COS433/Math 473: Cryptography

Mark Zhandry

Princeton University

Spring 2020

Reminders

Course website: cs.princeton.edu/~mzhandry/2020-Spring-COS433

- Enroll in Piazza
- Fill out OH poll
- HW1, PR1 to be released next week

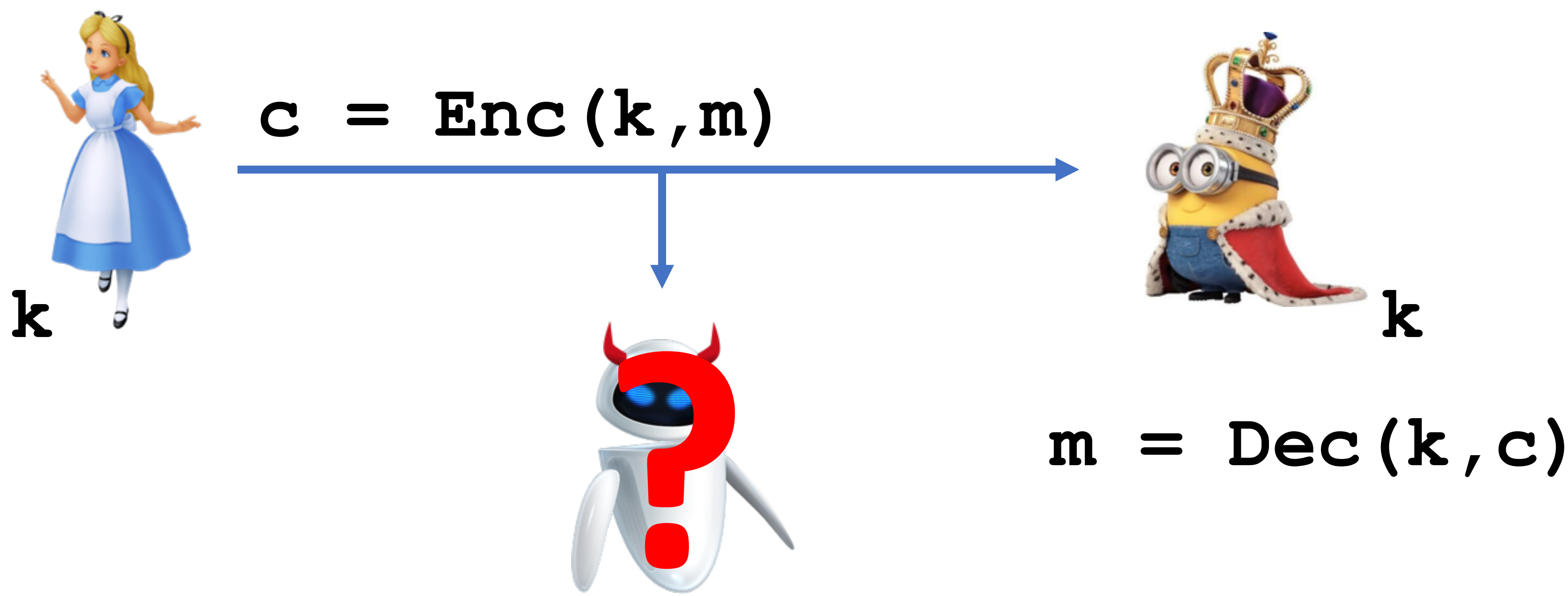
Find teams for projects (up to 4)

Previously on COS 433…

Pre-modern Cryptography

1900 B.C. – mid 1900’s A.D

With few exceptions, synonymous with **encryption**



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`**

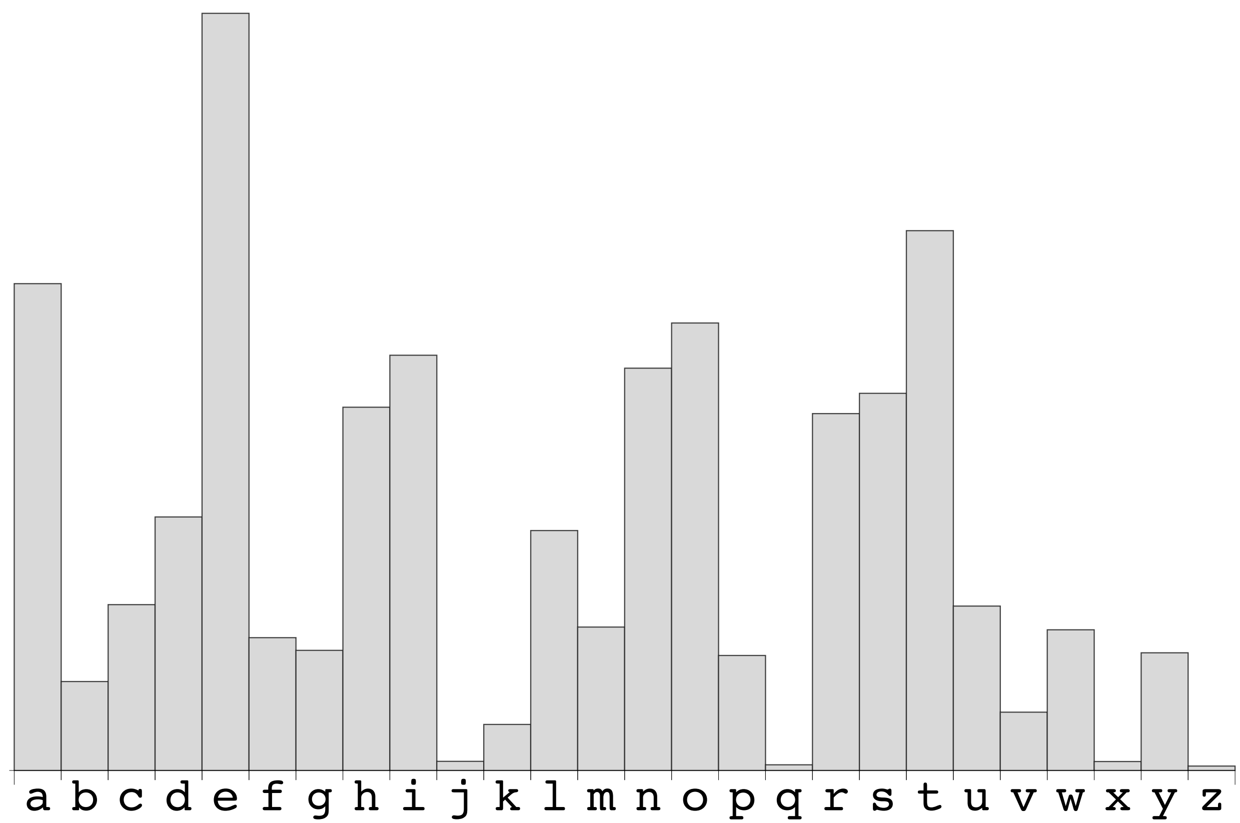
ciphertext: **`ARQYV AYSVYX EYAAFJY`**

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



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`

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: $\mathbf{26^d}$
- Symbol frequency decreases:
    - Most common

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

- Require much larger ciphertext to perform 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																					

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
```

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

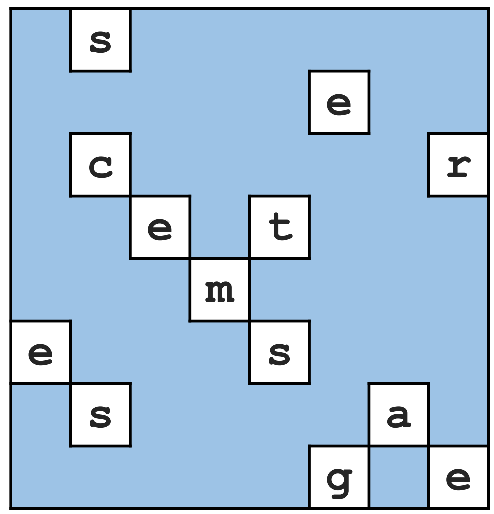
Transposition Ciphers

Shuffle plaintext characters

Greek Scytal (600's B.C.)

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

Grille (1500's A.D.)



a	s	h	o	e	v	q	k
g	i	p	c	e	e	f	j
e	c	n	i	d	z	w	r
g	i	e	b	t	e	b	o
k	c	d	m	i	z	d	p
e	b	i	d	s	h	e	r
n	s	d	u	r	e	a	v
h	k	e	g	u	g	a	e

<u>Today</u>

“Pre-modern” Crypto Part II:

Enter Technology

Disk-based Substitution Ciphers

First Invented by Alberti, 1467

*

†

‡

* cropped from http://www.cryptomuseum.com/crypto/usa/ccd/img/301058/000/full.jpg
† cropped from https://www.flickr.com/photos/austinmills/13430514/sizes/l
‡ https://commons.wikimedia.org/wiki/File:Captain-midnight-decoder.jpg

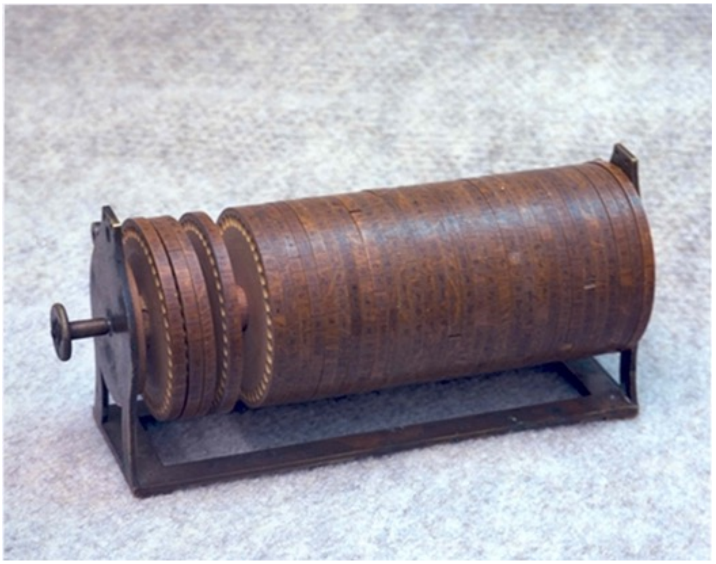
Disk-based Substitution Ciphers

In most basic form, simple monoalphabetic cipher

Alberti Cipher – rotate the disk periodically

- Considered the first polyalphebetic cipher

Jefferson disk: used by US military until WWII



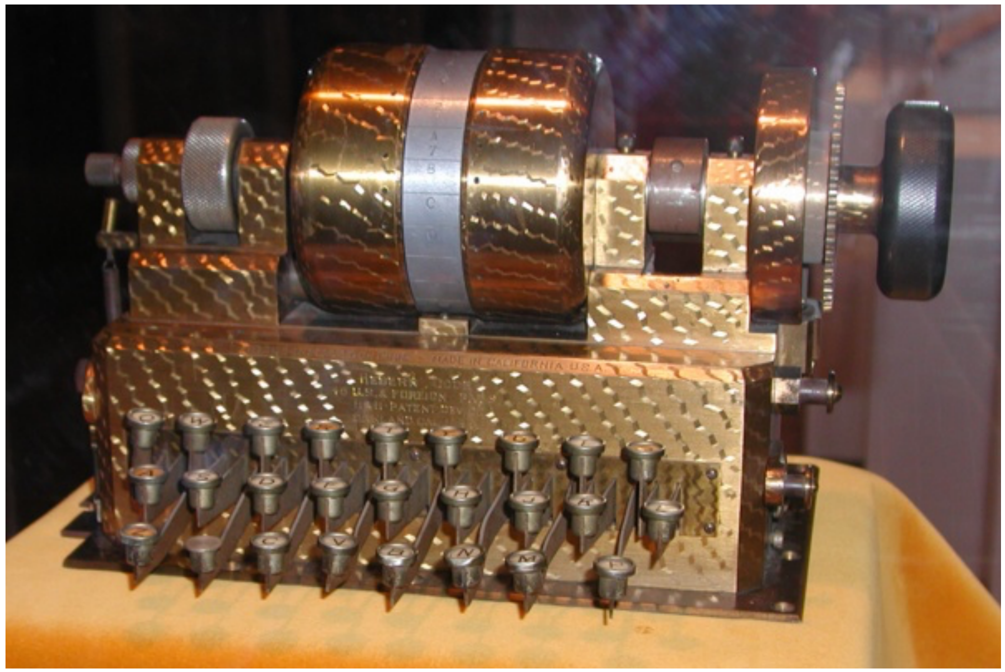
Rotor Machines

Widespread starting in the 1920’s

Automatically advance rotor in regular intervals

- Automate process of rotating disk to change substitution
- Eventually allow for more complex substitutions

Rotor Machines



https://commons.wikimedia.org/wiki/File:Hebern1.jpg

Rotor contains substitution, advances by one after each stroke, creating different substitution

Rotor Machines

More rotors!

http://americanhistory.si.edu/collections/search/object/nmah_694514

Every time one rotor completes a revolution, it advances the next rotor

Cryptanalysis of Rotor Machines?

d rotors → polyaphabetic cipher with key length $\mathbf{26^d}$

Possible to break via brute force if only a few rotors

But what if you don't know the permutation given by the rotors?

Edward Hebern vs. William Friedman

Hebern invented machines using 1 to 5 rotors

Tried to sell to US Military, but rejected

Unknown to Hebern, US cryptanalyst Friedman had shown to break machine, given just **10 ciphertexts**

- And, Friedman wasn’t even given rotor wirings!

PURPLE

Diplomatic cipher used by Japanese Foreign Office

Using knowledge gained from cryptanalyzing Hebern's machine, US Intelligence was able to complete reconstruct the cipher machine **using only intercepted ciphertexts**

Friedman's technique applies to essentially any cipher-based machine where fast rotor at one end

Determining Rotor Wirings

Each rotor represents a permutation $R_1, R_2, \ldots$ on $\mathbb{Z}_{26}$

If rotor i has rotated j times, then it applies the permutation

$$C^j \circ R_i \circ C^{-j}$$

Where C maps "a" to "b", "b" to "c", etc

Overall permutation:

$$C^l \circ R_3 \circ C^{-l} \circ C^k \circ R_2 \circ C^{-k} \circ C^j \circ R_1 \circ C^{-j}$$

Determining Rotor Wirings

For first 26 letters, only first rotor ever turns

Can write permutation as

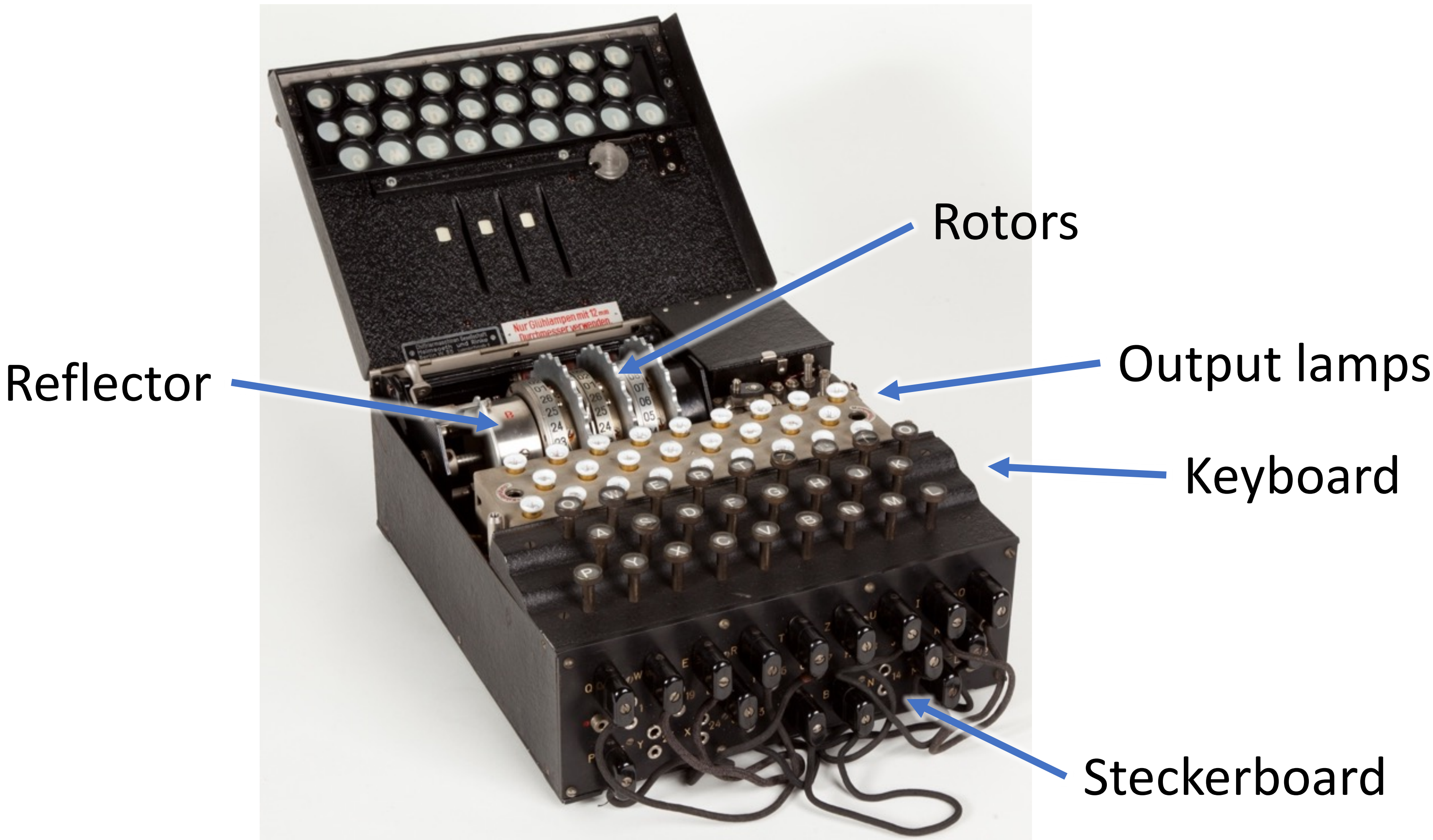
$$\mathbf{L} \circ \mathbf{C}^{j} \circ \mathbf{R}_1 \circ \mathbf{C}^{-j}$$

For next 26 letters, identical, except different **L**:

$$\mathbf{L}' \circ \mathbf{C}^{j} \circ \mathbf{R}_1 \circ \mathbf{C}^{-j}$$

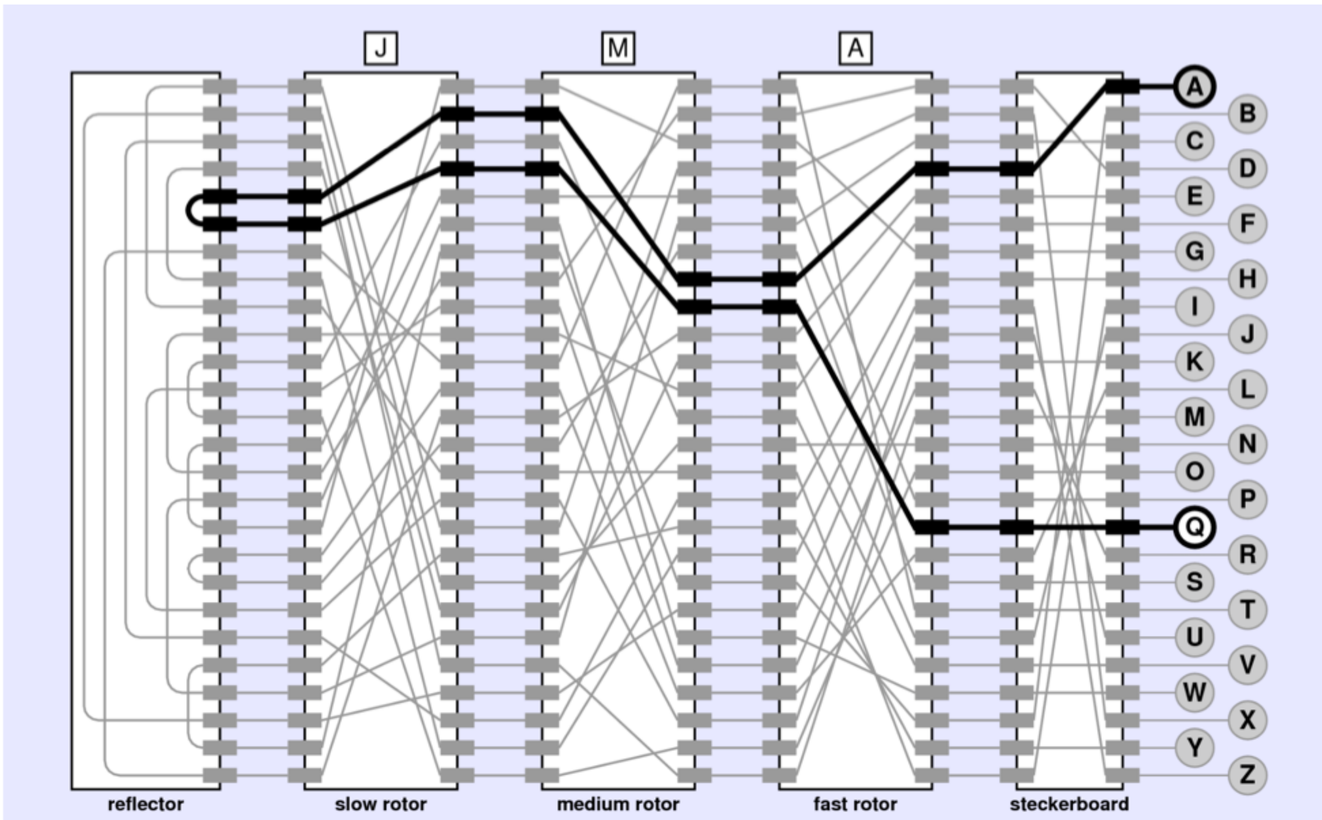
A lot of structure in cipher to exploit

The German Enigma Machine



https://commons.wikimedia.org/wiki/File:Enigma_(crittografia)_-_Museo_scienza_e_tecnologia_Milano.jpg

Enigma Diagram



http://stanford.edu/class/archive/cs/cs106a/cs106a.1164/handouts/29A-CryptographyChapter.pdf

Enigma Keys

Key:

- Selection of 3 rotors out of 5 (60 possibilities)
- Initial rotor setting (26^3)
- Steckerboard wiring (216,751,064,975,576)

Possible attack strategies?

- Brute force
  - 2^{68} possible keys: feasible today, but not in WWII
- Frequency analysis
  - Polyalphabetic with key length $26^3 = 17576$
  - Likely no key was used to encrypt enough material

Cracking the Enigma

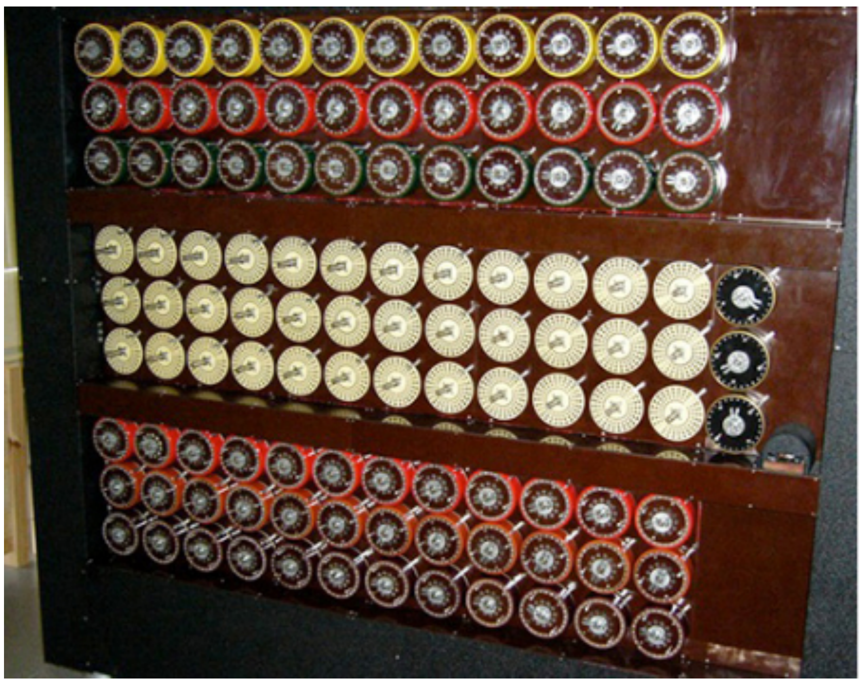
Key Factors:

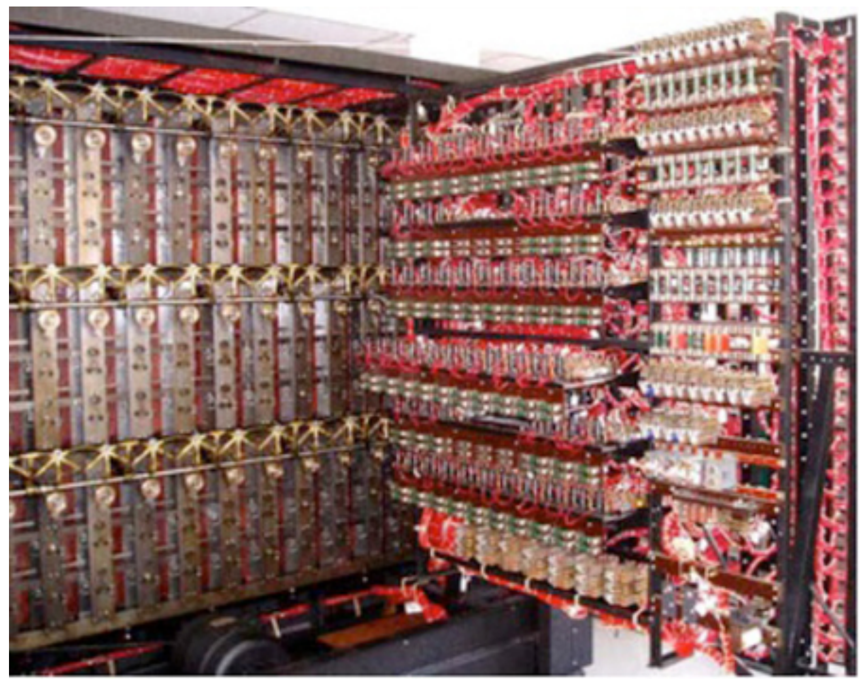
- Captured Enigma device

Cracking the Enigma

Key Factors:

- Technology

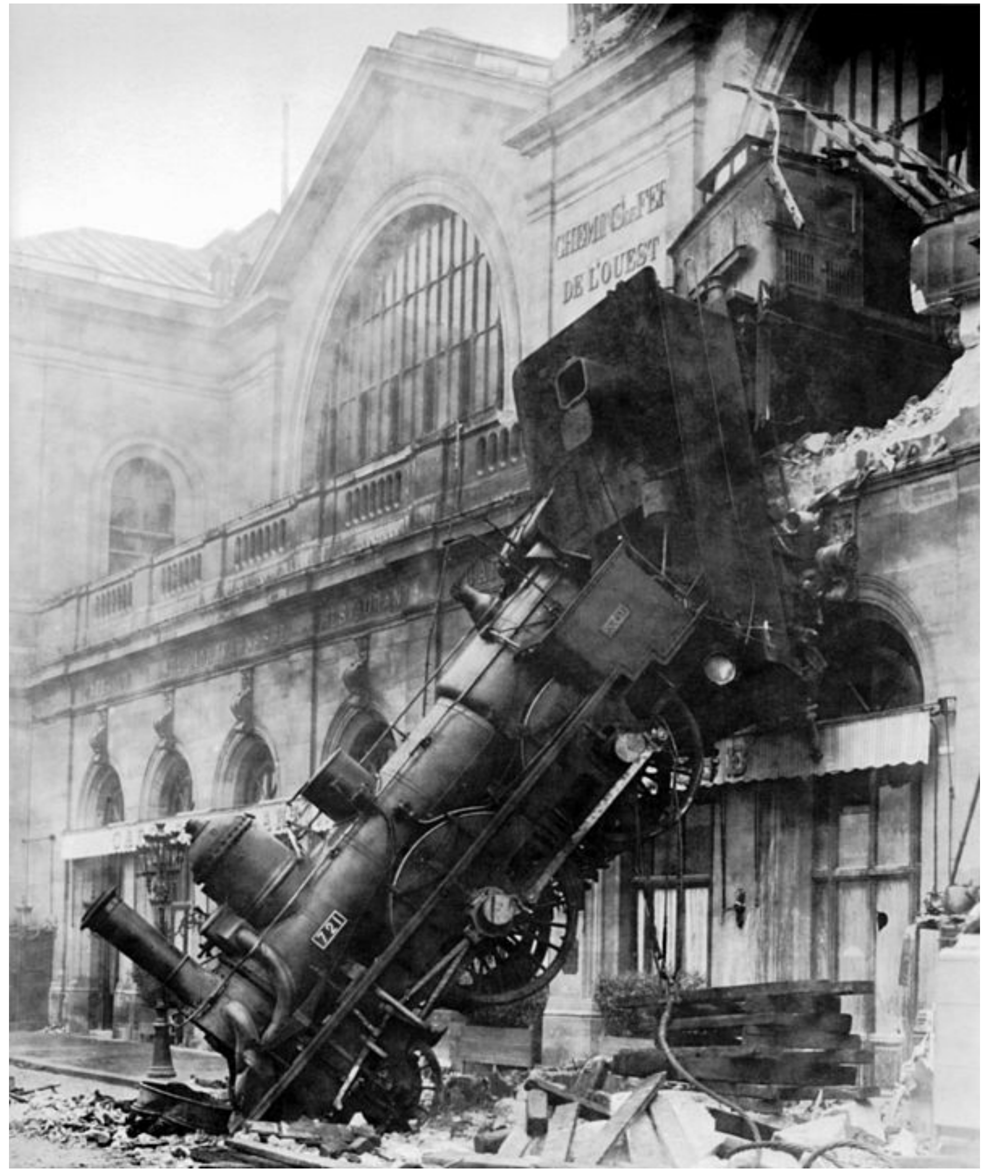




Cracking the Enigma

Key Factors:

- User error/bad practices



Cracking the Enigma

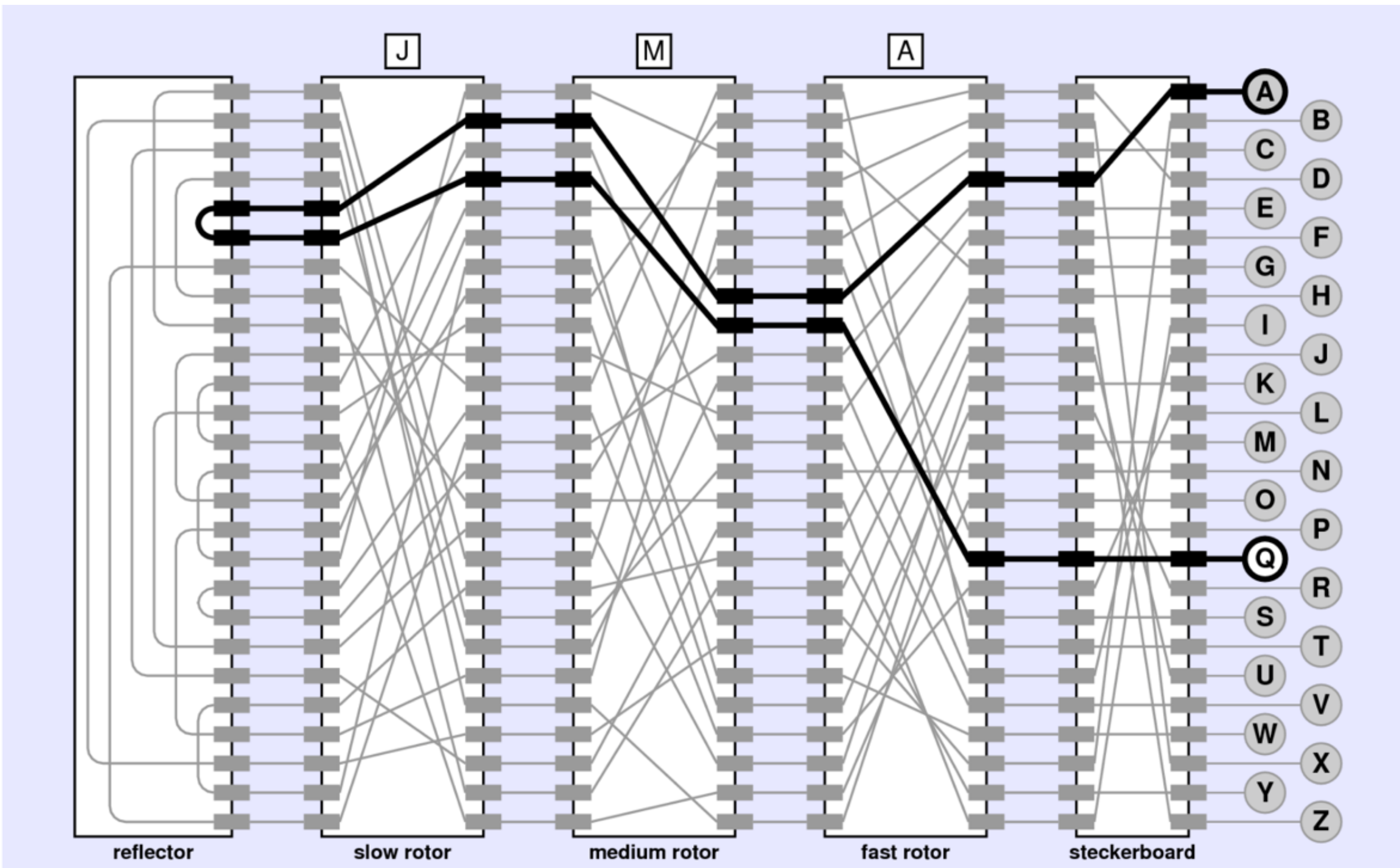
Key Factors:

- Known/chosen plaintexts

Cracking the Enigma

Key Factors:

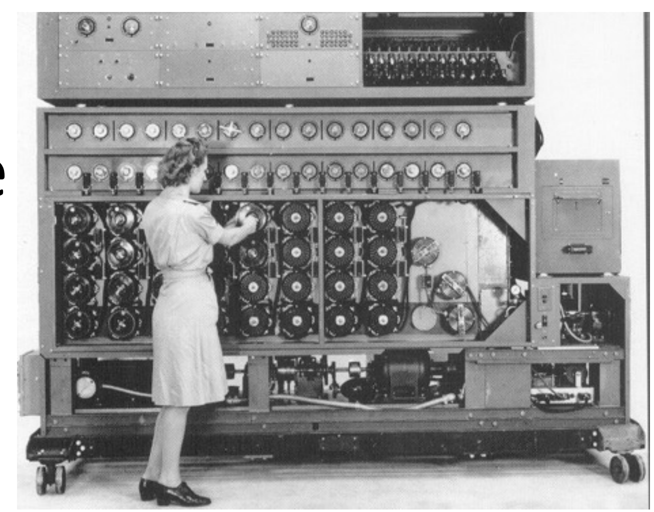
- Mathematical weaknesses



A Key Insight: Loops

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

- Loops unaffected by steckerboard wiring
- Only need to search the $\approx 2^{20}$ rotor positions to find one that generates such a loop
- Possible at the time using the Bombe



http://www.cryptomuseum.com/crypto/bombe/img/us_bombe_full.jpg

Takeaway: Crypto is Hard

Designing crypto is hard, even experts get it wrong

- Just because I don't know how to break it doesn't mean someone else can't

Unexpected attack vectors

- Known/chosen plaintext attack
- Chosen *ciphertext* attack
- Timing attack
- Power analysis
- Acoustic cryptanalysis

Takeaway: Crypto is Hard

Don't design your own crypto
- You'll probably get it wrong
- Use peer-reviewed schemes instead

Actually, don't even implement your own crypto
- Instead, use well studied crypto library built and tested by many experts

Takeaway: Need for Formalism

For most of history, cipher design and usage based largely on intuition

- Intuition in many cases false

Instead, need to formally define the usage scenario

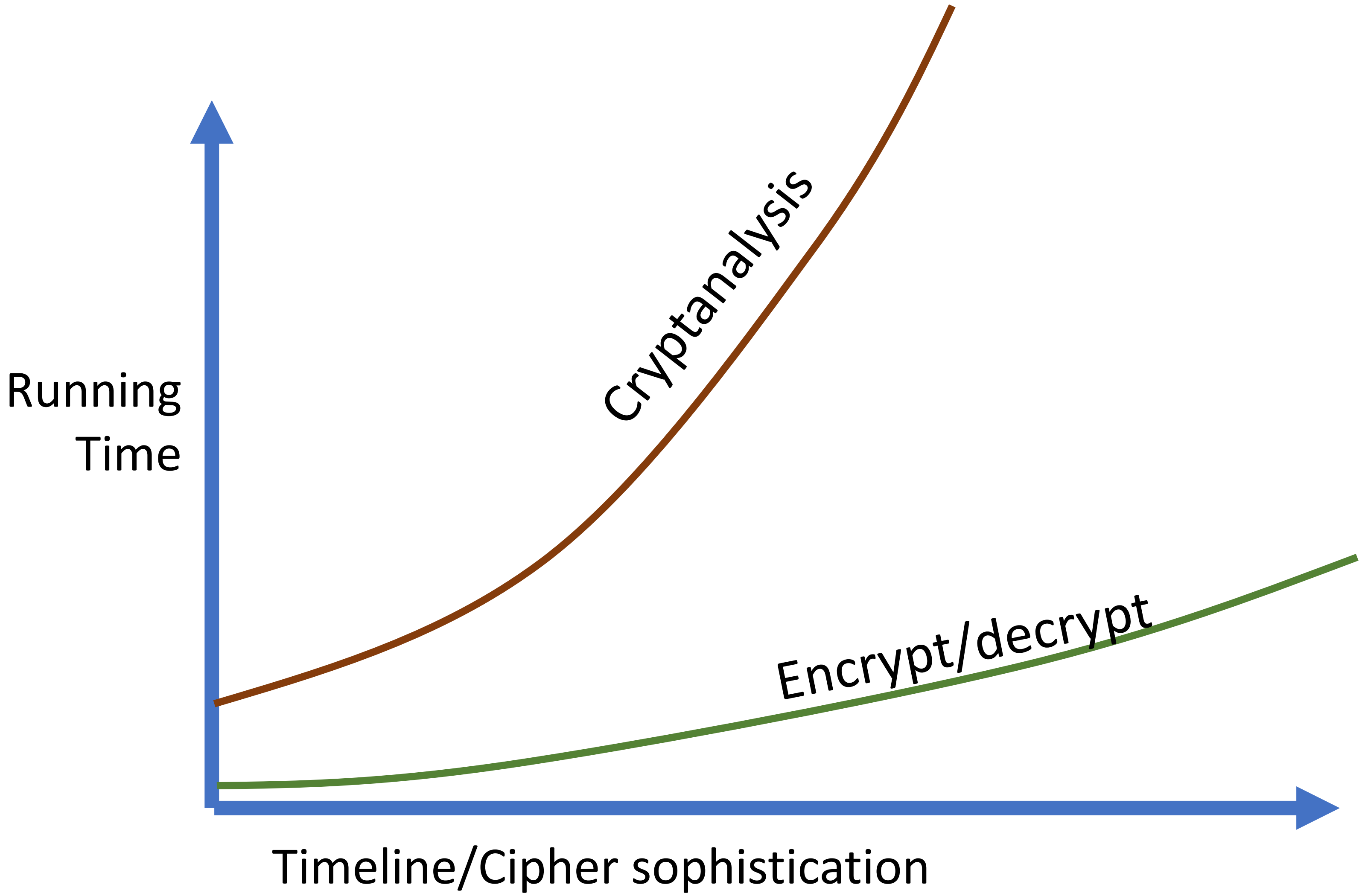
- Prove that scheme is secure in scenario
- Only use scheme in that scenario

Takeaway: Kerckhoffs's Principle

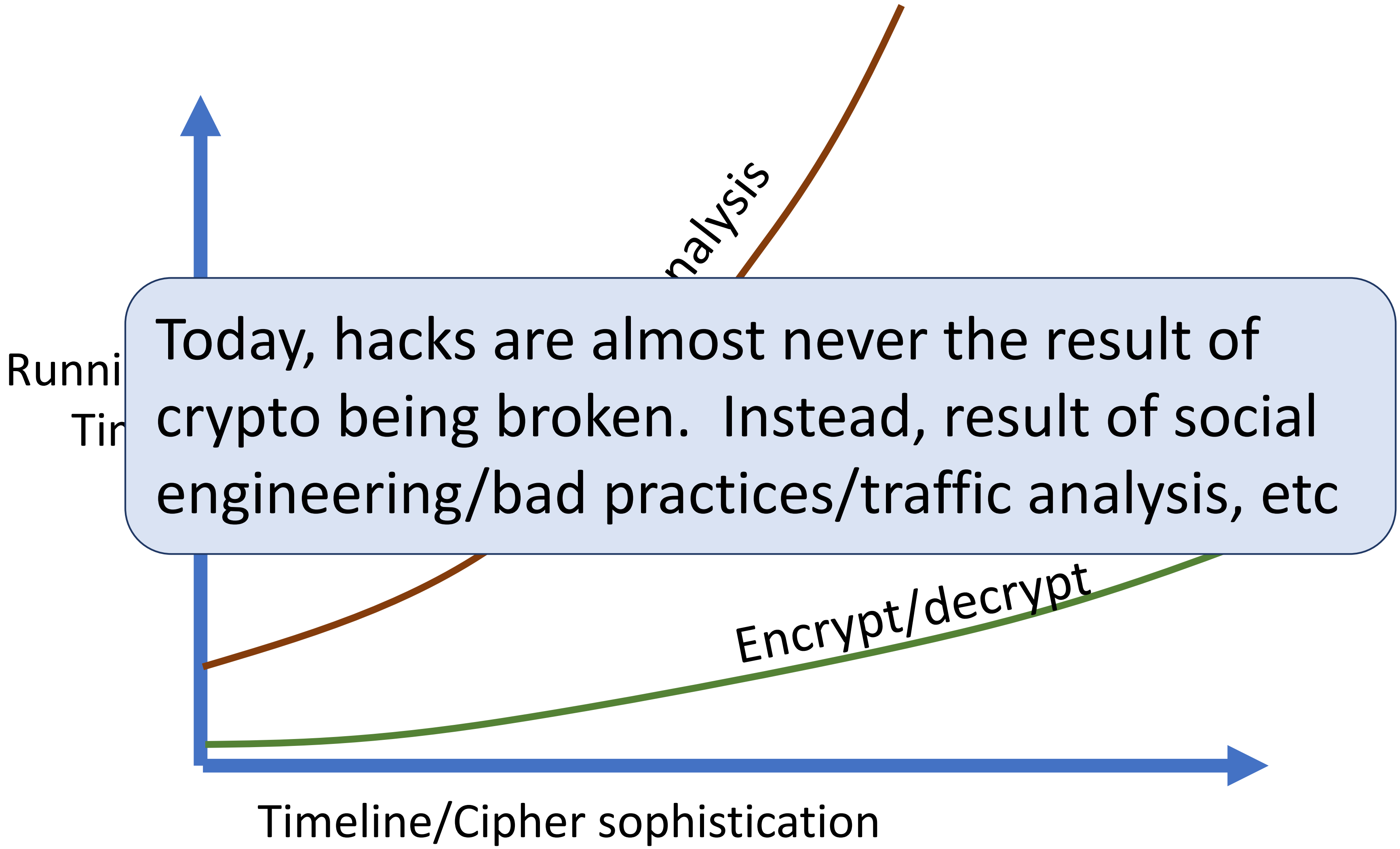
Kerckhoffs's Principle: A cryptosystem should be secure even if everything about the system, except the key, is public knowledge.

- Leaks happen. Should only have to update key, not redesign entire system
  - Even worse, cipher can potentially be reconstructed from ciphertexts
- More eyes means more likely to be secure
- Necessary for formalizing crypto

Takeaway: Importance of Computers



Takeaway: Importance of Computers



Today, hacks are almost never the result of crypto being broken. Instead, result of social engineering/bad practices/traffic analysis, etc

Modern Cryptography

Basics of Defining Crypto

Usually three pieces:

1. **Syntax:** what algorithms are there, what are the inputs/outputs
2. **Correctness/completeness:** how do the algorithms interact
3. **Security:** what should an adversary be permitted/prevented from doing

Formalizing Encryption

Syntax:

- Key space $\mathbf{K}$
- Message space $\mathbf{M}$
- Ciphertext space $\mathbf{C}$
- $\mathbf{Enc: K \times M \rightarrow C}$
- $\mathbf{Dec: K \times C \rightarrow M}$

Correctness:

- For all $\mathbf{k \in K,\ m \in M,\ Dec(k,\ Enc(k,m)\ ) = m}$

Example: Substitution Cipher

K?
M?
C?

Example: Transposition Cipher

K?
M?
C?

Example: One-Time Pad

K?
M?
C?

Example: Vigenère Cipher

K?
M?
C?

Encryption Security?

Questions to think about:

<u>What kind of messages?</u>
<u>What does the adversary already know?</u>
<u>What information are we trying to protect?</u>

Examples:

- Messages are always either “attack at dawn” or “attack at dusk”, trying to hide which is the case
- Messages are status updates (“<person> reports <event> at <location>”). Which data is sensitive?

Encryption Security?

Questions to think about:

<u>What kind of messages?</u>
<u>What does the adversary already know?</u>
<u>What information are we trying to protect?</u>

Goal:

Rather than design a separate system for each use case, design a system that works in all possible settings

Semantic Security

Idea:

- Plaintext comes from an arbitrary distribution
- Adversary initially has some information about the plaintext
- Seeing the ciphertext should not reveal any more information
- Model unknown key by assuming it is chosen uniformly at random

(Perfect) Semantic Security

Definition: A scheme **(Enc,Dec)** is **(perfectly) semantically secure** if, for all:

- Distributions **D** on **M** ← Plaintext distribution
- Functions $\mathbf{I:M\rightarrow\{0,1\}^*}$ ← Info adv gets
- Functions $\mathbf{f:M\rightarrow\{0,1\}^*}$ ← Info adv tries to learn
- Functions $\mathbf{A:C\times\{0,1\}^*\rightarrow\{0,1\}^*}$ ← Adversary

"Simulator"

There exists a function $\mathbf{S:\{0,1\}^*\rightarrow\{0,1\}^*}$ such that

$$\Pr[\ A(\ Enc(k,m)\ ,\ I(m)\ )\ =\ f(m)\ ]$$
$$=\Pr[\ S(\ I(m)\ )\ =\ f(m)\ ]$$

where probabilities are taken over $\mathbf{k\leftarrow K,\ m\leftarrow D}$

Semantic Security

Captures what we want out of an encryption scheme

But, complicated, with many moving parts

Want: something simpler…

Notation

Two random variables $\mathbf{X,Y}$ over a finite set $\mathbf{S}$ have identical distributions if, for all $\mathbf{s} \in \mathbf{S}$,

$$\Pr[\ X = s] \ = \ \Pr[\ Y = s]$$

In this case, we write

$$X \overset{d}{=} Y$$

Perfect Secrecy [Shannon'49]

Definition: A scheme **(Enc,Dec)** has **perfect secrecy** if, for any two messages $\mathbf{m_0}, \mathbf{m_1} \in \mathbf{M}$

$$\mathbf{Enc}(\mathbf{K}, \mathbf{m_0}) \stackrel{d}{=} \underbrace{\mathbf{Enc}(\mathbf{K}, \mathbf{m_1})}$$

Random variable corresponding to uniform distribution over **K**

Random variable corresponding to encrypting $\mathbf{m_1}$ using a uniformly random key

Semantic Security = Perfect Secrecy

Theorem: A scheme **(Enc,Dec)** is semantically secure if and only if it has perfect secrecy

Perfect Secrecy $\Rightarrow$ Semantic Security

Given arbitrary:

- Distribution $\mathbf{D}$ on $\mathbf{M}$
- Function $\mathbf{I:M \rightarrow \{0,1\}^*}$
- Function $\mathbf{f:M \rightarrow \{0,1\}^*}$
- Function $\mathbf{A:C \times \{0,1\}^* \rightarrow \{0,1\}^*}$

Know: $\mathbf{E(K,\ m_0)} \stackrel{d}{=} \mathbf{E(K,\ m_1)}$

Goal: Construct $\mathbf{S:\{0,1\}^* \rightarrow \{0,1\}^*}$ such that

$$\mathbf{Pr[\ A(\ Enc(k,m)\ ,\ I(m)\ ) = f(m)\ ]}$$
$$\mathbf{= Pr[\ S(\ I(m)\ ) = f(m)\ ]}$$

Perfect Secrecy $\Rightarrow$ Semantic Security

S(i):

- Choose random $\mathbf{k} \leftarrow \mathbf{K}$
- Set $\mathbf{c} \leftarrow \mathbf{Enc(k,0)}$
- Run and output $\mathbf{A(c,i)}$

$$\Pr[\, S(\, I(m)\,) = f(m)\,]$$
$$= \Pr[\, A(\, Enc(k,0)\,,\, I(m)\,) = f(m) : m \leftarrow D\,]$$
$$= \Sigma_{m,c} \Pr[D=m] \Pr[Enc(K,0)=c] \Pr[\, A(c,I(m)) = f(m)\,]$$
$$= \Sigma_{m,c} \Pr[D=m] \Pr[Enc(K,m)=c] \Pr[\, A(c,I(m)) = f(m)\,]$$
$$= \Pr[\, A(\, Enc(k,m)\,,\, I(m)\,) = f(m)\,]$$

Semantic Security $\Rightarrow$ Perfect Secrecy

Proof by contrapositive:

- Assume $\exists$ $\mathbf{m_0, m_1}$ s.t. $\mathbf{Enc(K, m_0)} \overset{d}{\neq} \mathbf{Enc(K, m_1)}$
- Devise **D,I,f,A** such that no **S** exists

D: pick $\mathbf{b \leftarrow \{0,1\}}$ at random, output $\mathbf{m_b}$

I: empty

$\mathbf{f(m_b) = b}$

$\mathbf{A(c) = 1}$ iff $\mathbf{Pr[Enc(K,m_1) = c] > Enc(K,m_0) = c]}$

Semantic Security $\Rightarrow$ Perfect Secrecy

Let $\mathbf{T} = \{c : \Pr[\mathrm{Enc}(K,m_1) = c] > \mathrm{Enc}(K,m_0) = c]\}$

$$\Pr[\ A(\ \mathrm{Enc}(K,m)\ ) = f(m) : m \leftarrow D]$$

$$= \tfrac{1}{2} \Pr[A(\ \mathrm{Enc}(K,m_0)\ ) = 0\ ] + \tfrac{1}{2} \Pr[A(\ \mathrm{Enc}(K,m_1)\ ) = 1\ ]$$

$$= \tfrac{1}{2} \Pr[\ \mathrm{Enc}(K,m_0) \notin T] + \tfrac{1}{2} \Pr[\ \mathrm{Enc}(K,m_1) \in T]$$

$$= \tfrac{1}{2} + \tfrac{1}{2} (\Pr[\ \mathrm{Enc}(K,m_1) \in T] - \Pr[\ \mathrm{Enc}(K,m_0) \in T])$$

Semantic Security $\Rightarrow$ Perfect Secrecy

$$\Pr[\,\mathrm{Enc}(K,m_b) \in T\,]$$
$$= \Sigma_{c \in T} \Pr[\mathrm{Enc}(K,m_b) = c]$$
$$= 1 - \Sigma_{c \notin T} \Pr[\mathrm{Enc}(K,m_b) = c]$$

$$\Pr[\,\mathrm{Enc}(K,m_1) \in T] - \Pr[\,\mathrm{Enc}(K,m_0) \in T]$$
$$= \Sigma_{c \in T} \Pr[\mathrm{Enc}(K,m_1) = c] - \Pr[\mathrm{Enc}(K,m_0) = c]$$
$$= \Sigma_{c \notin T} \Pr[\mathrm{Enc}(K,m_0) = c] - \Pr[\mathrm{Enc}(K,m_1) = c]$$
$$= \tfrac{1}{2}\, \Sigma_c \,|\, \Pr[\Pr[\mathrm{Enc}(K,m_1)=c] - \Pr[\mathrm{Enc}(K,m_0)=c]\,|$$

$$\Rightarrow \Pr[\,A(\,\mathrm{Enc}(K,m)\,) = f(m) : m \leftarrow D] > 1/2$$

Semantic Security $\Rightarrow$ Perfect Secrecy

Recall:
D: pick $\mathbf{b} \leftarrow \{0,1\}$ at random, output $\mathbf{m_b}$
I: empty
$\mathbf{f(m_b) = b}$

We just proved:
$\mathbf{Pr[\ A(\ Enc(K,m)\ ) = f(m) : m \leftarrow D] > 1/2}$

On the other hand, any **S** gets no input and tries to guess a random bit **b**
$\Rightarrow \mathbf{Pr[\ S(\ I(m)\ ) = f(m)\ ] = 1/2}$

Perfect vs. Semantic Security

Semantic security is the "right" notion to intuitively capture the desired security goals

Perfect is much simpler and easier to reason about

Fortunately, we know both are identical
$\Rightarrow$ perfect security is almost always what is used

Any perfectly/semantically secure schemes?

Proper Use Case for Perfect Security

- Message can come from any distribution ✓
- Adversary can know anything about message ✓
- Encryption hides anything ✓

- But, definition only says something about an adversary that sees a single message ✘

  $\Longrightarrow$ If two messages, no security guarantee

- Assumes no side-channels ✘
- Assumes key is uniformly random ✘

Next Time

How to shrink key length

How to handle multiple messages

Reminders

Course website: cs.princeton.edu/~mzhandry/2020-Spring-COS433

- Enroll in Piazza
- Fill out OH poll
- HW1, PR1 to be released next week

Find teams for projects (up to 4)