{nH} n symbol messages that can be drawn from the meaningful English sample space.
- How many possible cipher-texts make sense?
  - $H(P^n)+H(K) > H(C^n)$
  - nH_E + lg(|K|) > n lg(|$\square$|)
  - lg(|K|)/(lg(|$\square$|)- H_E)>n
  - R = 1- H_E /lg(|$\square$|)

Unicity and random ciphers

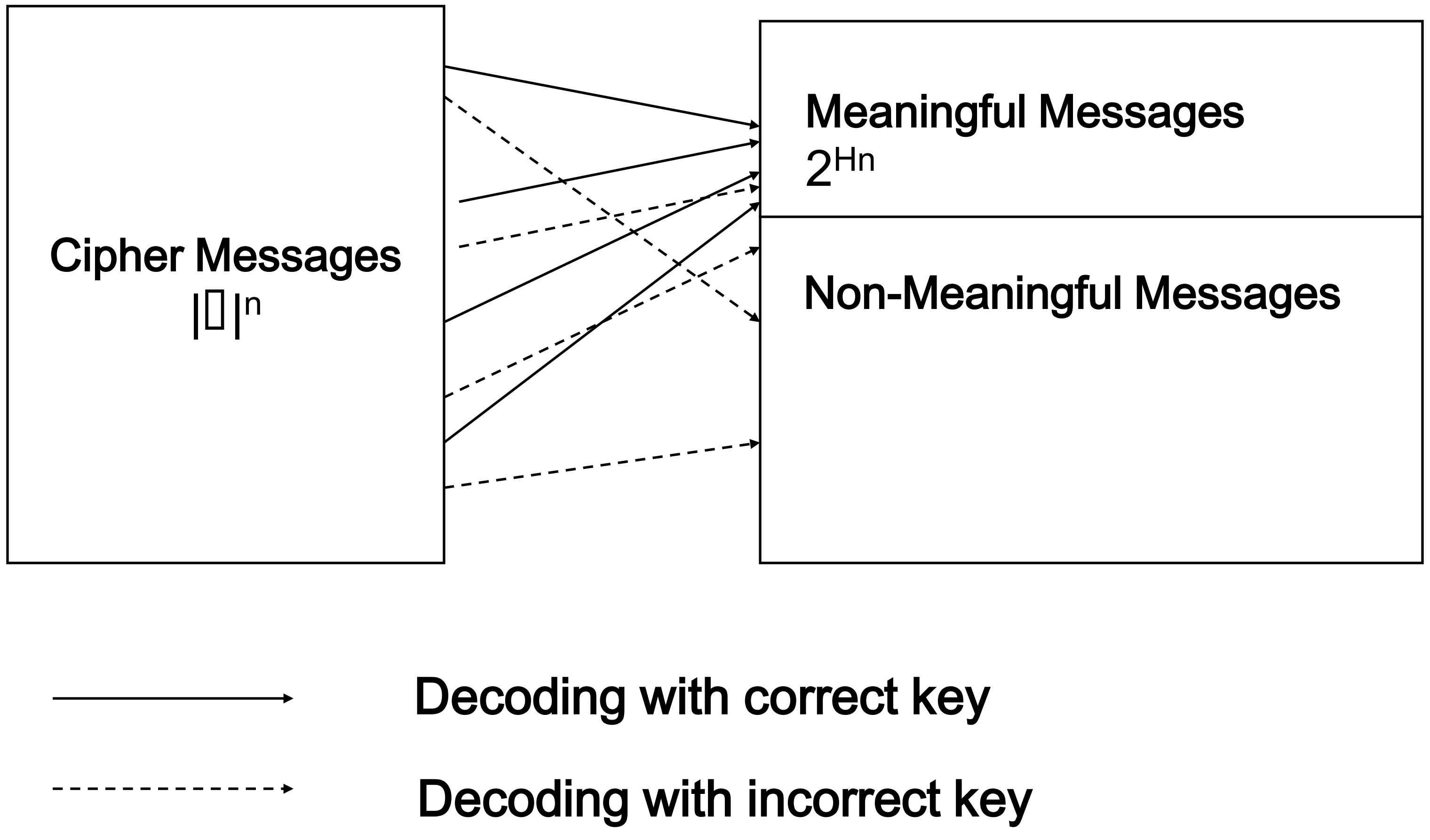
Question: How many messages do I need to trial decode so that the expected number of false keys for which all m messages land in the meaningless subset is less than 1?

Answer: The unicity point.

Nice application of Information Theory.

Theorem: Let H be the entropy of the source (say English) and let □ be the alphabet. Let K be the set of (equiprobable) keys. Then u= lg(|K|)/(lg(|□□)-H).

Unicity for random ciphers



Unicity distance for mono-alphabet

$H_{CaeserKey}= H_{random} = lg(26)= 4.7004$
$H_{English} \approx 1.2.$

- For Caeser, u $\approx$ lg(26)/(4.7-1.2) $\approx$ 4 symbols, for ciphertext only attack. For known plaintext/ciphertext, only 1 corresponding plain/cipher symbol is required for unique decode.
- For arbitrary substitution, u $\approx$ lg(26!)/(4.7-1.2) $\approx$ 25 symbols for ciphertext only attack. For corresponding plain/ciphertext attack, about 8-10 symbols are required.

- Both estimates are remarkably close to actual experience.

Information theoretic estimates to break mono-alphabet

Cipher	**Type of Attack**	**Information Resources**	**Computational Resources**
Caeser	Ciphertext only	U= 4.7/1.2=4 letters	26 computations
Caeser	Known plaintext	1 corresponding plain/cipher pair	1
Substitution	Ciphertext only	~30 letters	O(1)
Substitution	Known plaintext	~10 letters	O(1)

One Time Pad (OTP)

- The one time pad or Vernam cipher takes a plaintext consisting of symbols $\mathbf{p} = (p_0, p_1, \ldots, p_n)$ and a keystream $\mathbf{k} = (k_0, k_1, \ldots, k_n)$ where the symbols come from the alphabet Z_m and produces the ciphertext $\mathbf{c} = (c_0, c_1, \ldots, c_n)$ where $c_i = (p_i + k_i) \pmod{m}$.
- Perfect security of the one time pad: If $P(k_i=j)=1/m$ and is iid, $0<=j<m$, then $H(\mathbf{c}|\mathbf{p})=H(\mathbf{p})$ so the scheme is secure.
- m=2 in the binary case and m=26 in the case of the roman alphabet.
- Stream ciphers replace the ‘perfectly random’ sequence **k** with a pseudo-random sequence **k’** (based on a much smaller input key $\mathbf{k_s}$ and a stream generator R).

One-time pad alphabetic encryption

Plaintext +Key (mod 26)= Ciphertext

B	U	L	L	W	I	N	K	L	E	I	S	A	D	O	P	E	*Plaintext*
1	20	11	11	22	08	13	10	11	04	08	18	00	03	14	15	04	
N	O	W	I	S	T	H	E	T	I	M	E	F	O	R	A	L	*Key*
13	14	22	08	18	19	07	04	19	08	12	04	05	14	17	00	11	
14	8	07	19	14	01	20	14	04	12	20	22	05	17	05	15	15	*Ciphertext*
O	S	H	T	0	B	U	O	E	M	U	W	F	R	F	P	P	

Legend

A	B	C	D	E	F	G	H	I	J	K	L	M
00	01	02	03	04	05	06	07	08	09	10	11	12
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

JLM 20080915

One-time pad alphabetic decryption

Ciphertext+26-Key (mod 26)= Plaintext

14	8	07	19	14	01	20	14	04	12	20	22	05	17	05	15	15	*Ciphertext*
O	S	H	T	0	B	U	O	E	M	U	W	F	R	F	P	P	
N	O	W	I	S	T	H	E	T	I	M	E	F	O	R	A	L	*Key*
13	14	22	08	18	19	07	04	19	08	12	04	05	14	17	00	11	
B	U	L	L	W	I	N	K	L	E	I	S	A	D	O	P	E	*Plaintext*
1	20	11	11	22	08	13	10	11	04	08	18	00	03	14	15	04	

Legend

A	B	C	D	E	F	G	H	I	J	K	L	M
00	01	02	03	04	05	06	07	08	09	10	11	12
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

Binary one-time pad

Plaintext $\oplus$ Key = Ciphertext

Ciphertext $\oplus$ Key = Plaintext

10101110011100000101110110110000	*Plaintext*
00101010011010110001010110010111	*Key*
10100100000110110100100000100111	*Ciphertext*
00101010011010110001010110010111	*Key*
10101110011100000101110110110000	*Plaintext*

The one time pad has perfect security

- E is perfect if $H(X|Y)=H(X)$ where X is a plaintext distribution and Y is the ciphertext distribution with respect to a cipher E.
- To show a one time pad on a (binary) plaintext message of length L with ciphertext output a message of length L with keys taken from a set K consisting of 2^L keys each occurring with probability 2^{-L}, we need to show $H(X|Y)=H(X)$.

Proof:

$H(X|Y) = -\sum_{y \text{ in } Y} P(Y=y)\, H(X|Y=y)) = -\sum_{y \text{ in } Y} P(Y=y) \sum_{x \text{ in } X} P(X=x|Y=y) \lg(P(X=x|Y=y)).$

$P(X=x|Y=y)\, P(Y=y)= P(X=x, Y=y)$ and $P(X=x,Y=y) = \Pr(X=x, K=x+y)= P(X=x)P(K=k).$

So $H(X|Y) = -\sum_{y \text{ in } Y,\, x \text{ in } X} P(X=x,Y=y)\, [\lg(P(X=x,Y=y) - P(Y=y)]$

$= -\sum_{y \text{ in } Y,\, x \text{ in } X} P(X=x, Y=y) \lg(P(X=x, Y=y)) + \sum_{y \text{ in } Y,\, x \text{ in } X} P(X=x,Y=y) \lg(P(Y=y))$

$= -\sum_{x \text{ in } X,\, y \text{ in } Y} P(X=x)P(K=x+y)\lg(P(X=x) - \sum_{x \text{ in } X,\, y \text{ in } Y} P(X=x)\, P(Y=x+k)\lg(P(Y=x+k)$

$+\sum_{y \text{ in } Y,\, x \text{ in } X} P(X=x)\, P(Y=Y)\lg(P(Y=y))$

$= H(X)$

Mixing cryptographic elements to produce strong cipher

- Diffusion – transposition
  - Using group theory, the action of a transposition □ on a_1 a_2 …a_k could be written as $a_{□(1)}$ $a_{□(2)}$ …$a_{□(k)}$.
- Confusion – substitution
  - The action of a substitution □ on a_1 a_2 …a_k can be written as □(a_1) □(a_2) … □(a_k) .
- Transpositions and substitutions may depend on keys. Keyed permutations may be written as $□_k$(x). A block cipher on b bits is nothing more than a keyed permutation on 2^b symbols.
- Iterative Ciphers – key dependant staged iteration of combination of basic elements is very effective way to construct cipher. (DES, AES)

Linear Feedback Shift Registers

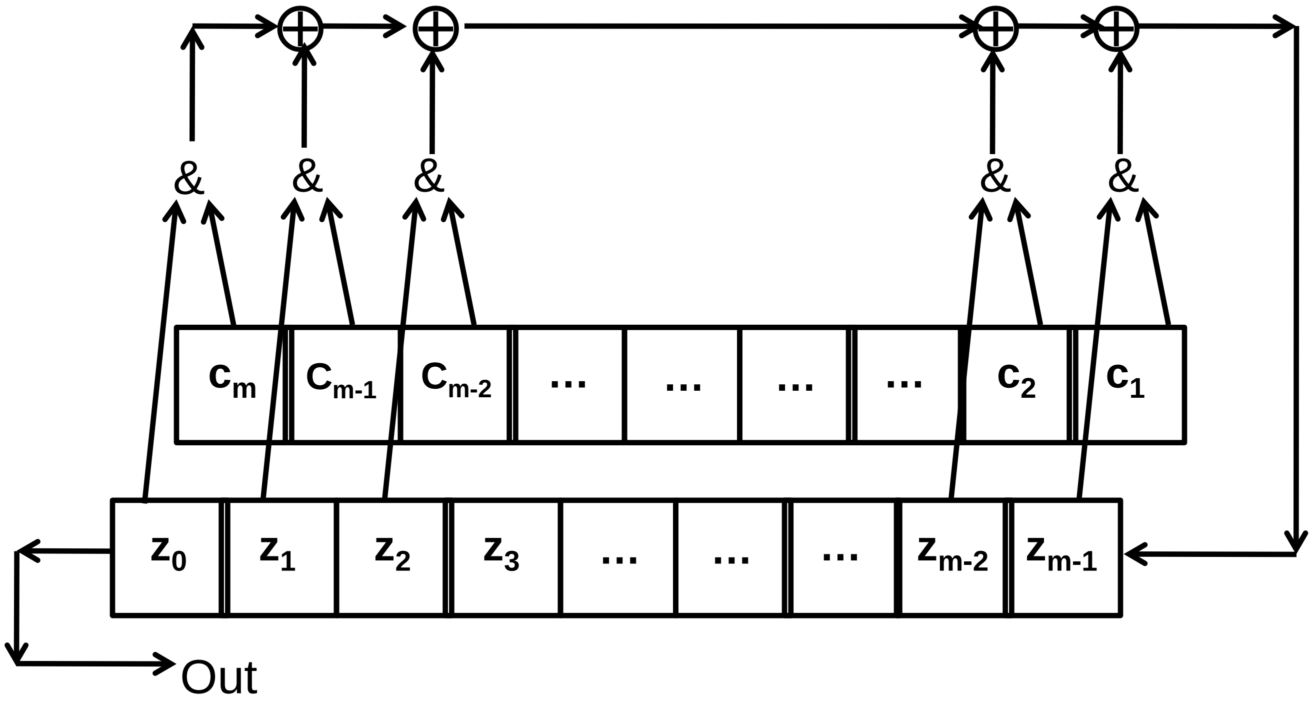
Binary one-time pad

Plaintext $\oplus$ Key = Ciphertext

Ciphertext $\oplus$ Key = Plaintext

10101110011100000101110110110000	*Plaintext*
00101010011010110001010110010111	*Key*
10100100000110110100100000100111	*Ciphertext*
00101010011010110001010110010111	*Key*
10101110011100000101110110110000	*Plaintext*

Linear Feedback Shift Registers (LFSR)



- State at time t: $S(t)= <z_0, z_1, \ldots, z_{m-1}>=<s_t, s_{t+1}, \ldots, s_{t+m-1}>$.
- Recurrence is $s_{j+1}= c_1 s_j + \ldots + c_m\ s_{j-m-1}$,
- At time t, LFSR outputs $z_0 = s_t$, shifts, and replaces z_{m-1} with $c_1 z_{m-1} + \ldots + c_m\ z_0$.

LFSR as linear recurrence

- G(x) is power series representing the LFSR, coefficients are outputs.
- $G(x)= a_0 + a_1 x + a_2 x^2 + \dots + a_k x^k + \dots$
- Let $c(x)= c_1 x + \dots + c_m x^m$.
- Because of the recurrence, $a_{t+m} = \sum_{0<i<m+1} c_i a_{t+m-i}$,
  - $G(x)= a_0 + a_1 x + a_2 x^2 + \dots + a_{m-1} x^{m-1} + x^m (c_1 a_{m-1} + \dots + c_m a_0) + x^{m+1} (c_1 a_m + \dots + c_m a_1) + x^{m+2} (c_1 a_{m+1} + \dots + c_m a_2) + \dots$
  - After some playing around, this can be reduced to an equation of the form $G(x)= K/(1-c(x))$, where K is a constant that depends on initial state only. Let $f(x)= 1-c(x)$ be the called the connection polynomial. [$1-c(x)=1+c(x) \pmod 2$, of course].
  - If the period of the sequence is p, $G(x)= (a_0 + a_1 x + \dots + a_{p-1} x^{p-1}) + x^p(a_0 + a_1 x + \dots + a_{p-1} x^{p-1}) + \dots = (a_0 + a_1 x + \dots + a_{p-1} x^{p-1})(1+x^p + x^{2p} + \dots)$
- We get $(a_0 + a_1 x + \dots + a_{p-1} x^{p-1})/(1-x^p)= K/(f(x))$ so $f(x) \mid 1-x^p$ and f(x) is the equation for a root of 1. If f(x) is a primitive root of 1 p will be as large as possible, namely, $p=2^m-1$.

LFSR performance metrics

- The output sequence of and LFSR is periodic for all initial states. The maximal period is 2^m-1.
- A non-singular LFSR with primitive feedback polynomial has maximal period of all non-zero initial states
- A length m LFSR is determined by 2m consecutive outputs
- Linear complexity of sequence $z_0, z_1, \ldots, z_n$ is the length of the smallest LFSR that generates it
- Berlekamp-Massey: $O(n^2)$ algorithm for determining linear complexity

Linear Complexity, simple $O(n^3)$ algorithm

- There is a non-singular LFSR of length m which generates $s_0, s_1, \ldots, s_k \ldots$ iff there are $c_1, \ldots, c_m$ such that:

$$s_{m+1} = c_1 s_m + c_2 s_{m-1} + \ldots + c_m s_1$$
$$s_{m+2} = c_1 s_{m+1} + c_2 s_m + \ldots + c_m s_2$$
$$\ldots$$
$$s_{2m} = c_1 s_{2m-1} + c_2 s_{2m-2} + \ldots + c_m s_{m+1}$$

- To solve for the c_i's just use Gaussian Elimination (see math summary) which is $O(n^3)$.
- But there is a more efficient way!

Berlekamp-Massey

- Given output of LFSR, $s_0, s_1, \ldots, s_{N-1}$, calculate length, L, of smallest LFSR that produces $<s_i>$. Algorithm below is $O(n^2)$. In the algorithm below, the connection polynomial is: $c(x) = c_0 + c_1 x + \ldots + c_L x^L$ and $c_0=1$ always.

```
c(x)=1; L= 0; m= -1; b(x)=1;
for(n=0; n<N; n++)
    d= $s_n + \sum_{i=1}^{L-1} c_i\ s_{n-i}$    //  d is the “discrepency”
    if(d!=0) {
        t(x)= c(x);
        c(x)= c(x) + b(x) $x^{n-m}$;
        if(L<=n/2)) {
            L=n+1-L;
            m= n;
            b(x)= t(x);
            }
    }
}
```

Berlekamp-Massey example

- $s_0, s_1, \ldots, s_{N-1}$ = 001101110, N=9

n	**s_n**	**t(x)**	**c(x)**	**L**	**m**	**b(x)**	**d**
-	-	-	1	0	-1	1	-
0	0	-	1	0	-1	1	0
1	0	-	1	0	-1	1	0
2	1	1	$1+x^3$	3	2	1	1
3	1	$1+x^3$	$1+x+x^3$	3	2	1	1
4	0	$1+x+x^3$	$1+x+x^2+x^3$	3	2	1	1
5	1	$1+x+x^2+x^3$	$1+x+x^2$	3	2	1	1
6	1	$1+x+x^2+x^3$	$1+x+x^2$	3	2	1	0
7	1	$1+x+x^2$	$1+x+x^2+x^5$	5	7	$1+x+x^2$	1
8	0	$1+x+x^2+x^5$	$1+x^3+x^5$	5	7	$1+x+x^2$	1

Linear complexity and linear profile

- "Best" (i.e.-highest) linear complexity for $S_N = s_0, s_1, \ldots, s_{N-1}$ is L=N/2.
- Complexity profile for S is the sequence of linear complexities $L_1, L_2, \ldots, L_{N-1}$ for $S_1, S_1, \ldots, S_N$.
- For a "strong" shift register, we want not just large L but large L_k for subsequences (thus hug the line L= N/2).
- $E(L(< s_0, s_1, \ldots, s_{N-1}>)) = N/2 + (4+(\sum_{i=0}^{N-1} s_i) \pmod 2)/18 - 2^{-N}(N/3+2/9)$

Example: Breaking a LFSR

- $z_{n+1} = c_1 z_n + \ldots + c_m z_{n-m-1}$. m=8.
- Plain: 1 0 0 1 1 1 0 1 0 1 1 1 0 0 1 0 1 1 1
- Cipher: 1 1 1 1 0 0 1 0 1 0 1 0 1 1 0 0 0 1 0
- LFSR Output: 0 1 1 0 0 1 1 1 1 1 0 1 1 1 1 0 1 0 1

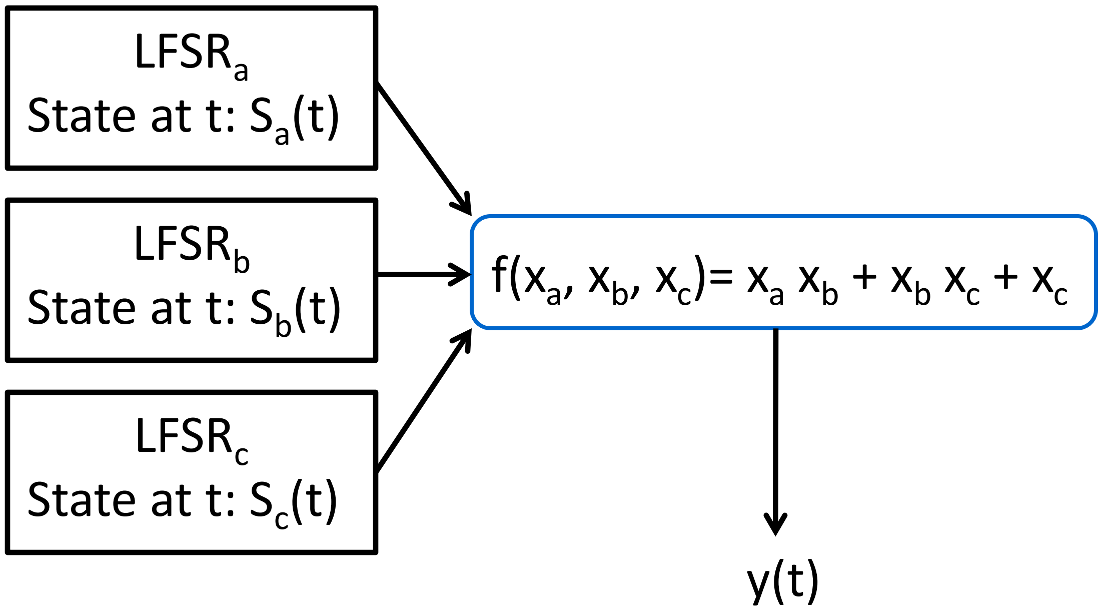
	c_8	c_7	c_6	c_5	c_4	c_3	c_2	c_1	
i	Z_0	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	Z_7	S_{i+8}
0	0	1	1	0	0	1	1	1	1
1	1	1	0	0	1	1	1	1	1
2	1	0	0	1	1	1	1	1	0
3	0	0	1	1	1	1	1	0	1
4	0	1	1	1	1	1	0	1	1
5	1	1	1	1	1	0	1	1	1
6	1	1	1	1	0	1	1	1	1
7	1	1	1	0	1	1	1	1	0

- GE gives solution $(c_1, c_2, \ldots, c_8)$: 10110011

Geffe Generator

- Three LFSRs of maximal periods (2^a-1), (2^b-1), (2^c-1) respectively.
- Output filtered by $f(x_a, x_b, x_c)= x_a x_b + x_b x_c + x_c$
- Period: $(2^a-1)(2^b-1)(2^c-1)$
- Linear complexity: $ab+bc+c$
- Simple non-linear filter.

Geffe Generator



x_a	x_b	x_c	$f(x_a, x_b, x_c)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

- Note that x_c and $f(x_a, x_b, x_c)$ agree 75% of the time.

Correlation attack: breaking Geffe

- Guess $S_c(0)$ and check the agreement of $S_c(t)_{out}$ and $y(t)$.
  - If guess is right, they will agree much more often than half the time
  - If guess is wrong, they will agree about half the time
  - In this way, we obtain $S_c(0)$.
- Now guess $S_b(0)$.
  - Compare $y(t)$ and $x_a\ S_b(t)_{out}+S_b(t)_{out}\ S_c(t)_{out}+S_c(t)_{out}$.
  - If guess is right they will agree much more often than half the time.
  - If not they will agree about half the time.
  - In this way, we obtain $S_b(0)$.
- Now guess $S_a(0)$.
  - $y(t)$ and $S_a(t)\ S_b(t)_{out} + S_b(t)_{out}\ S_c(t)_{out} + S_c(t)_{out}$ will be the same as $y(t)$ for the correct guess.

- Complexity of attack (on average) is about $2^{a-1}+ 2^{b-1}+ 2^{c-1}$ rather than about $2^{a+b+c-1}$ which is what we'd hoped for.

Shrinking Generator

- Two LFSRs of maximal periods (2^s-1), (2^a-1) respectively. $(a,s)=1$.
- Output is output of A clocked by S.
- Period: $(2^{s-1}-1)(2^a-1)$.
- Linear Complexity: $a2^{s-2}<c<a2^{s-1}$
- SEAL cipher from Coppersmith.

Observations

- Matching Alphabets as monotonic process.
- Statistics and Hill climbing.
- Polynomials over finite fields are easier to solve because there are no round-off errors.
- Polynomials over finite fields are harder to solve because there is no intermediate value theorem.
- We’ll stop here with classical ciphers although we could go much further by examining some other systems like Lorenz, Purple, M-209 and SIGABA.

Applying Shannon’s Design Principles

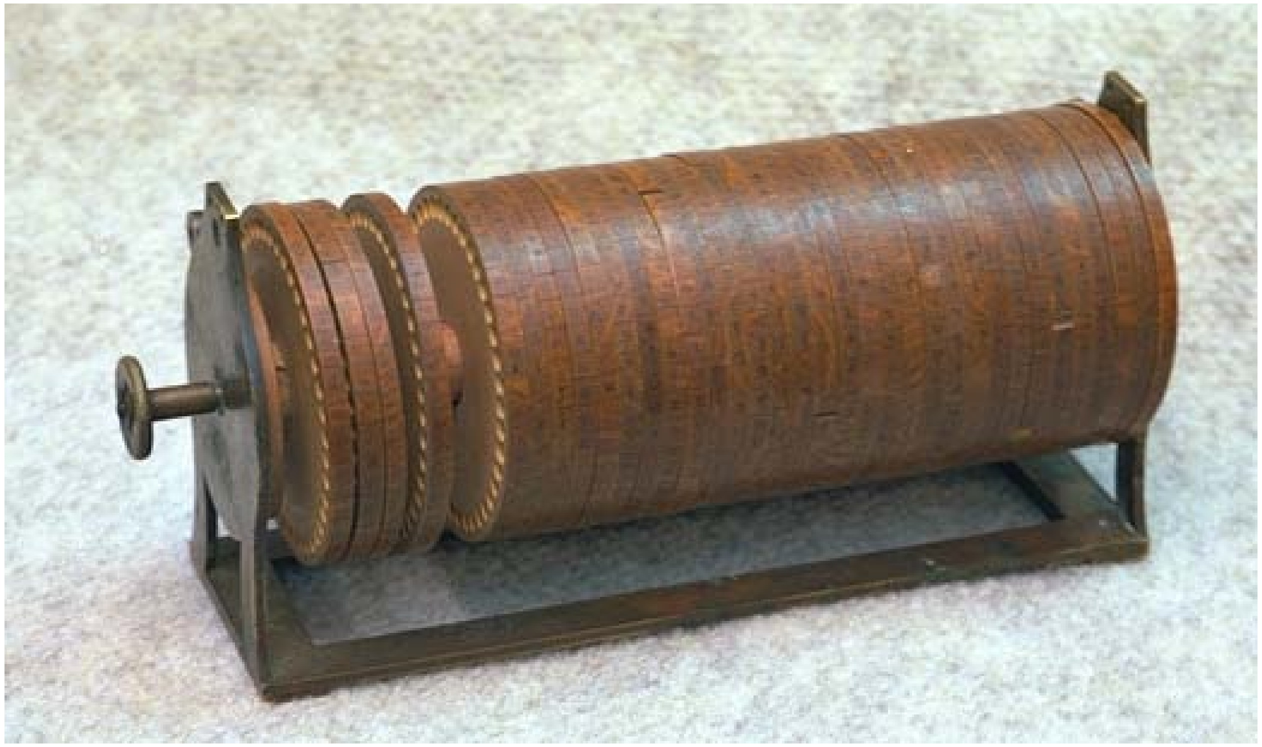
- Two basic building blocks for any cryptographic system
- Diffusion
  - statistical structure of the plain text is dissipated into long-range statistics of the ciphertext
  - each plaintext digit affects many ciphertext digits
  - each ciphertext digit is affected by many plaintext digits
  - achieved using permutation (P)
- Confusion
  - make the relationship between the statistics of the ciphertext and the value of the encryption key as complex as possible
  - this is achieved by the complex subkey generation algorithm and non-linear substitutions

Rise of the Machines

The “Machine” Ciphers

- Simple Manual Wheels
  - Wheatstone
  - Jefferson
- Rotor
  - Enigma
  - Heburn
  - SIGABA
  - TYPEX
- Stepping switches
  - Purple
- Mechanical Lug and cage
  - M209

Jefferson Cipher

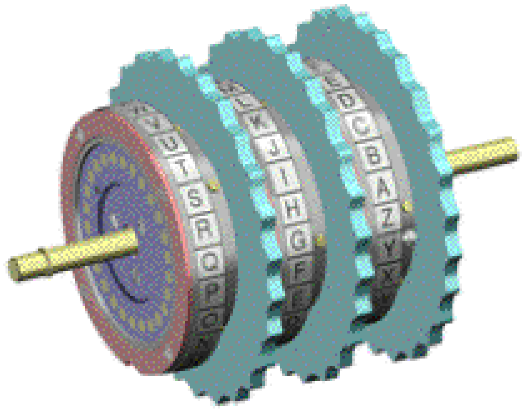


I’d vote for Jefferson. The French have another name for this cipher. They liked Jefferson too but not that much.

Enigma

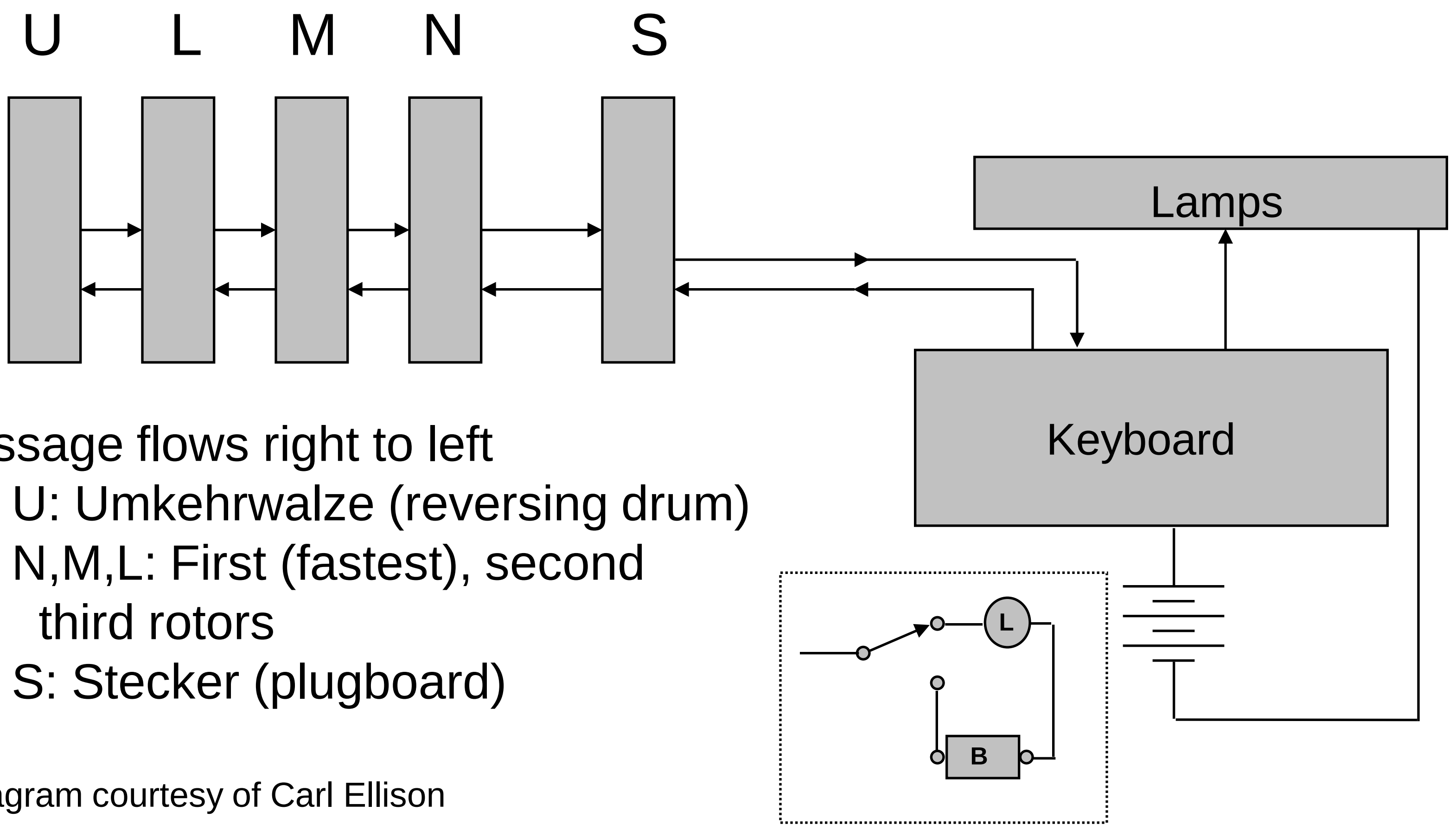
Enigma Cryptographic Elements (Army Version)

- Three moveable rotors
  - Select rotors and order
  - Set initial positions
- Moveable ring on rotor
  - Determine rotor 'turnover'
- Plugboard (Stecker)
  - Interchanges pairs of letters
- Reversing drum (Umkehrwalze)
  - Static reflector
  - See next page



Three Rotors on axis

Diagrammatic Enigma Structure



Message flows right to left

U: Umkehrwalze (reversing drum)

N,M,L: First (fastest), second third rotors

S: Stecker (plugboard)

Diagram courtesy of Carl Ellison

Enigma Data

Rotors

Input	ABCDEFGHIJKLMNOPQRSTUVWXYZ
Rotor I	EKMFLGDQVZNTOWYHXUSPAIBRCJ
Rotor II	AJDKSIRUXBLHWTMCQGZNPYFVOE
Rotor III	BDFHJLCPRTXVZNYEIWGAKMUSQO
Rotor IV	ESOVPZJAYQUIRHXLNFTGKDCMWB
Rotor V	VZBRGITYUPSDNHLXAWMJQOFECK
Rotor VI	JPGVOUMFYQBENHZRDKASXLICTW
Rotor VII	NZJHGRCXMYSWBOUFAIVLPEKQDT

Ring Turnover

Rotor I	R
Rotor II	F
Rotor III	W
Rotor IV	K
Rotor V	A
Rotors VI	A/N

Reflector B	(AY) (BR) (CU) (DH) (EQ) (FS) (GL) (IP) (JX) (KN) (MO) (TZ) (VW)
Reflector C	(AF) (BV) (CP) (DJ) (EI) (GO) (HY) (KR) (LZ) (MX) (NW) (TQ) (SU)

Group Theory for Rotors

- Writing cryptographic processes as group operation can be very useful. For example, if R denotes the mapping of a “rotor” and C=(1,2,…,26), the mapping of the rotor “turned” one position is CRC^{-1}.
- A prescription for solving ciphers is to represent the cipher in terms of the basic operations and then solve the component transformations. That is how we will break Enigma.
- For most ciphers, the components are substitution and transposition; some of which are “keyed”.
- For Enigma, you should know the following:
  - Theorem: If $\square=(a_{11}\ a_{12} \ldots a_{1i})\ (a_{11} \ldots a_{1j}) \ldots (a_{11} \ldots a_{1k})$ then $\square\square\,\square^{-1}=(\square a_{11}\ \square a_{12} \ldots \square a_{1i})\ (\square a_{11} \ldots \square a_{1j}) \ldots (\square a_{11} \ldots \square a_{1k})$.
  - When permutations are written as products of cycles, it is very easy to calculate their order. It is the LCM of the length of the cycles.

Military Enigma

Encryption Equation

- $c = (p)\, P^{i}NP^{-i}\, P^{j}MP^{-j}\, P^{k}LP^{-k}\, U\, P^{k}L^{-1}P^{-k}\, P^{j}M^{-1}P^{-j}\, P^{i}N^{-1}P^{-i}$
  - K: Keyboard
  - P=(ABCDEFGHIJKLMNOPQRSTUVWXYZ)
  - N: First Rotor
  - M: Second Rotor
  - L: Third Rotor
  - U: Reflector. Note: $U=U^{-1}$.
  - i,j,k: Number of rotations of first, second and third rotors respectively.
- Later military models added plugboard (S) and additional rotor (not included). The equation with Plugboard is:
- $c=(p)S\, P^{i}NP^{-i}\, P^{j}MP^{-j}\, P^{k}LP^{-k}\, U\, P^{k}L^{-1}P^{-k}\, P^{j}M^{-1}P^{-j}\, P^{i}N^{-1}P^{-i}\, S^{-1}$

Military Enigma Key Length

- Key Length (rotor order, rotor positions, plugboard)
  - 60 rotor orders. lg(60)= 5.9 bits.
  - 26*26*26 = 17576 initial rotor positions. lg(17576)= 14.1 bits of key
  - 10 exchanging steckers were specified yielding C(26,2) C(24,2)…C(8,2)/10! = 150,738,274,937,250. lg(150,738,274,937,250)= 47.1 bits as used
  - Bits of key: 5.9 + 14.1 + 47.1 = 67.1 bits
  - Note: plugboard triples entropy of key!
- Rotor Wiring State
  - lg(26!) = 88.4 bits/rotor.
- Total Key including rotor wiring:
  - 67.1 bits + 3 x 88.4 bits = 312.3 bits

Method of Batons

- Applies to Enigma
  - Without plugboard
  - With fast rotor ordering known and only the fast rotor moving
  - With a “crib”
- Let N be the fast rotor and Z the combined effect of the other apparatus, then $N^{-1}ZN(p)=c$.
- Since $ZN(p)=N(c)$, we know the wiring of N and a crib, we can play the crib against each of the 26 possible positions of N for the plaintext and the ciphertext. In the correct position, there will be no “scritches” or contradictions in repeated letters.
- This method was used to “analyze” the early Enigma variants used in the Spanish Civil War and is the reason the Germans added the plugboard. Countermeasure: Move fast rotor next to reflector.

Changes German use of Enigma

1. Plugboard added– 6/30
2. Key setting method – 1/38
3. Rotors IV and V – 12/38
4. More plugs - 1/39
5. End of message key pair encipherment – 5/40

German Key Management before 5/40

- The Germans delivered a global list of keys. This was big advantage in terms of simplicity but introduced a problem.
- Each daily key consisted of a line specifying:
  - (date, rotor order, ring settings, plug settings -10)
- Daily keys were distributed on paper monthly by courier.
- If everyone used the keys for messages, the first letter (and in general the kth letter) in every message would form a mono-alphabet which is easily broken by techniques we've seen.
- To address this weakness, the Germans introduced ephemeral keys as follows:
  1. Operator chose a 3-letter sequence ("indicator").
  2. Operator set rotor positions to indicator and encrypted text *twice*.
  3. Machine rotor positions were reset to indicator position and the message encrypted..

The basic theorems: prelude to the Polish attack

- *Theorem 1:* If S= $(a_1, a_2, \ldots, a_{n1})$ $(b_1, b_2, \ldots, b_{n2})\ldots$ and T is another permutation, then the effect of $T^{-1}ST$, operating from the left, is $T^{-1}ST = (a_1T, a_2T, \ldots, a_{n1}T)$ $(b_1T, b_2T, \ldots, b_{n2}T)\ldots$

- *Theorem 2:* Let S be a permutation of even degree. S can be decomposed into pairs of cycles of equal length if and only if it can be written as the product of two transpositions.

Plan for the Polish attack

- *Define*

  $E(i,j,k) = P^{i}NP^{-i}\ P^{j}MP^{-j}\ P^{k}LP^{-k}\ U\ P^{k}L^{-1}P^{-k}\ P^{j}M^{-1}P^{-j}\ P^{i}N^{-1}P^{-i}$
- Let A= E(1,j,k), B= E(2,j,k), C= E(3,j,k), D= E(4,j,k), E= E(5,j,k), F= E(6,j,k) and suppose the six letter indicator for a message is `ktz svf`. Then,

  □A=k, □D=s; □B=t, □E=v; and □C=z, □F=f, for unknown letters □ □□□□□.

  Since, A= A^{-1}, etc., we obtain t(AD)=s, v(BE)= z(CF).

- The attack proceeds as follows.
  - Use message indicators to construct (AD), (BE) and (CF).
  - Use the knowledge of (AD), (BE) and (CF) to find A, B, C, D, E, F.
- Set
  - Set $Q= MLRL^{-1}M^{-1}$, $U= NP^{-1}QPN^{-1}$, $V= NP^{-2}QP^{2}N^{-1}$, $W= NP^{-3}QP^{3}N^{-1}$, $X= NP^{-4}QP^{4}N^{-1}$, $Y= NP^{-5}QP^{5}N^{-1}$, $Z= NP^{-6}QP^{6}N^{-1}$, $H=NPN^{-1}$.

Plan for the Polish attack - continued

- Note that
  - $U=P^{-1}S^{-1}ASP^{1}$
  - $V=P^{-2}S^{-1}ASP^{2}$
  - $W=P^{-3}S^{-1}ASP^{3}$
  - $X=P^{-4}S^{-1}ASP^{4}$
  - $Y=P^{-5}S^{-1}ASP^{5}$
  - $Z=P^{-6}S^{-1}ASP^{6}$
- Now suppose we have obtained S somehow (say, by stealing it). Then we can calculate:
  - $UV= NP^{-1}(QP^{-1}QP)P^{1}N^{-1}$, $VW= NP^{-2}(QP^{-1}QP)P^{2}N^{-1}$.
  - $WX= NP^{-3}(QP^{-1}QP)P^{3}N^{-1}$, $XY= NP^{-4}(QP^{-1}QP)P^{4}N^{-1}$,
  - $YZ= NP^{-5}(QP^{-1}QP)P^{5}N^{-1}$.
  - $(VW)= H^{-1}(UV)H$, $(WX)= H^{-1}(VW)H$,
  - $(XY)= H^{-1}(WX)H$, $(YZ)= H^{-1}(XY)H$.
- Now we can calculate H and thus N.

Polish (Rejewski) Attack

- Rejewski exploited weakness in German keying procedure to determine rotor wiring
  - Rejewski had ciphertext for several months but no German Enigma.
  - Rejewski had Stecker settings for 2 months (from a German spy via the French in 12/32), leaving 265.2 bits of key (the wirings) to be found. He did.
- Poles determined the daily keys
  - Rejewski catalogued the characteristics of rotor settings to detect daily settings. He did this with two connected Enigmas offset by 3 positions (the “cyclotometer”).
  - In 9/38, when the “message key” was no longer selected from standard setting (the Enigma operator to choose a different encipherment start called the indicator), Rejewski’s characteristics stopped working.
  - Zygalski developed a new characteristic and computation device (“Zygalski sheets”) to catalog characteristics which appeared when 1st/4th, 2nd/5th,3rd/6th ciphertext letters in encrypted message keys (“Females”) were the same.

Calculate (AD), (BE), (CF)

$c=(p)S\ P^{i}NP^{-i}\ P^{j}MP^{-j}\ P^{k}LP^{-k}\ U\ P^{k}L^{-1}P^{-k}\ P^{j}M^{-1}P^{-j}\ P^{i}N^{-1}P^{-I}\ S^{-1}$

- Using the message indicators and:
    - $AD= SP^{1}NP^{-1}QP^{1}N^{-1}P^{3}NP^{-4}QP^{4}N^{-1}P^{-4}S^{-1}$. $(c_1)AD= c_4$.
    - $BE= SP^{2}NP^{-2}QP^{2}N^{-1}P^{3}NP^{-5}QP^{5}N^{-1}P^{-5}S^{-1}$. $(c_2)BE= c_5$.
    - $CF= SP^{3}NP^{-3}QP^{3}N^{-1}P^{3}NP^{-6}QP^{6}N^{-1}P^{-6}S^{-1}$. $(c_3)CF= c_6$.

- We can find AD, BE and CF after about 80 messages.

Calculate A, B, C, D, E, F

- Suppose
  - `AD= (dvpfkxgzyo)(eijmunqlht)(bc)(rw)(a)(s)`
  - `BE= (blfqveoum)(hjpswizrn)(axt)(cgy)(d)(k)`
  - `CF= (abviktjgfcqny)(duzrehlxwpsmo)`
- Cillies
  - syx scw
  - Arises from “aaa” encipherments (look for popular indicators)
  - (as) in A, (ay) in B, (ax) in C, (as) in D, (ac) in E, (aw) in F
  - With Theorem 2, this allows us to calculate A,B,C,D,E,F.
  - Example (C): `(abviktjgfcqny)(duzrehlxwpsmo)`
    - **a**`bviktjgfcqny`
    - **x**`lherzudomspw`
    - `C= (ax)(bl)(vh)(ie)(kr)(tz)(ju)`
      `(gd)(fo)(cm)(qs)(np)(yw)`

Calculate A, B, C, D, E, F

```
A= (as)(bw)(cr)(dt)(vh)(pl)(fq)(kn)(xu)(gm)(zj)(yi)(oe)

B= (dk)(ay)(xg)(tc)(bj)(lh)(fn)(qr)(vz)(ei)(ow)(us)(mp)

C= (ax)(bl)(vh)(ie)(kr)(tz)(ju)(gd)(fo)(cm)(qs)(np)(yw)

D= (as)(bw)(cr)(ft)(kh)(xl)(gq)(zn)(yu)(om)(dj)(vi)(pe)

E= (dh)(xy)(tg)(ac)(qn)(vr)(ez)(oi)(uw)(ms)(bp)(lj)(fh)

F= (co)(qm)(ns)(xp)(aw)(bx)(vl)(ih)(ke)(tr)(jz)(yu)(fd)
```

U, V, W, X, Y, Z

- $A = SPUP^{-1}S^{-1}$ so $U = P^{-1}S^{-1}ASP^{1}$. This and similar equations yield:

- $U = P^{-1}S^{-1}ASP^{1}$
- $V = P^{-2}S^{-1}BSP^{2}$
- $W = P^{-3}S^{-1}CSP^{3}$
- $X = P^{-4}S^{-1}DSP^{4}$
- $Y = P^{-5}S^{-1}ESP^{5}$
- $Z = P^{-6}S^{-1}FSP^{6}$

- S was obtained through espionage.
- $S = (ap)(bl)(cz)(fh)(jk)(qu)$

- Putting this all together, we get U,V,W,X,Y,Z.

U, V, W, X, Y, Z as cycles

```
U=(ax)(bh)(ck)(dr)(ej)(fw)(gi)(lp)(ms)(nz)(oh)(qt)(uy)

V=(ar)(bv)(co)(dh)(fl)(gk)(iz)(jp)(mn)(qy)(su)(tw)(xe)

W=(as)(bz)(cp)(dg)(eo)(fw)(gj)(hl)(iy)(kr)(mu)(nt)(vx)

X=(ap)(bf)(cu)(dv)(ei)(gr)(ho)(jn)(ky)(lx)(mz)(qf)(tw)
```

Calculate (UV), (VW), (WX), (XY), (YZ)

```
UV= (aepftybsnikod)(rhcgzmuvqwljy)
VW= (ydlwnuakjcevz)(ibxopgrsmtvhq)

VW= (ydlwnuakjcevz)(ibxopgrsmtvhq)
WX= (uzftjryehxdsp)(caqvloikgnwbm)

H= (ayuricxqmgovskedzplfwtnjhb)

N:  abcdefghijklmnopqrstuvwxyz
    azfpotjyexnsiwkrhdmvclugbq

N= (a)(bzqhy)(cftvlsmieoknwu)(dpr)(gjx)
```

Turing Bombe - Introduction

- Assume we know all rotor wirings and the plaintext for some received cipher-text. We do not know plugboard, rotor order, ring and indicator.

- We need a crib characteristic that is plugboard invariant.

```
Position   123456789012345678901234
Plain Text OBERKOMMANDODERWEHRMACHT
CipherText ZMGERFEWMLKMTAWXTSWVUINZ
```

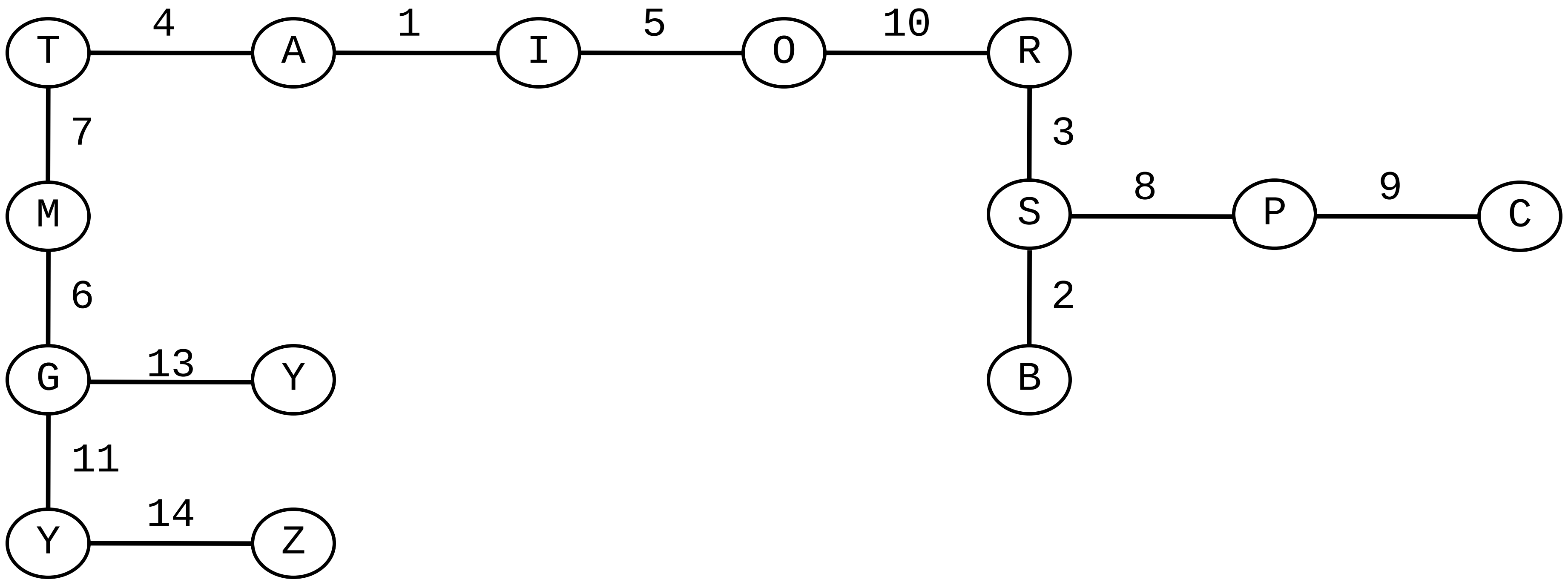
Observe the loop A[9]$\rightarrow$M[7]$\rightarrow$E[14]$\rightarrow$A.

- If M_i is the effect of the machine at position i and S is the Stecker, for the above we have `“E”= (“M”)S` M_7`S` and `(“E”)`$M_7M_9M_{14}$`=“E”`. This return could happen by accident so we use another (E[4]$\rightarrow$R[15]$\rightarrow$W[8]$\rightarrow$M[7]$\rightarrow$E) to confirm as `C(“E”)`$M_4M_{15}M_8M_7$`(“E”)`.

Turing Bombe – the menu

- Want short enough text for no “turnovers”.

```
Position   12345678901234567890l234
Plain Text ABSTIMMSPRUQYY
CipherText ISOAOGTPCOGNYZ
```



Turing Bombe -1

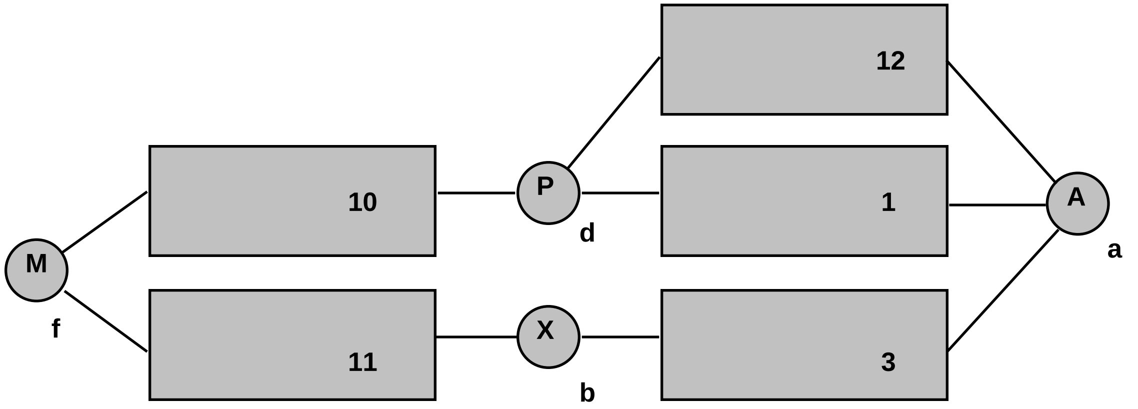
- Each cycle can be turned into a ring of Enigma machines.
- In a ring of Enigmas, ***all*** the S cancel each other out!
- The key search problem is now reduced from 67.5 to 20 bits !!!!
- At 10 msec/test, 20 bits takes 3 hours.
- Turing wanted ~4 loops to cut down on “false alarms.”
- About 20 letters of “crib” of know plaintext were needed to fine enough loops.

- Machines which did this testing were called “Bombe’s”.
- Built by British Tabulating Machine Company.

Courtesy of Carl Ellison

Test Register in Bombes

In the diagram below, each circle is a 26-pin connector and each line a 26-wire cable. The connector itself is labeled with a letter from the outside alphabet while its pins are labeled with letters from the inside alphabet. Voltage on X(b) means that **X** maps to **b** through the plugboard.



X: 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

Courtesy of Carl Ellison

Welchman’s Improvement

- With enough interconnected loops, when you apply voltage to X(b), you will see one of three possibilities on the pins of connector X:

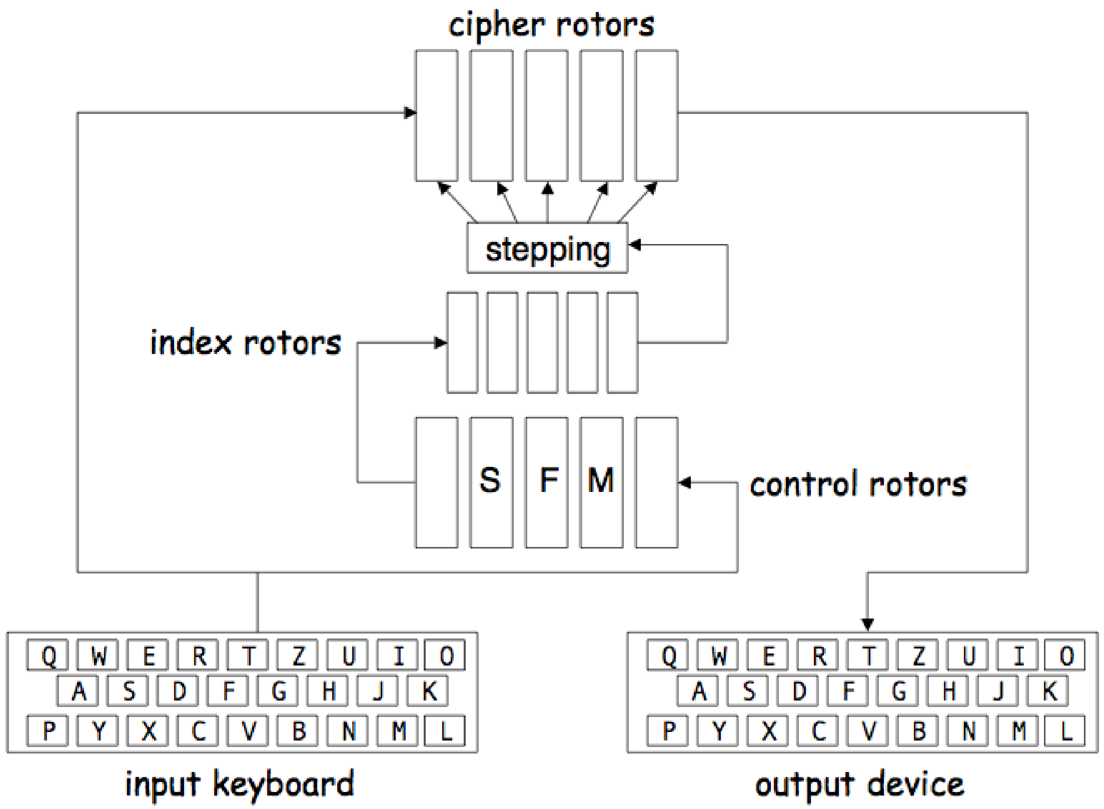
  `01000000000000000000000000` **X** maps to **b**

  `11101111111111111111111111` **X** really maps to **d**

  `11111111111111111111111111` wrong Enigma key
- Gordon Welchman realized that if X(b) then B(x), because the plugboard was a self-inverse (S == S^{-1}).
- His diagonal board wired X(a) to A(x), D(q) to Q(d), etc.
- With that board, the cryptanalyst didn’t need loops -- just enough text
- This cut the size of the required crib in half.

Courtesy of Carl Ellison

Sigaba Wiring Diagram

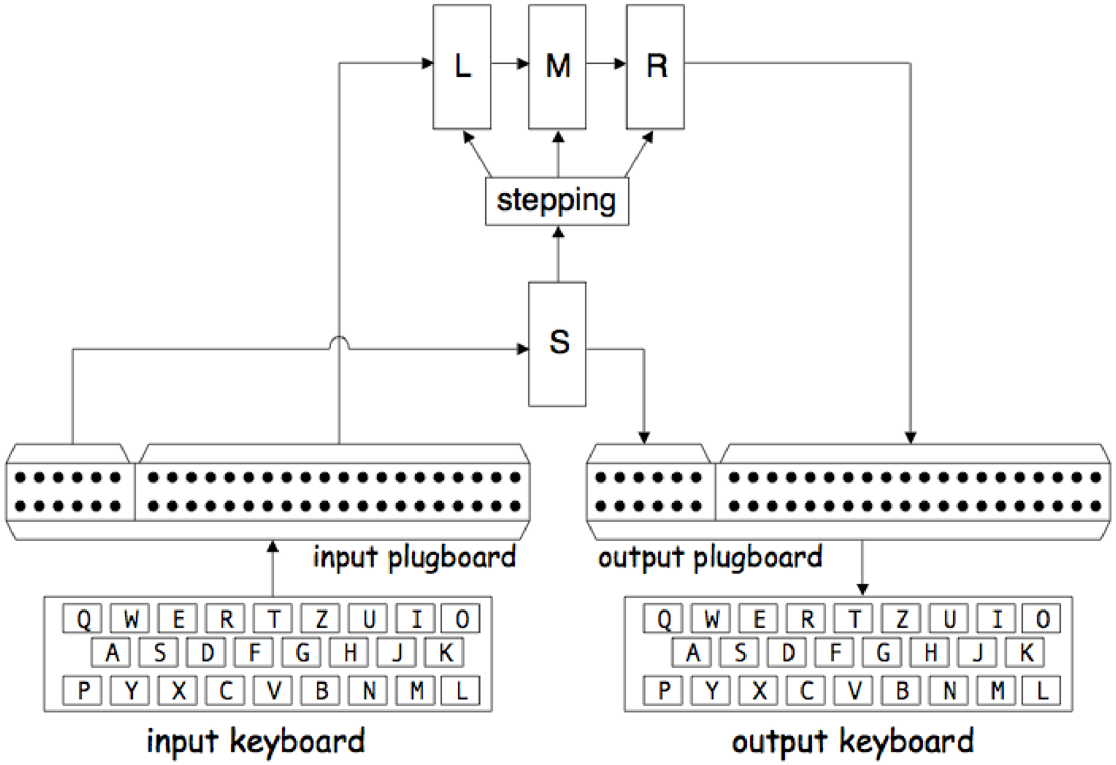


- Control and index rotors determine stepping of cipher rotors

Slide by Mark Stamp

Purple

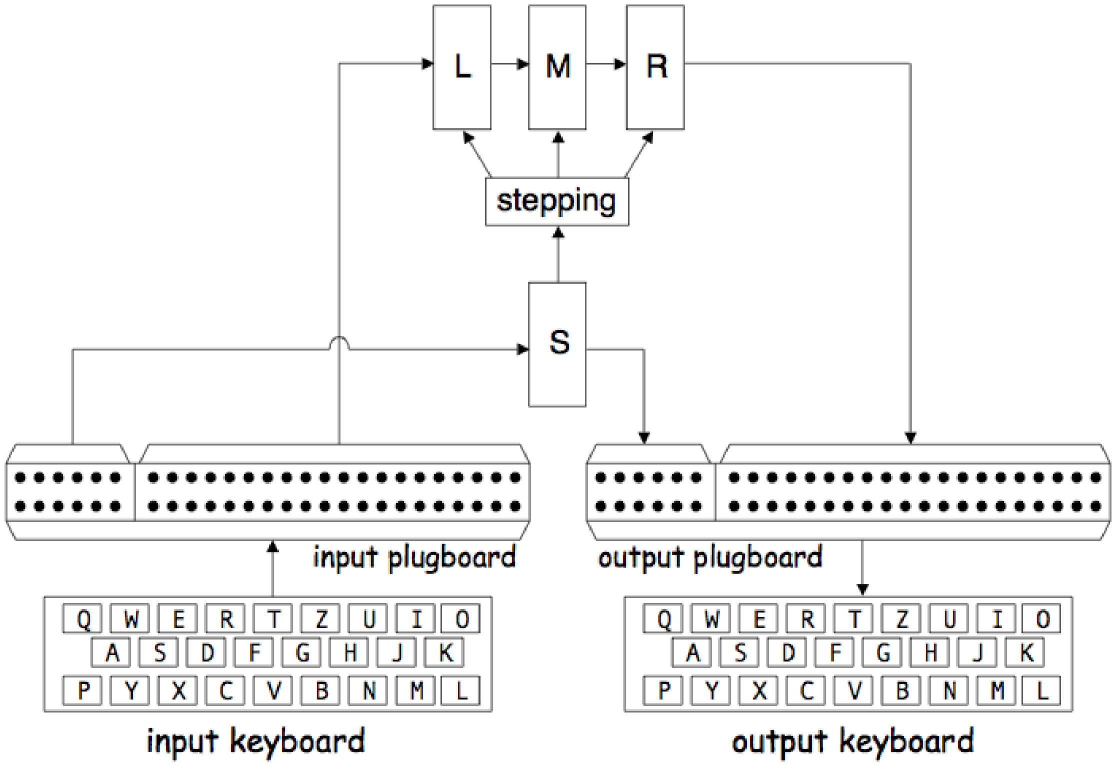
- **Switched** permutations
  - **Not** rotors!!!
- S,L,M, and R are switches
  - Each step, one of the perms switches to a different permutation



Slide by Mark Stamp

Purple

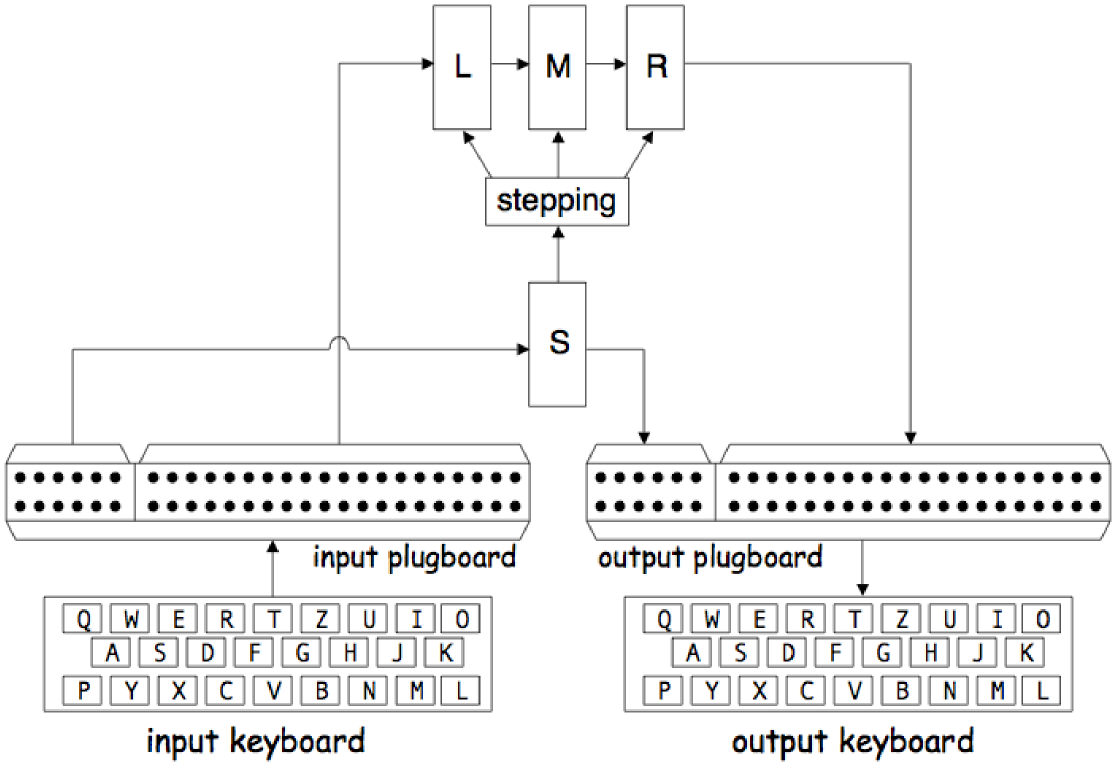
- Input letter permuted by plugboard, then…
- Vowels and consonants sent thru different switches
- The “6-20 split”



Slide by Mark Stamp

Purple

- Switch S
  - Steps once for each letter typed
  - Permutes vowels
- Switches L,M,R
  - One of these steps for each letter typed
  - L,M,R stepping determined by S



Slide by Mark Stamp

End