Classical Cryptography

Basics: monoalphabetic substitution

Karst Koymans

Informatics Institute

University of Amsterdam

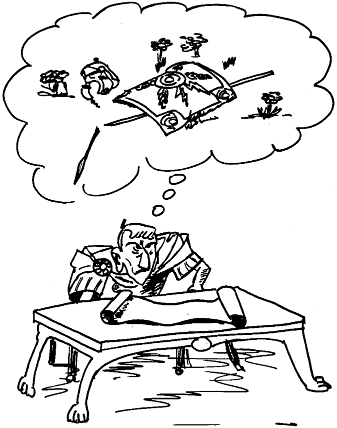
(version 22.4, 2023/02/06 11:06:14 UTC)

Friday, February 10, 2023

Outline

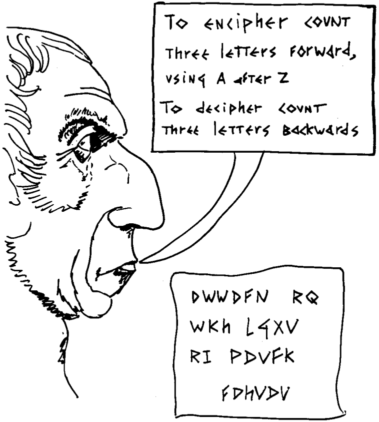
Outline

Caesar wants to hide his plans



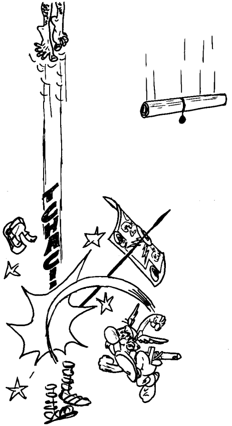
Source: Slides Hans van der Meer

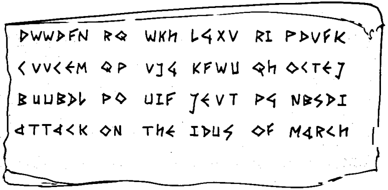
Caesar's cryptosystem



Source: Slides Hans van der Meer

Interception and cryptanalysis





Who notices the peculiarities here?

Source: Slides Hans van der Meer

Caesar encryption

- Caesar encryption is a forward[1] rotation of the alphabet by 3 places

abcdefghijklmnopqrstuvwxyz
DEFGHIJKLMNOPQRSTUVWXYZABC

Figure 1: Rotation by 3 positions

- An example encryption

an example encryption
DQ HADPSOH HQFUBSWLRQ

Figure 2: Encryption of "an example encryption"

[1] although, historically, Suetonius calls it backward

Caesar decryption

- Caesar decryption works by turning around the encryption process

```
DEFGHIJKLMNOPQRSTUVWXYZABC
abcdefghijklmnopqrstuvwxyz
```

Figure 3: Encryption turned around (backward rotation by 3 places)

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
xyzabcdefghijklmnopqrstuvw
```

Figure 4: The same decryption reordered

Outline

Encoding (numbering) the alphabet

	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
modern	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
legacy	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

- Modern mathematics starts counting at 0
- The legacy variant, starting at 1, is equivalent to ordering the alphabet as

  zabcdefghijklmnopqrstuvwxy

- This is because, when rotating the alphabet, we consider $26 = 0$

Outline

Clock arithmetic

$24 = 0$ (or maybe $12 = 0$)

- $\mathbb{Z}_{24} = \mathbb{Z}/24\mathbb{Z} = \{0, 1, 2, \ldots, 23\}$
- $23 + 1 \equiv 24 \equiv 0 \pmod{24}$

Definition ($n \in \mathbb{N}, n > 1, a, b \in \mathbb{Z}$)

$a \equiv b \pmod{n} \iff n \mid (a - b) \iff \exists k \in \mathbb{Z}(k \cdot n = (a - b))$

Theorem

"$. \equiv . \pmod{n}$" is an **equivalence** *relation on $\mathbb{Z}$, in fact a* **congruence**.
$\mathbb{Z}_n = \mathbb{Z}/n\mathbb{Z} = \{0, 1, \ldots, n-1\}$ is the set of integers modulo n,
using the standard representatives for the equivalence classes.

Corollary

Addition and multiplication can be performed $\pmod{n}$ as usual.

Clock arithmetic

Examples

$$22 + 5 \equiv 3 \pmod{24}$$

$$22 \cdot 5 \equiv 110 \equiv 14 \pmod{24}$$

$$-2 \cdot 5 \equiv -10 \equiv 14 \pmod{24}$$

$$2 \cdot 12 \equiv 24 \equiv 0 \pmod{24}$$

$$2 \not\equiv 0 \pmod{24}$$

$$12 \not\equiv 0 \pmod{24}$$

$\mathbb{Z}_{24}$ has **divisors of zero** or **zero divisors**,

which is considered an unwanted property in general.

Clock arithmetic

Convention

$\pmod{n}$ as a function

The function application $a \pmod{n}$ means the unique b such that

$0 \leq b < n$ and $a \equiv b \pmod{n}$, as a relation.

- The use of $\pmod{n}$ both as a binary relation

  as well as a function can be confusing:

$$(a \pmod{n} \equiv a) \pmod{n}$$

$$a \pmod{n} = (a \pmod{n})$$

Who's afraid of zero?

or the AM/PM mess

- Splitting up 24 hours as $2 \cdot 12$ hours the sensible way
  - 0:00 AM (midnight), 1:00 AM, ..., 11:59 AM
  - 0:00 PM (midday, noon), 1:00 PM, ..., 11:59 PM
  - In Japan 00:00 AM (==12:00 PM?) is midnight and 12:00 AM (==00:00 PM) is noon
- Splitting up 24 hours as $2 \cdot 12$ hours the confusing way
  - 12:00 AM (midnight), 12:59 AM, 1:00 AM, ..., 11:59 AM
  - 12:00 PM (midday, noon), 12:59 PM, 1:00 PM, ..., 11:59 PM
  - $12 \equiv 0 \pmod{12}$, but $12 \not\equiv 0 \pmod{24}$, hence using 12 hours here is confusing

Outline

Caesar mathematically

Caesar encryption and decryption

$$\mathcal{E}(p) = (p + 3) \pmod{26} \tag{1}$$

$$\mathcal{D}(c) = (c - 3) \pmod{26} \tag{2}$$

- This works exactly the same with modern and legacy encoding
- Encryption and decryption are **keyless**
- Algorithm must be kept secret

Caesar variants with a key

Let k be a key, where $0 \leq k < 26$. (What happens if $k = 0$?)

Caesar encryption and decryption with key k

$$\mathcal{E}_k(p) = (p + k) \pmod{26} \tag{3}$$

$$\mathcal{D}_k(c) = (c - k) \pmod{26} \tag{4}$$

- Even if the algorithm is known the key protects the encryption
- Since the key space is very small a brute force search is doable
- We call this a **shift cipher** or an **additive cipher**

Outline

Caesar brute force decrypting “VLONY ZILWY”

Caesar brute force decrypting “VLONY ZILWY”

vlony zilwy

Caesar brute force decrypting “VLONY ZILWY”

vlony zilwy

uknmx yhkvx

Caesar brute force decrypting "VLONY ZILWY"

vlony zilwy

uknmx yhkvx

tjmlw xgjuw

Caesar brute force decrypting “VLONY ZILWY”

vlony zilwy

uknmx yhkvx

tjmlw xgjuw

silkv wfitv

Caesar brute force decrypting “VLONY ZILWY”

vlony zilwy

uknmx yhkvx

tjmlw xgjuw

silkv wfitv

rhkju vehsu

Caesar brute force decrypting “VLONY ZILWY”

vlony zilwy

uknmx yhkvx

tjmlw xgjuw

silkv wfitv

rhkju vehsu

qgjit udgrt

Caesar brute force decrypting "VLONY ZILWY"

vlony zilwy

uknmx yhkvx

tjmlw xgjuw

silkv wfitv

rhkju vehsu

qgjit udgrt

pfihs tcfqs

Caesar brute force decrypting “VLONY ZILWY”

vlony zilwy

uknmx yhkvx

tjmlw xgjuw

silkv wfitv

rhkju vehsu

qgjit udgrt

pfihs tcfqs

oehgr sbepr

Caesar brute force decrypting “VLONY ZILWY”

vlony zilwy

uknmx yhkvx

tjmlw xgjuw

silkv wfitv

rhkju vehsu

qgjit udgrt

pfihs tcfqs

oehgr sbepr

ndgfq radoq

Caesar brute force decrypting "VLONY ZILWY"

vlony zilwy

uknmx yhkvx

tjmlw xgjuw

silkv wfitv

rhkju vehsu

qgjit udgrt

pfihs tcfqs

oehgr sbepr

ndgfq radoq

mcfep qzcnp

lbedo pybmo

kadcn oxaln

jzcbm nwzkm

iybal mvyjl

hxazk luxik

gwzyj ktwhj

fvyxi jsvgi

euxwh irufh

Caesar brute force decrypting “VLONY ZILWY”

vlony zilwy

uknmx yhkvx

tjmlw xgjuw

silkv wfitv

rhkju vehsu

qgjit udgrt

pfihs tcfqs

oehgr sbepr

ndgfq radoq

mcfep qzcnp

lbedo pybmo

kadcn oxaln

jzcbm nwzkm

iybal mvyjl

hxazk luxik

gwzyj ktwhj

fvyxi jsvgi

euxwh irufh

dtwvg hqteg

csvuf gpsdf

brute force

aqtsd enqbd

zpsrc dmpac

yorqb clozb

xnqpa bknya

wmpoz ajmxz

Caesar brute force decrypting “VLONY ZILWY”

vlony zilwy

uknmx yhkvx

tjmlw xgjuw

silkv wfitv

rhkju vehsu

qgjit udgrt

pfihs tcfqs

oehgr sbepr

ndgfq radoq

mcfep qzcnp

lbedo pybmo

kadcn oxaln

jzcbm nwzkm

iybal mvyjl

hxazk luxik

gwzyj ktwhj

fvyxi jsvgi

euxwh irufh

dtwvg hqteg

csvuf gpsdf

brute force

aqtsd enqbd

zpsrc dmpac

yorqb clozb

xnqpa bknya

wmpoz ajmxz

Outline

Outline

Monoalphabetic substitution

Definition

A monoalphabetic substitution is the systematic replacement of letters by other letters in a one-to-one way.

Example monoalphabetic encryption and decryption

abcdefghijklmnopqrstuvwxyz	ABCDEFGHIJKLMNOPQRSTUVWXYZ
DJEHKVNIOLARUQXPYWGTCSMFZB	kzuacxsdhbejwgipnlvtmfroqy

This example was generated using a Nomcom procedure with pool size 26 on input “1 2 ... 16”[2]

[2] see RFC 3797

Intermezzo: a real example (Spanish)

```
ADHRF SID QINVJX IH XDNAJIXJHAD
VFH YINEVJ YDZEVJHJ PFO J TTDPJX
J YE PDVEHJ JTTE DNAJ HFVWD DTTJ
DN YIO QFHEAJ O NEYLJAEVJ DNLDXF
WJVDXIHJ EYLXDNEFH QIDHJ
```

1. 1-letter word **a**, **y** or sometimes **o**
2. 2-letter word **u.** usually **un**
3. 3-letter word **..e** usually **que**
4. 4-letter pattern ABBC usually **alli** or **ella**
5. Doubled starting letter mostly **l** as in **llegar**, **llevar**, **lleno**, **lluvia**

Generating a monoalphabetic substitution from a keyword

abcdefghijklmnopqrstuvwxyz
KEYWORDABCFGHIJLMNPQSTUVXZ

Figure 5: Using “KEYWORD” as the keyword

abcdefghijklmnopqrstuvwxyz
REPATDLSBCFGHIJKMNOQUVWXYZ

Figure 6: Using “REPEATED LETTERS” as the keyword/keyphrase

Generating a monoalphabetic substitution using decimation

```
abcdefghijklmnopqrstuvwxyz
EJOTYDINSXCHMRWBGLQVAFKPUZ
```

Figure 7: Encryption using a **multiplicative cipher** (legacy)

```
abcdefghijklmnopqrstuvwxyz
AFKPUZEJOTYDINSXCHMRWBGLQV
```

Figure 8: Encryption using a **multiplicative cipher** (modern)

- A multiplicative cipher is also called a **decimation**

Decryption of these multiplicative ciphers

ABCDEFGHIJKLMNOPQRSTUVWXYZ
upkfavqlgbwrmhcxsnidytojez

Figure 9: Decryption of the **multiplicative cipher** (legacy)

ABCDEFGHIJKLMNOPQRSTUVWXYZ
avqlgbwrmhcxsnidytojezupkf

Figure 10: Decryption of the **multiplicative cipher** (modern)

- The encryption factor was 5. What is the decryption factor?

Mathematical description of decimation

Multiplicative encryption and decryption

$$\mathcal{E}_e(p) = ep \pmod{26} \tag{5}$$

$$\mathcal{D}_d(c) = dc \pmod{26} \tag{6}$$

- There is now a difference between modern and legacy encoding
- Modern encoding works best for programming
- d is the **multiplicative inverse**[3] of e

[3] Does this always exist?

Outline

Euclid

Source: https://cdpn.io/dloader/fullpage/BwvLBB

Greatest common divisor

An example of Euclid's algorithm

We want to find the gcd (greatest common divisor) of 49 and 35:

Euclid's reduction

$$49 = 1 \cdot 35 + 14 \implies \gcd(49, 35) = \gcd(35, 14)$$

$$35 = 2 \cdot 14 + 7 \implies \gcd(35, 14) = \gcd(14, 7)$$

$$14 = 2 \cdot 7 + 0 \implies \gcd(14, 7) = \gcd(7, 0) = 7$$

Euclid's reversal

$$7 = 35 - 2 \cdot 14 \quad \wedge \quad 14 = 49 - 1 \cdot 35$$

$$\begin{aligned} 7 &= 35 - 2 \cdot (49 - 1 \cdot 35) \\ &= -2 \cdot 49 + 3 \cdot 35 \end{aligned}$$

Greatest common divisor

Euclid's algorithm

Theorem

For all $a, b \in \mathbb{Z}$ we can (effectively) find $p, q \in \mathbb{Z}$ such that

$$gcd(a, b) = p \cdot a + q \cdot b$$

Finding p and q can be done using Euclid's algorithm and reversal.

Definition

a and b are called **relatively prime** iff $\gcd(a, b) = 1$.

Theorem

If a and b are relatively prime (the extended) Euclid's algorithm calculates p and q such that

$$p \cdot a + q \cdot b = 1$$

Application to decimation

In our example we had $e = 5$ and we want to find its inverse *d* modulo 26.

Calculation of inverse of 5 modulo 26

$$26 = 5 \cdot 5 + 1 \implies 1 \cdot 26 + (-5) \cdot 5 = 1$$

So the inverse of 5 modulo 26 is -5 (or 21).

- This explains why the decryption described earlier is indeed just a decimation with factor 21
- A decimation's inverse is another decimation, just with a different multiplication factor.
  - What happens if *e* and 26 are not relatively prime?

Outline

Combining multiple ciphers

- Combining two shift ciphers with key k_1 and k_2
  - Result is shift cipher with key $k_1 + k_2 = k_2 + k_1$
- Combining two decimations with key e_1 and e_2
  - Result is decimation with key $e_2 e_1 = e_1 e_2$
- Combining a decimation with key e and a shift with key k
  - First decimate, then shift gives the **affine cipher**
    defined by $\mathcal{E}_{e,k}(p) = ep + k \pmod{26}$
  - First shift, then decimate gives the cipher
    defined by $\mathcal{E}'_{e,k}(p) = e(p+k) \pmod{26}$
    or $\mathcal{E}'_{e,k}(p) = ep + ek = \mathcal{E}_{e,ek} \pmod{26}$, just another affine cipher

Legacy and modern encoding for affine ciphers

- Suppose we have affine cipher $\mathcal{E}_{e,k}(p) = ep + k \pmod{26}$
- Let d be the multiplicative inverse of $e \pmod{26}$
- For a given character C and shift amount n
  - Let $C + n$ be the result of a shift cipher encryption of character C with shift n
  - Let $L(C)$ be the result of the affine encryption using $\mathcal{E}_{e,k}$ of C in legacy encoding
  - Let $M(C)$ be the result of the affine encryption using $\mathcal{E}_{e,k}$ of C in modern encoding
- Then we can deduce the following relationships
  - $L(C) = M(C+1) - 1$ for all C
  - $L(C) = M(C) + (e-1)$ for all C
  - $L(C) = M(C + (1-d))$ for all C

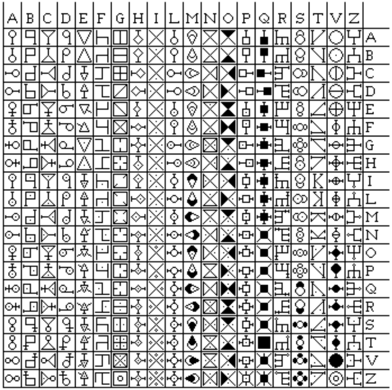
Outline

Extending the "alphabet"

- Until now substitutions are **monographic**
  - One letter of the alphabet is replaced with just one other letter
- What happens if we "extend the alphabet" (make it **polygraphic**)?
  - For instance replace a combination of two letters of the alphabet by another combination of two letters (hence using **digraphs**)
  - Effectively this extends our alphabet from 26 to $26 \cdot 26 = 676$ "letters" (or symbols, atoms, literals, ...)
  - The number of possible (monoalphabetic) substitutions increases from $26! = 403291461126605635584000000$ to $676! \approx 1.8837 \cdot 10^{1621}$

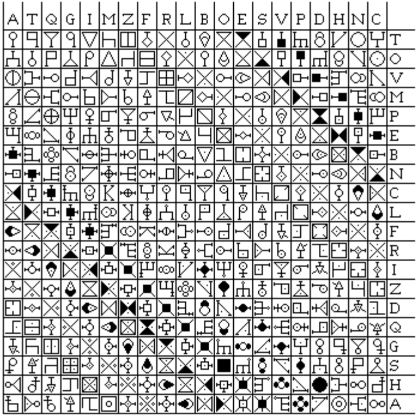
Outline

Giovanni Battista della Porta’s digraph encryption



Source: http://www.quadibloc.com/crypto/pp010302.htm
(Can you spot anomalies?)

Giovanni Battista della Porta’s digraph encryption (better variant)



Source: Slides Hans van der Meer

An example digraph substitution

	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
A	LZ	SW	BH	YJ	YR	WP	BC	FB	FW	XH	DY	MV	KC	UL	CJ	FJ	XW	BR	AD	JP	BJ	PM	JW	IU	OU	DE
B	AJ	GE	KT	AP	TN	VO	GY	CT	JS	OB	YM	MH	WJ	PF	PA	TA	IF	NR	GG	PV	LH	NX	KX	ST	UT	RP
C	LW	KW	DO	QF	JN	LX	DI	TL	DR	RM	SS	HF	RB	QU	UJ	KR	MY	GO	WF	TG	RS	YQ	SC	FI	CR	HK
D	UV	FP	PS	XZ	EV	GR	SV	KF	ZX	WL	RU	WO	YZ	JJ	NJ	VJ	IT	QT	XG	AS	KE	WE	ND	HS	YC	UH
E	AO	CZ	CI	SI	BV	OM	ZO	LE	GD	LB	OI	UK	RC	DK	PZ	YX	KJ	ZT	EM	BS	IZ	XL	RF	WA	YW	EL
F	OC	RI	SP	FY	VH	QE	SE	FC	IK	NZ	RG	LN	TX	NM	SD	JB	UQ	XY	ZG	ML	AV	JC	QM	PQ	AB	ZF
G	MD	VE	FX	MW	OD	PJ	XX	HT	IC	LC	NH	ZD	GC	YY	VP	YA	PC	BE	JF	DS	QK	SX	EQ	ET	YD	JH
H	BT	TK	PR	KY	EC	AN	HZ	SO	YV	MF	ES	YP	FU	AK	NI	SJ	YT	LY	TF	KV	NV	XV	DJ	WX	OO	QB
I	WR	GK	IE	QH	EZ	OY	MU	MT	LA	BP	HA	NM	TJ	QJ	AL	EE	SU	GA	HI	MG	YO	GW	KS	AY	JE	NO
J	VG	ZY	UE	FM	EH	FR	ZW	CA	DN	WD	KD	AU	GP	YS	XM	MR	NC	BQ	HC	NS	NN	ZJ	GJ	VB	RA	TH
K	KQ	UR	VQ	AT	OA	YI	FS	RJ	LT	JD	KI	PG	AC	MI	CD	BG	TZ	PH	OT	WQ	IH	LK	OK	XE	HY	CX
L	YE	VX	GS	VY	IM	HW	HB	JX	NE	ZI	IB	HL	BI	QO	VK	AH	LL	VT	YB	DL	ZC	QI	JA	DH	UY	ZH
M	HU	EW	UC	IJ	UO	SQ	OR	EP	ZE	MX	KL	IQ	TS	QZ	BM	TI	JV	VD	XS	OH	IX	TV	TB	QN	UW	KN
N	LM	CB	SK	EY	PO	FQ	LG	MS	RK	VS	RW	CL	II	RO	ZR	NP	HX	RN	BF	IV	DX	XI	UG	BX	JM	AQ
O	TQ	XN	SH	ZS	WK	OX	WU	HH	MQ	PT	GL	QA	EX	PX	ZB	HJ	VW	SB	PL	DB	NA	CM	UX	IA	JK	LU
P	XD	GM	TC	FG	EJ	FN	WT	NF	OG	QY	DZ	NB	NU	IN	ZV	HM	CS	JU	WV	QG	FH	RQ	TE	DA	GH	AF
Q	YG	DV	EF	HV	TU	HR	LJ	CQ	FK	VC	GF	FZ	ER	XK	NW	XU	VA	ED	MN	UI	RL	GX	WH	WS	TM	OW
R	OS	XR	ID	SG	CY	TY	KG	ZN	YL	KZ	OJ	GU	VF	VR	BD	JO	GV	ZU	FF	WG	XF	GZ	KP	KU	QD	JT
S	RY	GQ	ZZ	HP	CC	HQ	UF	AD	PK	DW	XQ	DU	RH	DC	GN	QR	DM	MK	SF	RZ	MC	FT	BZ	LQ	IO	LO
T	YF	BA	UU	YN	TR	LD	WB	NQ	TW	VN	RD	FA	YU	OP	OQ	LR	FL	JI	JZ	HO	QQ	QC	GI	QW	KH	MA
U	JQ	XO	CH	EA	SY	XJ	IG	PD	ZL	LF	LP	KO	JY	ZP	UD	KA	TD	NG	ZQ	CF	AI	XT	HD	XB	UB	CW
V	XC	EI	BU	VV	AX	DF	MZ	VU	VM	RV	UP	PN	WC	FE	DT	IL	ZM	CU	EK	WZ	OF	LS	BL	IS	XA	BB
W	LI	FO	KM	JR	CV	QP	EG	WN	UA	NT	AG	UN	KK	US	WY	MP	SL	MB	BK	KB	AR	YH	DD	OE	DG	VI
X	AE	FD	ZK	SA	QX	SM	HE	CE	ZA	QV	IY	CN	PY	HN	JG	XP	AZ	UZ	BN	BW	PI	MO	AW	QL	DP	HG
Y	RX	NY	TO	MJ	SR	PE	BO	TT	BY	OV	WM	VZ	GT	CO	JL	GB	SN	NK	OL	PU	EU	RE	PP	RT	AM	CG
Z	ON	ME	IP	PB	WI	EB	LV	PW	EN	VL	NL	AA	QS	WW	RR	SZ	DQ	UM	CP	TP	IW	YK	CK	OZ	FV	IR

Source: Slides Hans van der Meer

(Can you spot anomalies?)

Playfair square with keyword (Charles Wheatstone, 1854)

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

Figure 11: Playfair square (keyword STRANDBAL)

Source: Slides Hans van der Meer

Playfair and Wheatstone

Lord Lyon Playfair

Charles Wheatstone

Source: `https://en.wikipedia.org/wiki/Lyon_Playfair,_1st_Baron_Playfair`

Playfair (row based) substitutions

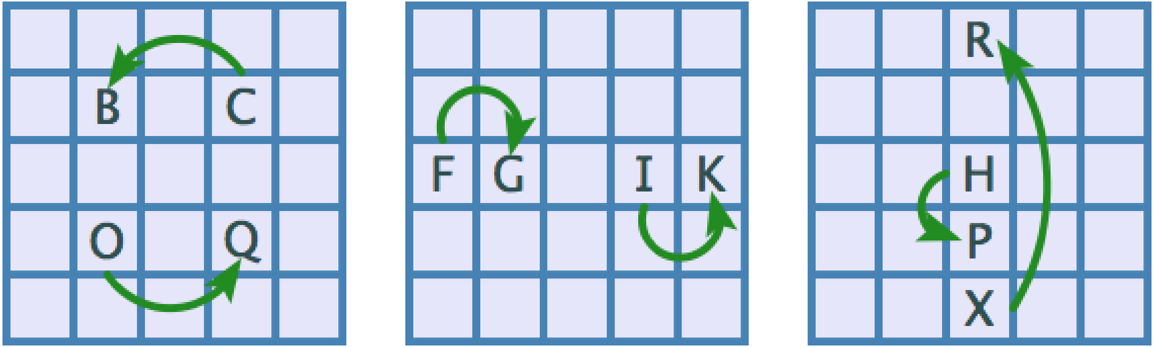


Figure 12: Playfair encryption (OC→QB; FI→GK; HX→PR)

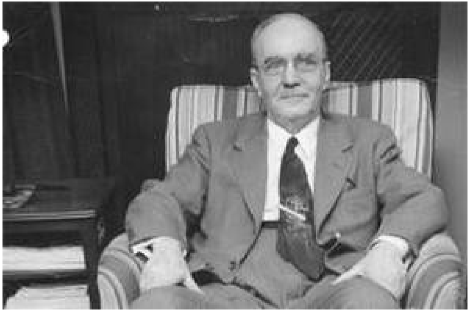
Source: Slides Hans van der Meer

Playfair repeated letters and final single letter

- Treatment of pairs consisting of the same letter pattern “ss”
  - Replace ss by sXs and recreate pairs, if s is not X
  - Replace XX by XQX and recreate pairs
- Treatment of single final letter “f”
  - Replace f by fX, if f is not X
  - Replace X by XQ
- An alternative would have been to use diagonals
  - How?

Outline

Lester S. Hill



Source: https://en.wikipedia.org/wiki/Lester_S._Hill

The (affine) Hill cipher

- Based on linear algebra
- Considers polygraphs as vectors
- An affine cipher built from
  - An (invertible) matrix
  - A translation vector
  - All modulo the size of the base alphabet

$$\begin{pmatrix} 3 & 5 \\ 6 & 1 \end{pmatrix} \begin{pmatrix} 10 \\ 1 \end{pmatrix} + \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 8 \\ 10 \end{pmatrix} \pmod{26}$$

Decrypting the Hill cipher uses inverse matrix

- Encryption

$$\mathcal{E}(p_1, p_2) = \begin{pmatrix} 3 & 5 \\ 6 & 1 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} + \begin{pmatrix} -1 \\ 1 \end{pmatrix} \pmod{26}$$

- Decryption

$$\begin{aligned} \mathcal{D}(c_1, c_2) &= \begin{pmatrix} -1 & 5 \\ 6 & -3 \end{pmatrix} \left[\begin{pmatrix} c_1 \\ c_2 \end{pmatrix} + \begin{pmatrix} 1 \\ -1 \end{pmatrix} \right] \pmod{26} \\ &= \begin{pmatrix} -1 & 5 \\ 6 & -3 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} + \begin{pmatrix} -6 \\ 9 \end{pmatrix} \pmod{26} \end{aligned}$$