COS 433/Math 473: Cryptography

Mark Zhandry

Princeton University

Fall 2020

What is Cryptography?

What is Cryptography

Concise Oxford English Dictionary: *“the art of writing or solving codes”*

Merriam-Webster: *“the enciphering and deciphering of messages in secret code or cipher”*

Wikipedia: *“the practice and study of techniques for secure communication in the presence of third parties called adversaries”*

<u>None of these capture the true breadth of the field</u>

My Definition

Cryptography is about using secrets to solve interesting tasks

(still doesn’t capture everything)

A Long & Rich History

Dates back almost 4000 years

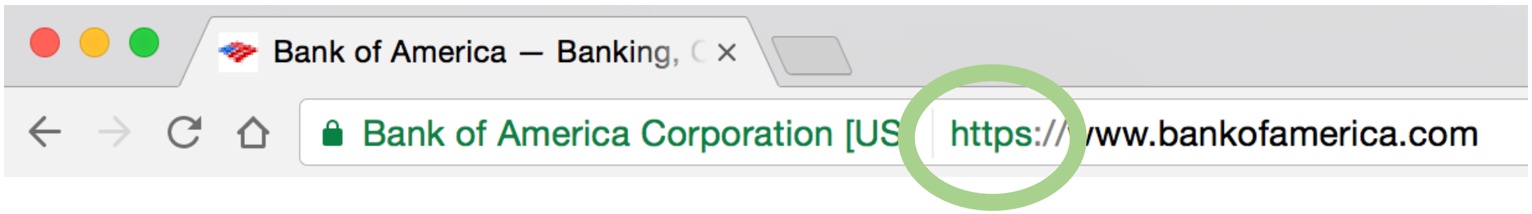
Important historical consequences

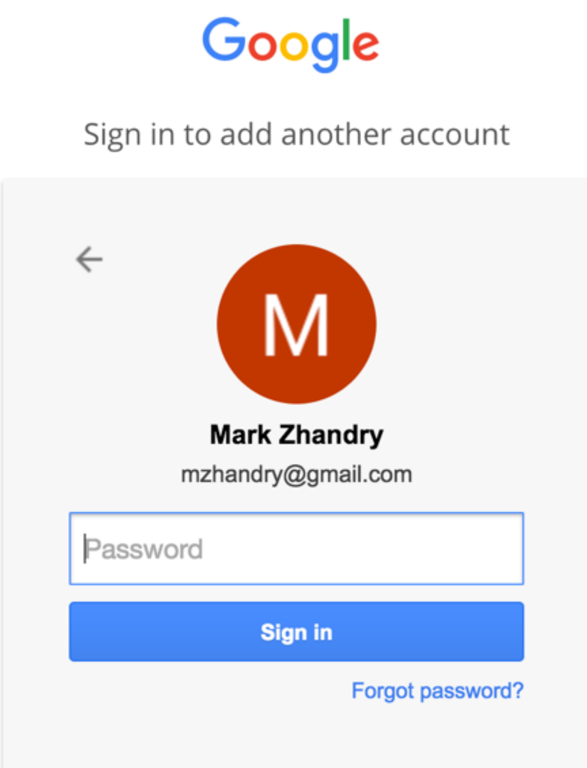
- 1587 – Babington Plot
- WWI – Zimmermann Telegram
- WWII – Enigma

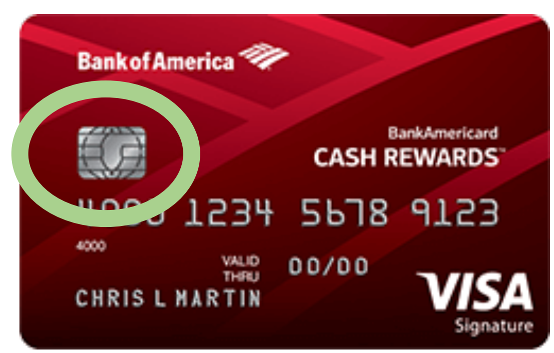
Intimately tied to development of modern computer

- First program written for Atlas supercomputer
- First magnetic core memories, high-speed tape drives, all-transistor computers, desktop-sized computers, remote workstations all built based on NSA orders

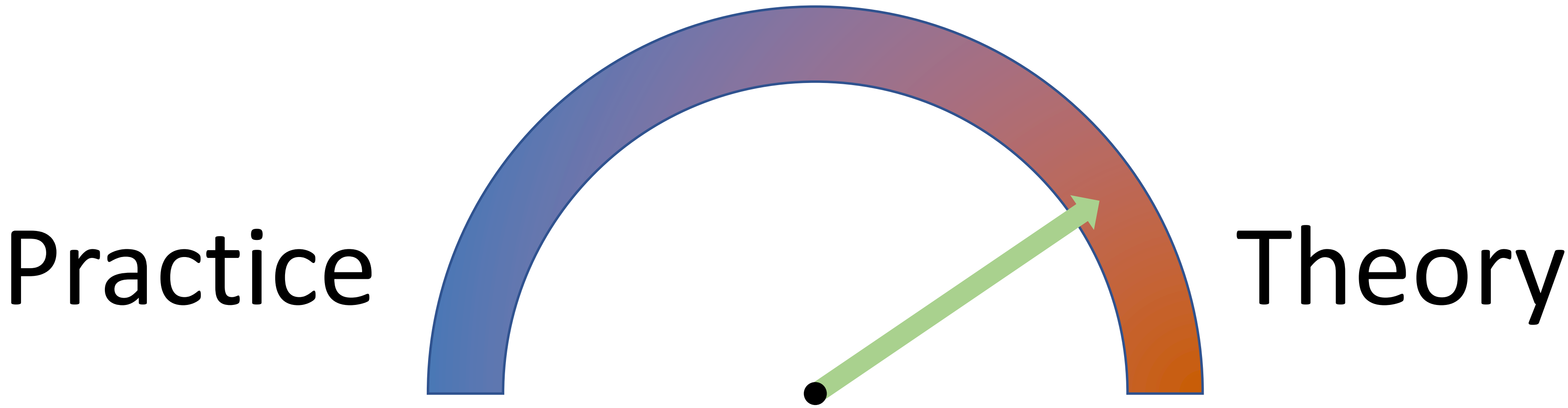
Cryptography Is Everywhere







COS 433



Inherent to the study of crypto
- Working knowledge of fundamentals is crucial
- Cannot discern security by experimentation
- Proofs, reductions, probability are necessary

COS 433

What you should expect to learn:

- Foundations and principles of modern cryptography
- Core building blocks
- Applications

Bonus:

- Debunking some Hollywood crypto

  email me scenes from movies/shows!
- Better understanding of crypto news

COS 433

What you will **not** learn:

- Hacking
- Implementing crypto
- How to design secure **systems**
- Viruses, worms, buffer overflows, etc

Administrivia

Course Information

Instructor: Mark Zhandry
TAs: Udaya Gai
Anunay Kulshrestha

Lectures: TuTh 11:00-12:20pm, Zoom

Webpage: cs.princeton.edu/~mzhandry/2020-Fall-COS433/

Office Hours: please fill out HW0 poll

Canvas

Main channel of communication

- Course announcements
- Submit assignments

Ed Discussion:

- Discuss homework problems with other students
- Ask content questions to instructors, other students
- Find project teams ($\leq$4 people)

Prerequisites

- Ability to read and write mathematical proofs
- Familiarity with algorithms, analyzing running time, proving correctness, O notation
- Basic probability (random variables, expectation)

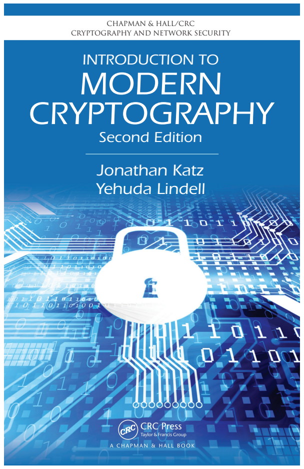
Helpful:
- Familiarity with NP-Completeness, reductions
- Basic number theory (modular arithmetic, etc)

Reading

No required text

But highly recommend:

Introduction to Modern Cryptography
by Katz, Lindell



For each lecture, page numbers for 2nd edition will be posted on course website

Grading

40% Homeworks

- ~1 every two weeks (6 total)
- **1 dropped homework**
- **2 late days per assignment**
- Only typed solutions, submission instructions TBA
- Collaboration encouraged, but write up own solutions

30% Projects

- More details soon

30% Take-home Final

- Individual

Classroom Policies

Please stop me if you have any questions

(Preferably by “raising hand”)

Lectures/slides will be recorded and made available

- I don’t take attendance

Feel free to call me “Mark”, “Professor”, “Hey You”, etc, though “Mark” is preferred

Approximate Course Outline

Week 1: Pre-modern crypto (≤ ~1950s)

Weeks 2-6: Foundations of modern cryptography

- Crypto theory
- Symmetric key cryptography

Weeks 7-12: Public key cryptography

- Number theory

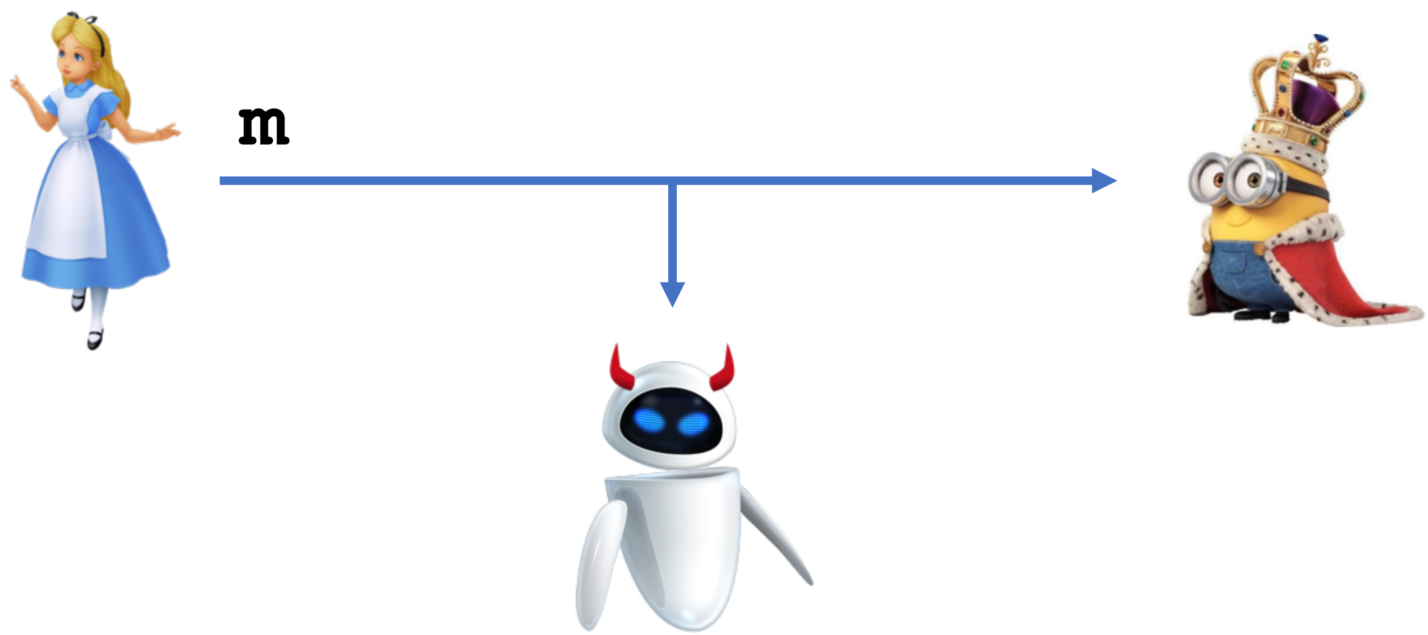
<u>Today</u>

"Pre-modern" Crypto Part I: Pencil & Paper Ciphers

Pre-modern Cryptography

1900 B.C. – mid 1900's A.D.

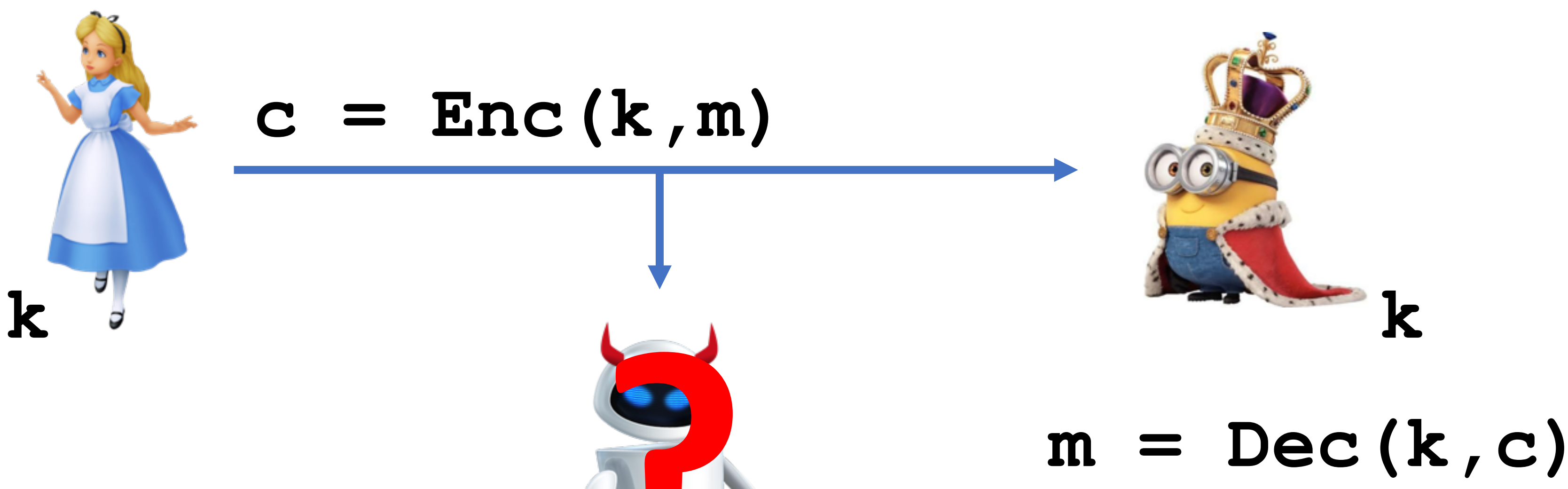
With few exceptions, synonymous with **encryption**



Pre-modern Cryptography

1900 B.C. – mid 1900's A.D.

With few exceptions, synonymous with **encryption**



For our discussions, assume **Enc**, **Dec** known, only **k** is secret

Ancient Crypto

1900 BC, Egypt

1500 BC, Mesopotamia

50 B.C. – Caesar Cipher

Used by Julius Caesar

Alphabet shift by 3

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
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

Example:

plaintext: **super secret message**

ciphertext: **VXSHU VHFUHW PHVVDJH**

Caesar not a true cipher: what's the secret key?

Generalization: Shift Ciphers

Shift by fixed, secret increment (**`k = 0,…,25`**)

Some examples:
- Shift by 1: Augustus Caesar; Jewish mezuzah
- Shift by 3: Caesar Cipher
- Shift by 13: ROT13

Sometimes also called “Caesar ciphers”

Security of Shift Ciphers?

Problem: only 26 possibilities for key

“Brute force” attack:

- Try all 26 possible shifts
- For each shift, see if something sensible comes out

Example Brute Force Attack

Ciphertext: **HJETG HTRGTI BTHHPVT**

Key	Plaintext
0	HJETG HTRGTI BTHHPVT
1	IKFUH IUSHUJ CUIIQWU
2	JLGVI JVTIVK DVJJRXV
3	KMHWJ KWUJWL EWKKSYW
4	LNIXK LXVKXM FXLLTZX
5	MOJYL MYWLYN GYMMUAY
6	NPKZM NZXMZO HZNNVBZ
7	OQLAN OAYNAP IAOOWCA
8	PRMBO PBZOBQ JBPPXDB
9	QSNCP QCAPCR KCQQYEC
10	RTODQ RDBQDS LDRRZFD
11	SUPER SECRET MESSAGE
12	TVQFS TFDSFU NFTTBHF

Key	Plaintext
13	UWRGT UGETGV OGUUCIG
14	VXSHU VHFUHW PHVVDJH
15	WYTIV WIGVIX QIWWEKI
16	XZUJW XJHWJY RJXXFLJ
17	YAVKX YKIXKZ SKYYGMK
18	ZBWLY ZLJYLA TLZZHNL
10	ACXMZ AMKZMB UMAAIOM
20	BDYNA BNLANC VNBBJPN
21	CEZOB COMBOD WOCCKQO
22	DFAPC DPNCPE XPDDLRP
23	EGBQD EQODQF YQEEMSQ
24	FHCRE FRPERG ZRFFNTR
25	GIDSF GSQFSH ASGGOUS

Security of Shift Ciphers?

Problem: only 26 possibilities for key

"Brute force" attack:

- Try all 26 possible shifts
- For each shift, see if something sensible comes out

To avoid brute force attacks, need large key space

- On modern hardware, typically need #(keys) $\geq 2^{80}$

(Usually choose at least 2^{128}, 2^{256})

Generalization: Substitution Ciphers

Apply fixed permutation to plaintext letters

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

Example:

plaintext: **super secret message**

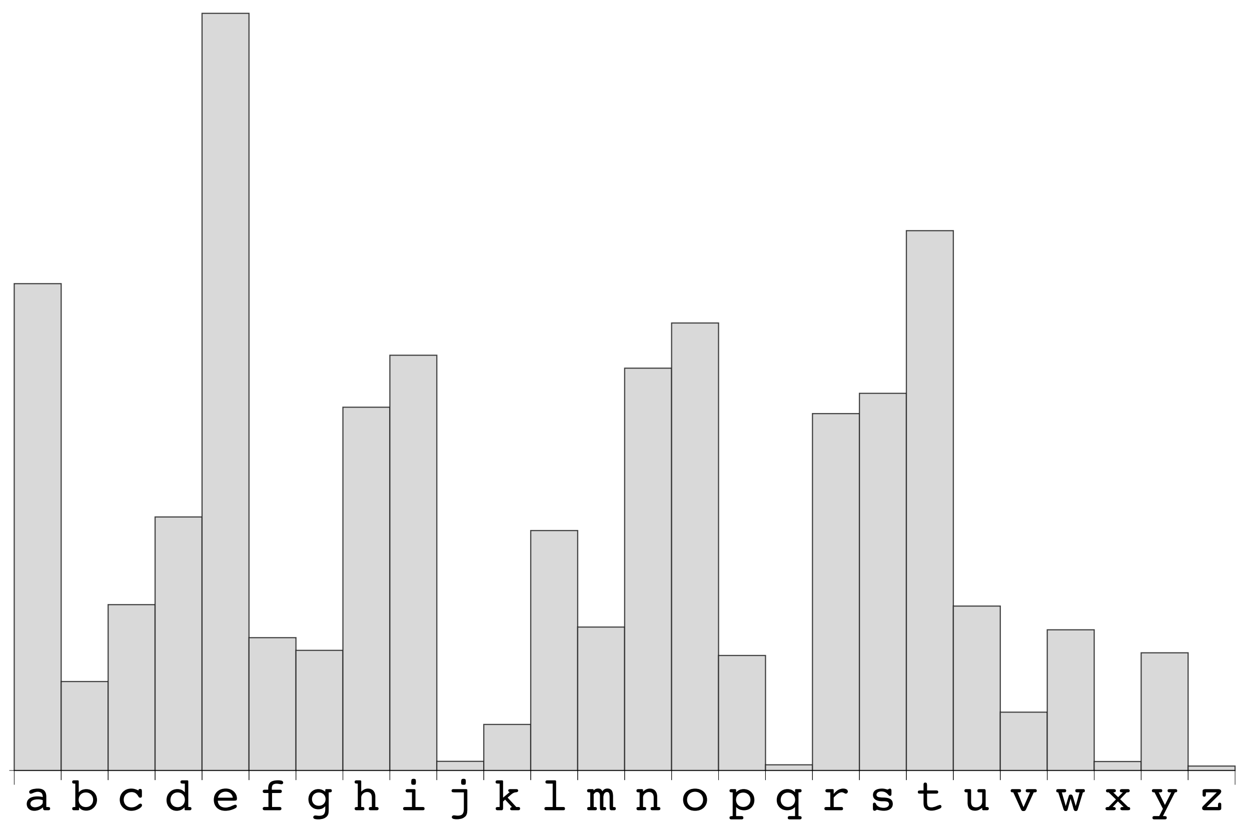
ciphertext: **ARQYV AYSVYX EYAAFJY**

Number of possible keys?

$26! \approx 2^{88}$ ➡ brute force attack expensive

800's A.D. – First Cryptanalysis

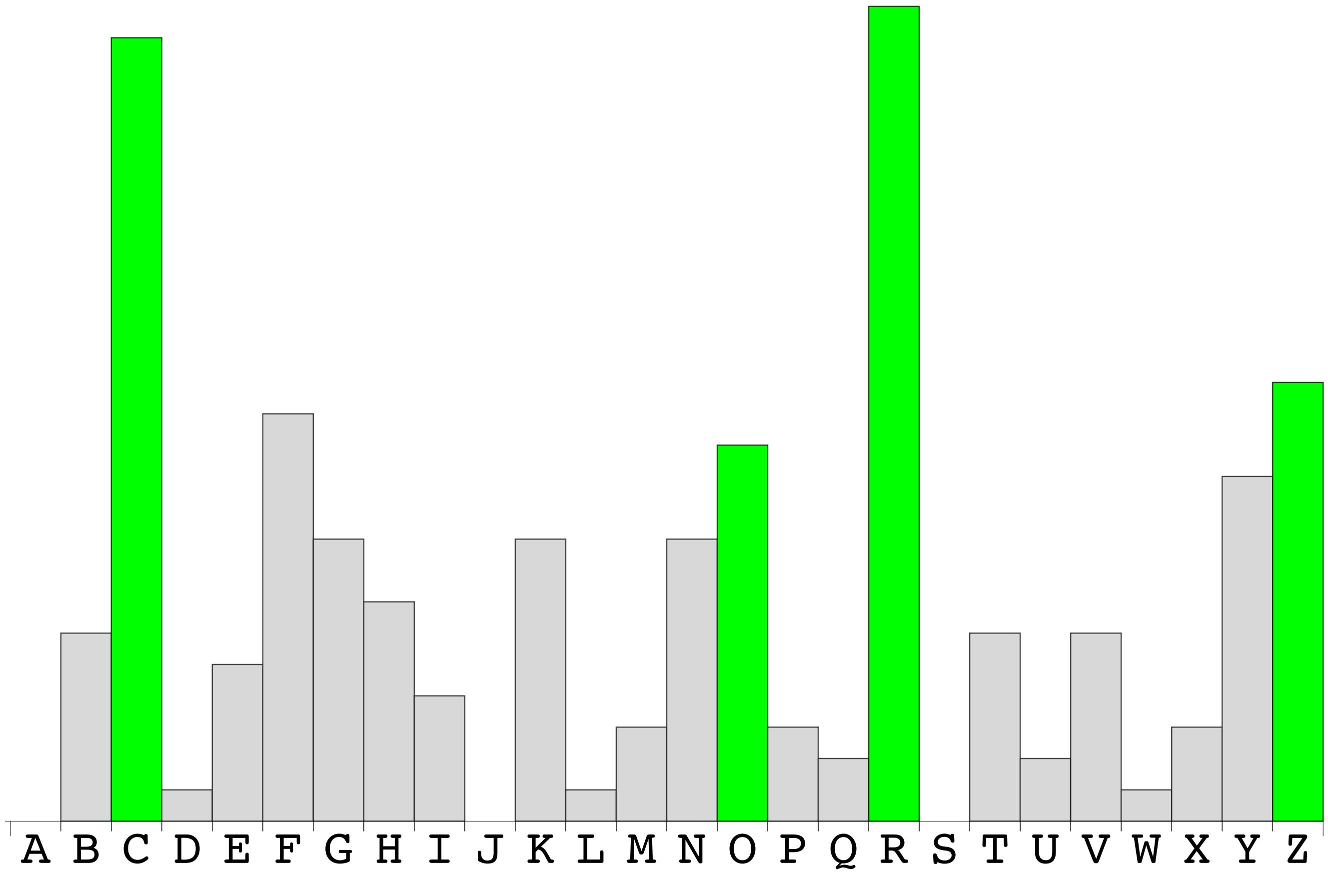
Al-Kindi – Frequency Analysis: some characters are more common than others



Example

BOFC HNR Z NHMNCYCHCYOF KYIVRG CO RFKOBR NRFNYCYPR BZCZ, RPRF CVOHXV CVRE ZGR GRNYTYRFC CO Z MGHCR WOGKR ZCCZKU. YFBRRB, ME KOHFCYFX TRCCRGN ZFB KODIZGYFX CO CEIYKZT CRQC, EOH KZF GRKOPRG CVR ITZYFCRQC ZN LRTT ZN CVR URE

Example



Reasonable conjecture:

e→R, t→C, a→Z, o→O

Example

BoFt HNe a NHMNtYtHtYoF KYIVeG to eFKoBe
NeFNYtYPe Bata, ePeF tVoHXV tVeE aGe
GeNYTYeFt to a MGHte WoGKe attaKU.
YFBeeB, ME KoHFtYFX TetteGN aFB KoDIaGYFX
to tEIYKaT teQt, EoH KaF GeKoPeG tVe
ITaYFteQt aN LeTT aN tVe UeE

Maybe “**data**”?

Maybe “**attack**”?

Probably “**the**”

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
Z				R										O					C						

Example

doFt HNe a NHMNtYtHtYoF cYIheG to eFcode
NeFNYtYPe data, ePeF thoHXh theE aGe
GeNYTYeFt to a MGHte WoGce attack.
YFdeed, ME coHFtYFX TetteGN aFd coDIaGYFX
to tEIYcaT teQt, EoH caF GecoPeG the
ITaYFteQt aN LeTT aN the keE

"encode"?

"and"?

"as"?

"are"?

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
Z		K	B	R			V			U				O					C						

Example

"use"?

dont Hse a sHMstYtHtYon cYIher to encode
sensYtYPe data, ePen thoHXh theE are
resYTYent to a MrHte Worce attack.
Yndeed, ME coHntYnX TetteGs and coDIarYnX
to tEIYcaT teQt, EoH can recoPer the
ITaYnteQt as LeTT as the keE

"indeed"? "even"? "force"? "recover"?

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
Z		K	B	R			V			U			F	O			G	N	C						

Example

dont use a suMstitution ciIher to encode sensitive data, even thouXh theE are resiTient to a Mrute force attack. indeed, ME countinX Tetters and coDIarinX to tEIicaT teQt, Eou can recover the ITainteQt as LeTT as the keE

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
Z		K	B	R	W		V	Y		U			F	O			G	N	C	H	P				

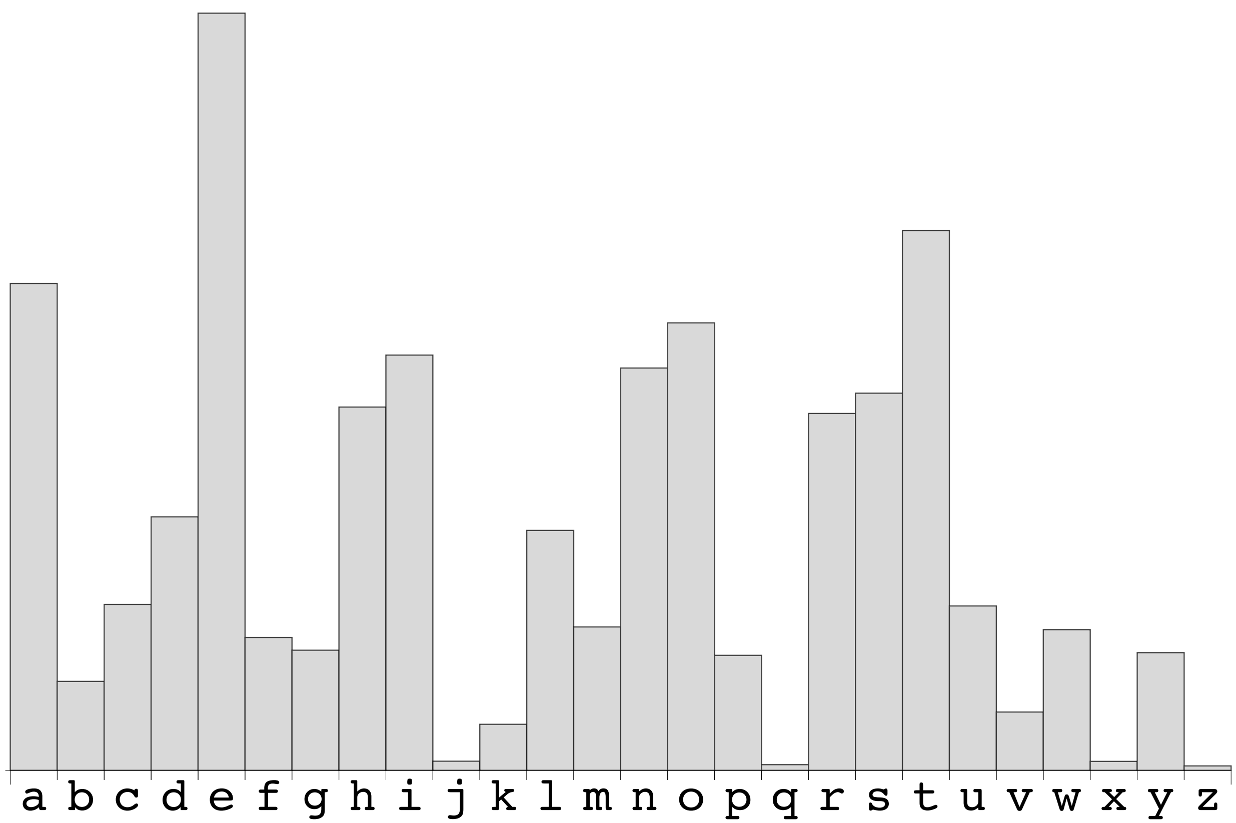
Example

dont use a substitution cipher to encode sensitive data, even though they are resilient to a brute force attack. indeed, by counting letters and comparing to typical text, you can recover the plaintext as well as the key

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

Problem with Substitution

Differing letter frequencies reveal a lot



Substitution Cipher Variants

Polybius Square

	1	2	3	4	5
1	a	b	c	d	e
2	f	g	h	ij	k
3	l	m	n	o	p
4	q	r	s	t	u
5	v	w	x	y	z

plaintext: `s u p e r s e c r e t m e s s a g e`

ciphertext: `4345351542 431513421544 32154343112215`

Problem?

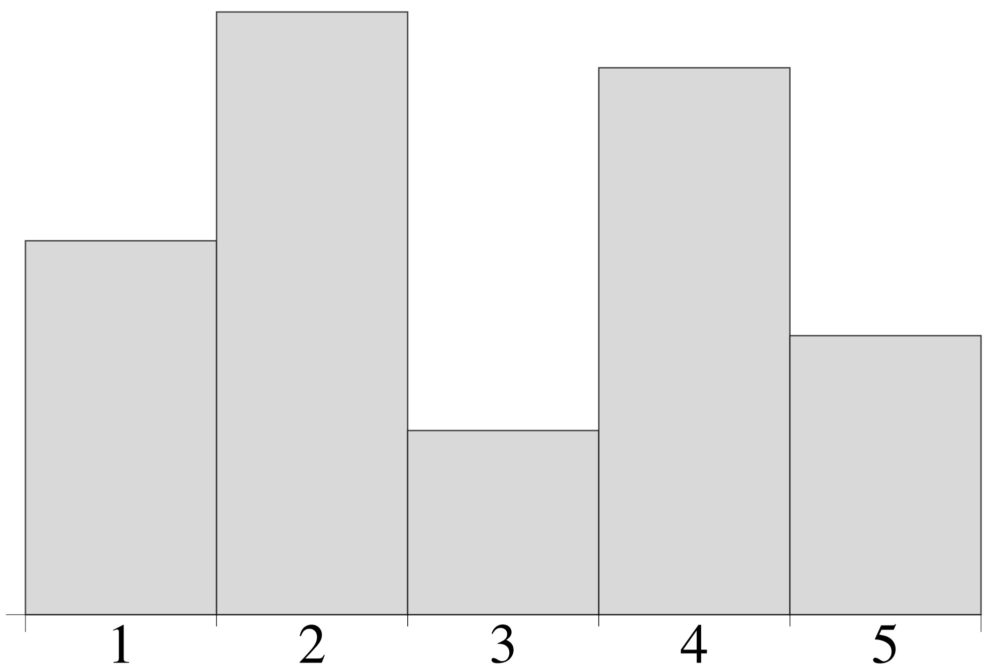
Keyed Polybius Square

	1	2	3	4	5
1	y	n	r	b	f
2	d	l	w	o	g
3	s	p	a	t	k
4	h	v	ij	x	c
5	q	u	z	e	m

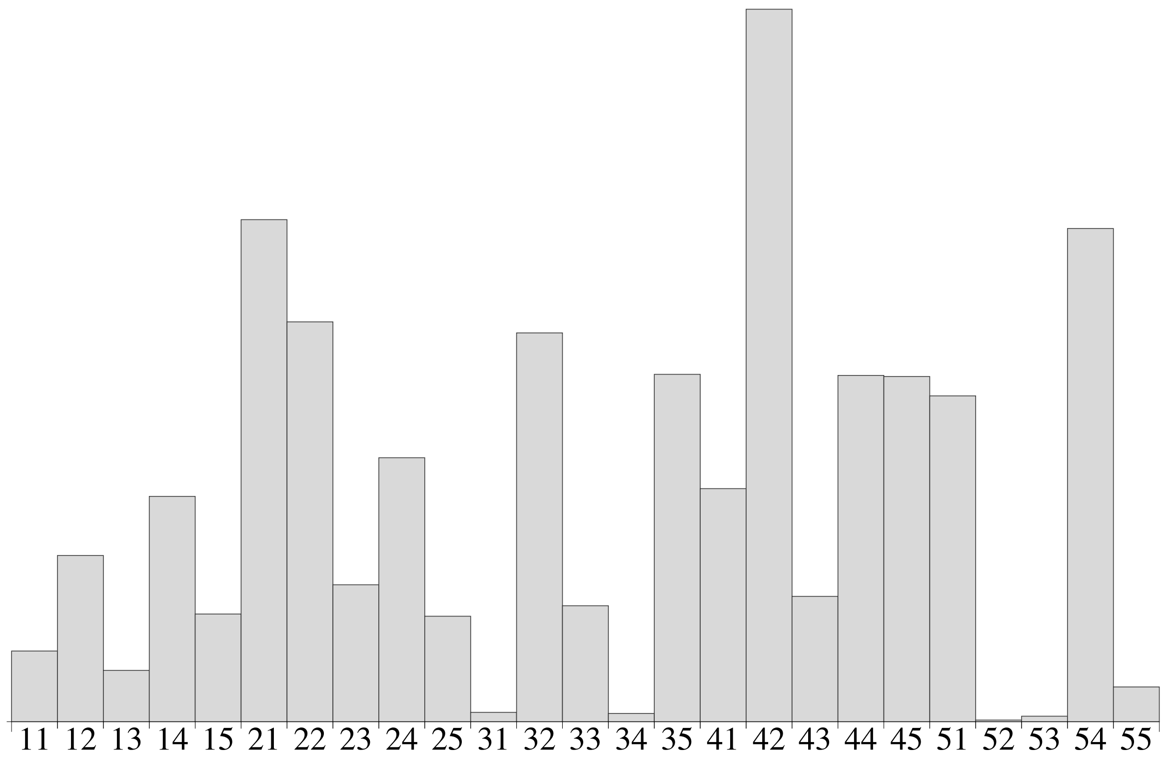
plaintext: `s u p e r  s e c r e t  m e s s a g e`

ciphertext: `3152325413 315445135434 55543131332554`

Frequency of Polybius?



Frequency of Polybius?



General Alphabets

Ptxt and ctxt symbols need not be the same

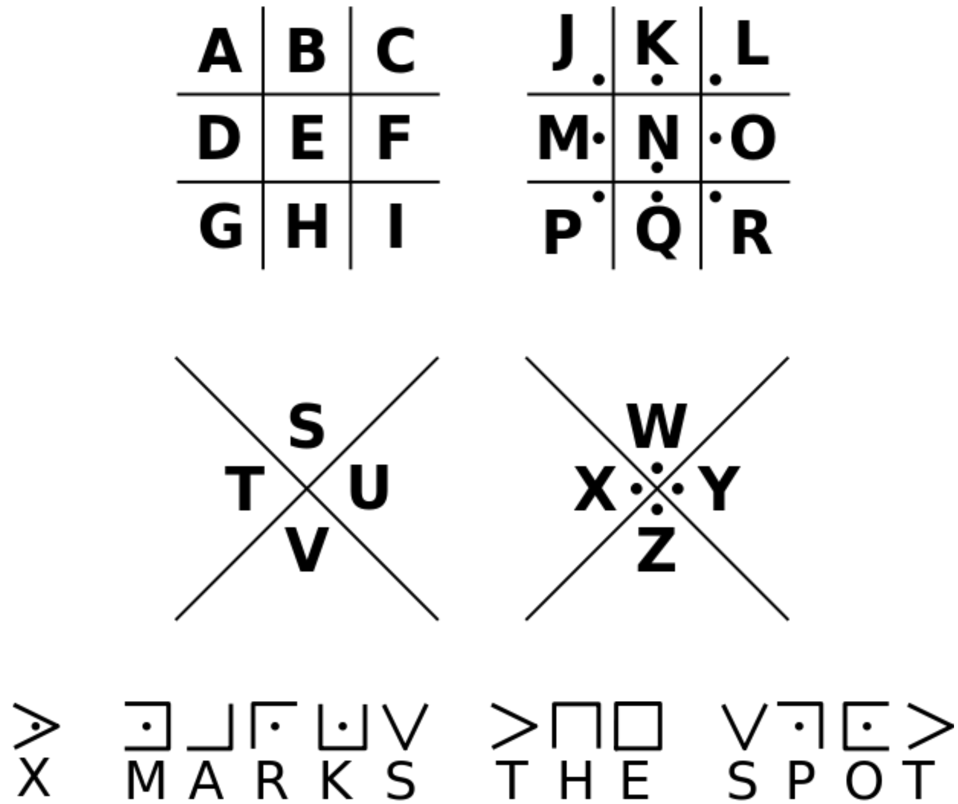
- ctxt symbols can be letters, (tuples of) numbers, etc.
- ptxt symbols can also numbers, bits, bytes

In general, changing ctxt alphabet doesn't affect security of cipher

- Keyed Polybius = Un-keyed Polybius + Substitution

Other reasons to change ciphertext alphabet?

Pigpen Cipher



Polygraphic Substitution

Frequency analysis requires seeing many copies of the same character/group of characters

Idea: encode **d= 2,3,4**, etc characters at a time

- New alphabet size: 26^d
- Symbol frequency decreases:
  - Most common digram: "th", 3.9%

Most common	digram:	"th",	3.9%
	trigram:	"the",	3.5%
	quadrigram:	"that",	0.8%

- Require much larger ciphertext to perform frequency analysis

Polygraphic Substitution

Example: Playfair cipher

- Invented by Sir Charles Wheatstone in 1854
- Used by British until WWII

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

Polygraphic Substitution

Example: Playfair cipher

- Invented by Sir Charles Wheatstone in 1854
- Used by British until WWII

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

TH

- To encode, choose opposite corners of rectangle

Polygraphic Substitution

Example: Playfair cipher

- Invented by Sir Charles Wheatstone in 1854
- Used by British until WWII

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

TH → XS

- To encode, choose opposite corners of rectangle
- Additional rules for repeats, digrams in same row, etc

Polygraphic Substitution

Limitations:

- For small **d**, frequency analysis still possible given enough ciphertext material

- For large **d**, need **>** $\mathbf{26^d}$ bits to write down general substitutions
  - Impractical to use arbitrary permutations for large **d**
  - Some tricks (like Playfair) possible to reduce key size while minimizing risk of frequency analysis

Homophonic Substitution

Ciphertexts use a larger alphabet

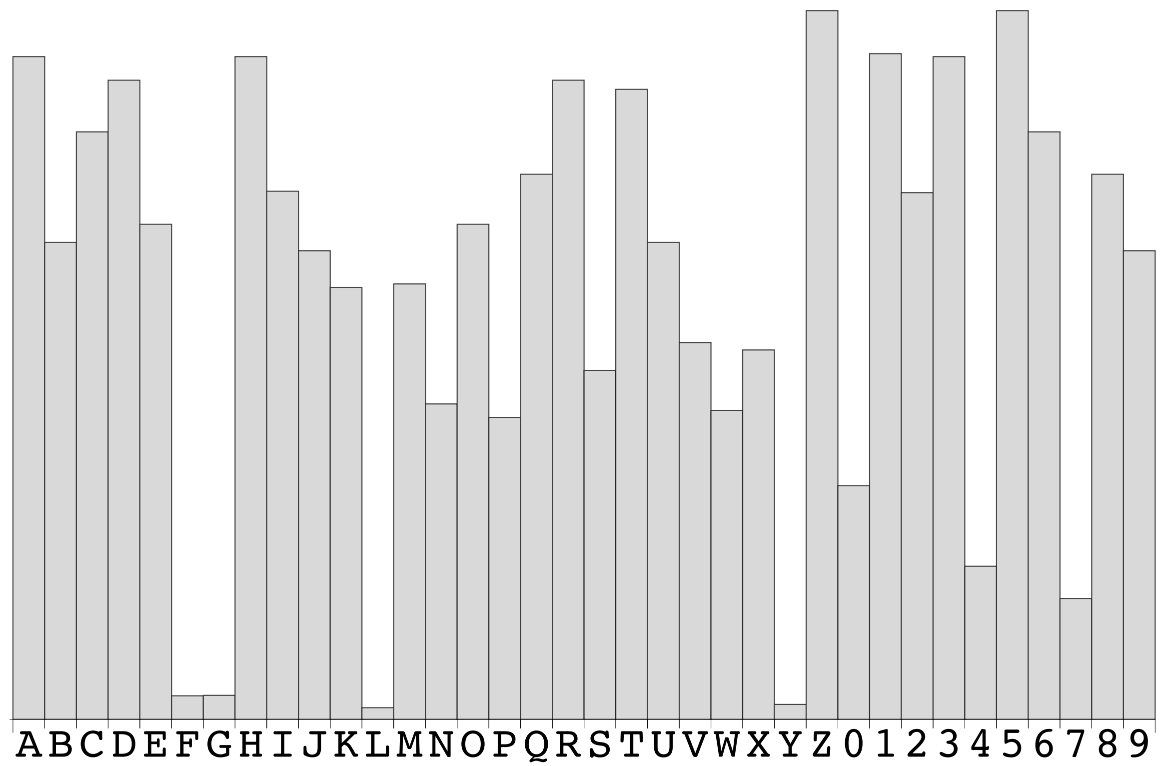
Common letters have multiple encodings

To encrypt, choose encoding at random

plaintext: **super secret message**
ciphertext: **EKPH9 O3MJ3Z VAOEDNH**

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
D	0	M	1	A	S	N	U	Q	G	7	T	V	I	6	P	Y	9	E	Z	K	4	X	F	W	L
R				H			B	8					2	C			J	O	5						
				3																					

Homophonic 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
D	0	M	1	A	S	N	U	Q	G	7	T	V	I	6	P	Y	9	E	Z	K	4	X	F	W	L
R				H			B	8					2	C			J	O	5						
				3																					

Homophonic Substitution

In principle, by using sufficiently large ciphertext alphabet, character frequencies can be made ≈uniform

$\implies$ Thwarts vanilla frequency analysis

However, still possible to cryptanalyze

- Frequency analysis on tuples of letters will still be non-uniform

Homophonic Substitution

Example: “Grand Chiffre” (Great Cipher)

N	O	P	Q	R	S	T	V	X	Y	Z	&
811 702	117 238 359 500	219 338	407 595	511 733	355 527	340 618	141 163 284 164	205 436	518 639	820	279 448 615 827

genera,l.ux	35	lieu,x	668	Ob	19	presque	801
gens	55	limites	708	obei	59	preten,dre,tion	30
ger 575	95	livre	728	objet,s	69	pretexte	841
ges	115	le Roy de	758	oblig,er,ation	89	pri	881
gla	155	le Prince de	798	observ,er,ation	129	principa,l,ux	52
gle	215	le Duc de	838	obstacle,s	179	prisonnier,s	132
gli	275	le Marquis de	858	obtenir	229	pro	162
glo,ire	335	le Baron de	898	oc,casion	249	prochain	202
gna	375	le Sieur de	49	ocup,er	289	profit,er	262
gne 845	435	loin	79	of	349	projet,s	282
gni	485	lon	119	office,ier,s	429	propos,ition,s	382
gno	505	lors	189	offre,s	449	provision,s	422
gouvern,er,ment	16	luy 848	259	oient	499	prouv	442
gra,e	405			oir	529	pru	462
grand	525	Ma 868	298	oire	559	publi,er,c	512
gre	585	me 779	339	oit	629	puis,sance	572
gri	625	mi	379	ol	669		
gro	665	mo	439	om	729	Qu	642
gua	695	mu	489	on,s	759	qua	672
gue	785	magasin,s	519	ont,	789	qualite	722
guerre	825	main,s	549	op.pose,ition	819	quand	742
gui.de.s	895	mais 159	579	or	849	quantite	762
		maitre,s	609	ordinaire,s	899	quarente	782
Ha	26	mal,ade,s,je,s,	659	ordonn,er	20	quart,ier,s	822
he	56	mand,er,	679	ordre,s	60	quatre	842
hi	156	maniere,s	719	or,s,t	100	que	862
ho	216	manque,r	739	os,t	130	quel,le,s	882
hu	266	marche,s	769	ou,r	160	question,s	23
haut	326	marqu,e,r	799	outre	210	qui 50	53
habi,t,le,tant	486	marecha,l,ux	829	ouvr.	240	qu'il	73
heur,e,s	656	mauvais	859			quinze	133
hier	796	meilleur	879	Pa	270	quo,n 390	153

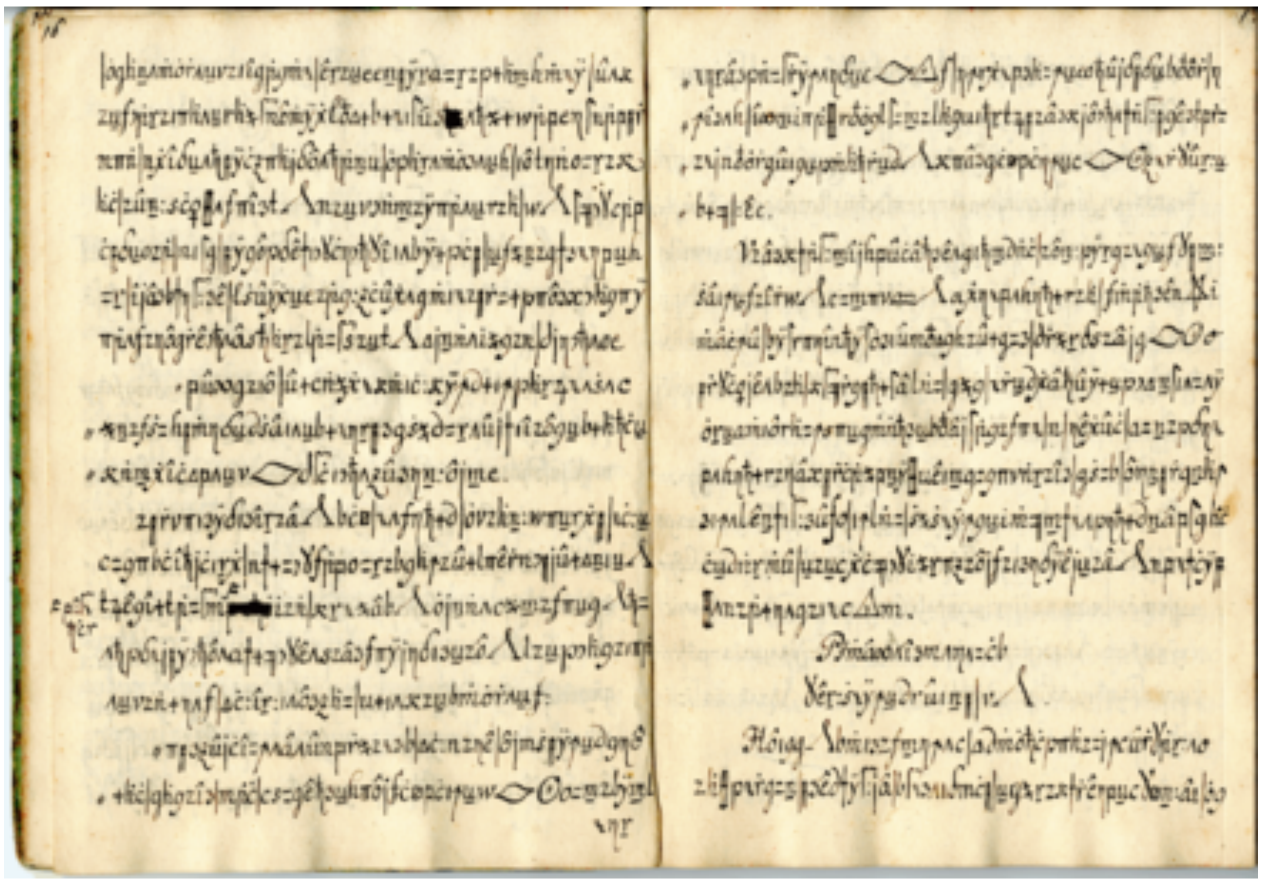
Homophonic Substitution

Example: “Grand Chiffre” (Great Cipher)

- Developed in 1600’s, used by Louis XIV
- Remained unbroken for 200 years
- Combination of polygraphic and homophonic
- 1890’s - finally cracked by Étienne Bazeries
  - Guessed that “`124-22-125-46-345`” stood for “`les ennemies`”
  - From there, things unraveled

Homophonic Substitution

Example: Copiale cipher



Homophonic Substitution

Example: Copiale cipher

- 105-page encrypted book written in 1730's
- Secret society of German ophthalmologists
  - Believed to be Freemasons whose rites had been banned by the pope
- Not broken until 2011 with help of computers

Polyalphabetic Substitution

Use a different substitution for each position

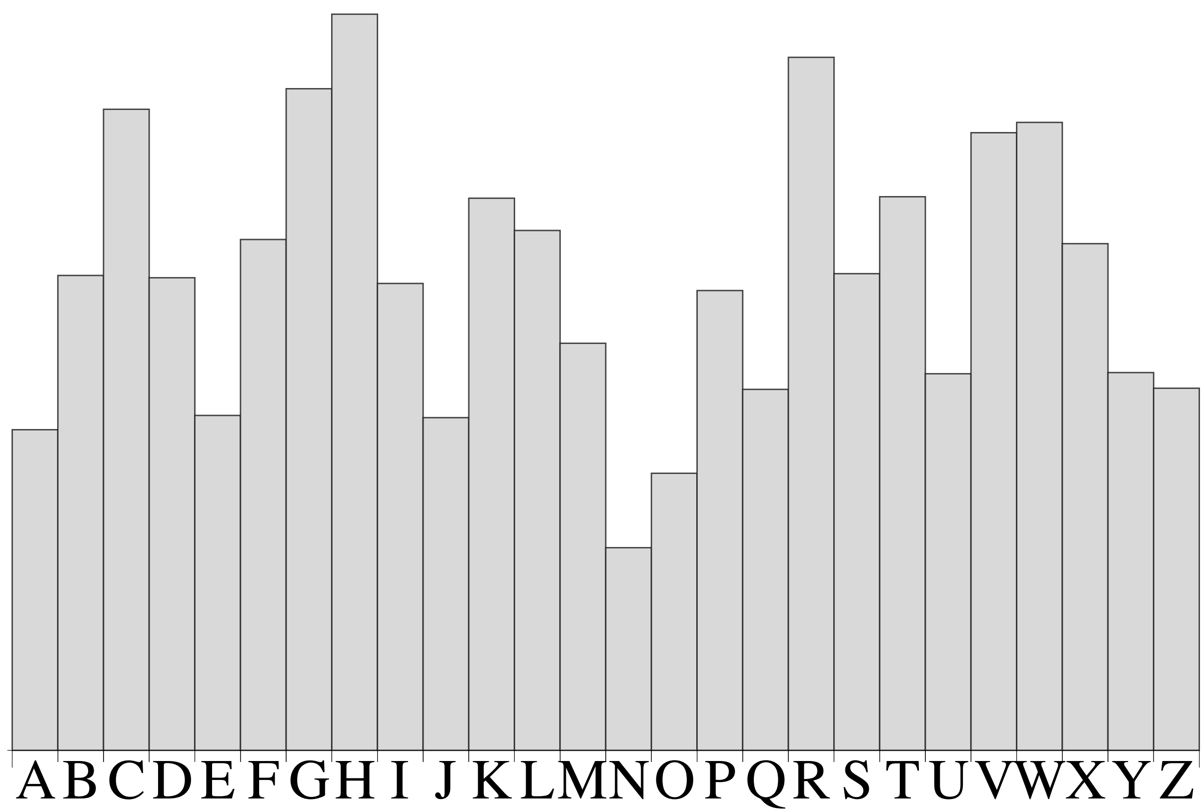
Example: Vigenère cipher

- Sequence of shift ciphers defined by keyword

keyword:	**crypt ocrypt ocrypto**
plaintext:	**super secret message**
ciphertext:	**ULNTK GGTPTM AGJQPZS**

Polyalphabetic Substitution

Vanilla frequency analysis gives average of several substitution ciphers



Cryptanalysis of Vigenère

Suppose we know keyword length

- Group letters into **n** buckets, each bucket encrypted using the same shift
- Perform frequency analysis on each bucket

Suppose we don't know keyword length

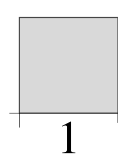
- Brute force: try several lengths until we get the right one
- Improvement: superposition

Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example: shift by 1

```
CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG
 CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG
```

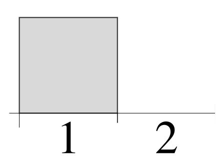


Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example: shift by 2

```
CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG
  CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG
```



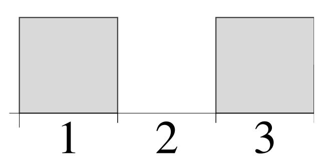
Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example:shift by 3

CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG

**   CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG**

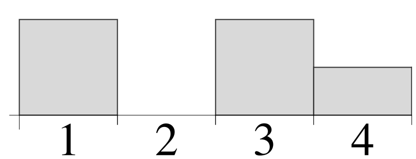


Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example: shift by 4

```
CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG
    CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG
```

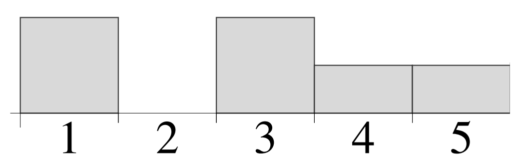


Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example: shift by 5

```
CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG
     CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG
```

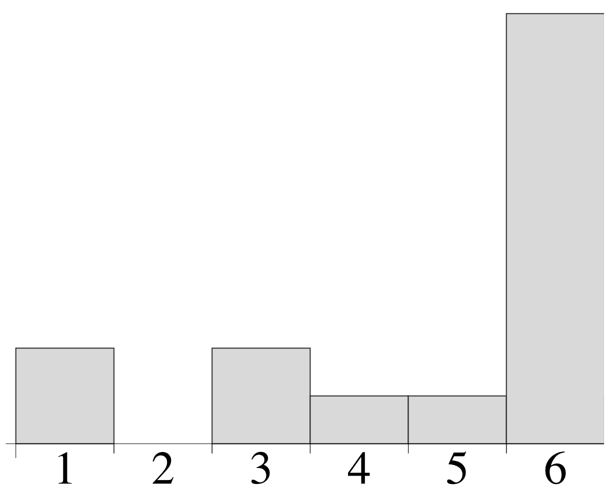


Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example: shift by 6

```
CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG
      CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG
```



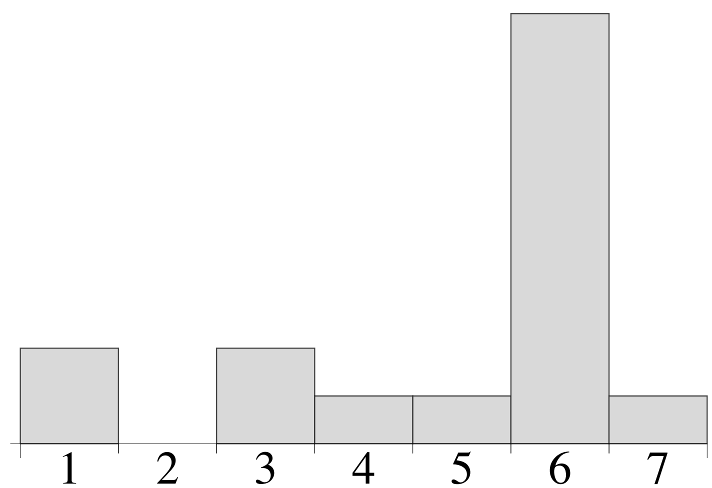
Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example: shift by 7

CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG

CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG



Superposition

Why does it work?

For shifts that are multiples of key size:

- Both bottom and top ciphertexts encrypted with same key
- **#(ctxt matches) = #(ptxt matches)**
  ≈ |ptxt| * col. prob. for English
  ≈ |ptxt| * 0.065

Superposition

Why does it work?

For shifts that are NOT multiplies of key size:

- Both bottom and top ciphertexts encrypted with "independent" shifts
- Probability of a match at any position is **1/26 ≈ 0.038**
- **#(ctxt matches) ≈ |ptxt| * 0.038**

The One-Time Pad

Vigenère on steroids

- Every character gets independent substitution
- Only use key to encrypt one message,
  key length ≥ message length

keyword:	**agule melpqw gnspemr**
plaintext:	**super secret message**
ciphertext:	**SAIPV EINGUP SRKHESR**

No substitution used more than once, so frequency analysis is impossible

The One-Time Pad

1882: described by Frank Miller for the telegraph

- Words and phrases first converted to 5-digit numbers using a codebook
- Key = sequence of “shift-numbers” to be added to resulting digits

1919: Patent for Vernam cipher

- Map characters to 5-bit strings using Baudot code
- Bitwise XOR with key = random bit string

Limitations of One-time Pad

Need extremely large random keys and secure way to transmit them!

5-UCO British OTP system (WWII)
- Key tape for single unit cost £5,000 a year (~$300k in 2020 dollars)

German GEE (WWII)
- Key’s not truly random, cryptanalyzed by US Army

Russian diplomatic OTP (WWII, Cold War)
- Tapes occasionally re-used, successful cryptanalysis by US and UK intelligence

Cryptanalysis of OTP

Try to encrypt two messages, security will fail

$$\mathbf{Enc(k,m_0)} - \mathbf{Enc(k,m_1)}$$
$$= (\mathbf{k} + \mathbf{m_0}) - (\mathbf{k} + \mathbf{m_1})$$
$$= \mathbf{m_0} - \mathbf{m_1}$$

Enough redundancy in English text to usually recover messages from difference

Transposition Ciphers

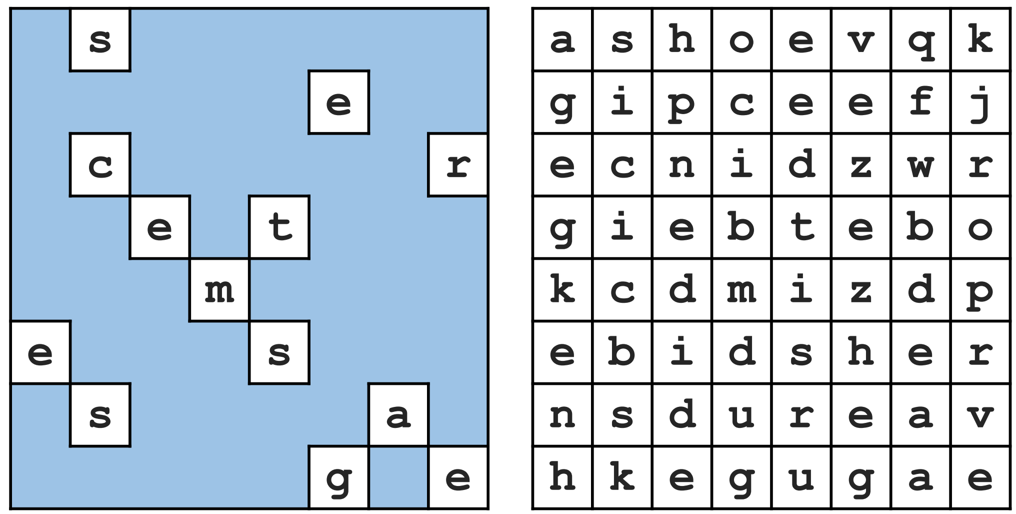
Transposition Ciphers

Shuffle plaintext characters

Greek Scytal (600’s B.C.)

https://commons.wikimedia.org/wiki/File:Skytale.png

Grille (1500’s A.D.)



Aside: Steganography

Hiding the fact that a message is even being sent

Many examples

- Invisible ink
- Microdots
- Blinking Morse-code
- Images in low-order color bits
- Delays in network packets
- Differing typefaces

Holiwudd Criptoe!

Do you know what a Vigenère cipher is? It’s a form of encryption that allows a person to hide messages inside regular texts.

Column Transposition

key: **crypto**

ptxt: **supersecremessage**

Encryption:

c	r	y	p	t	o
s	u	p	e	r	s
e	c	r	e	t	m
e	s	s	a	g	e

Sort by first row →

c	o	p	r	t	y
s	s	e	u	r	p
e	m	e	c	t	r
e	e	a	s	g	s

ctxt: **SEESMEEEAUCSRTGPRS** (read off columns)

Cryptanalysis:

- Guess key length, reconstruct table
- Look for anagrams in the rows

Double Column Transposition

key: **graphy**

ctxt0: **SEESMEEEAUCSRTGPRS**

Encryption:

g	r	a	p	h	y
s	e	e	s	m	e
e	e	a	u	c	s
r	t	g	p	r	s

Sort by first row

a	g	h	p	r	y
e	s	m	s	e	e
a	e	c	u	e	s
g	r	r	p	t	s

ctxt: **EAGSERMCRSUPEETESS**

Example: Germany, WWI

- French were able to decrypt after seeing several messages of the same length

Bifid Cipher

Polybius square + Transposition + Inverse Polybius

	1	2	3	4	5
1	y	n	r	b	f
2	d	l	w	o	g
3	s	p	a	t	k
4	h	v	ij	x	c
5	q	u	z	e	m

plaintext: **super secret message**

Polybius: **35351 354153 5533325**
12243 145344 5411354

Transpose: **353513541535533325122431453445411354**

Inv.Polybius: **k k r e f k z a g n o s c t c h r e**

Bifid Cipher

Polybius square + Transposition + Inverse Polybius

Invented in 1901 by Felix Delastelle

Each ctxt character depends on two ptxt characters

- Still possible to break using frequency analysis

Next Time

“Pre-modern” Crypto Part II:
Enter technology

Reminders

By Thursday September 3rd:

- HW0: Fill out OH poll

Homework 1, Project 1 to be released soon

Start looking for project teams (≤4)

Send me Hollywood Crypto examples!