COS433/Math 473: Cryptography

Mark Zhandry
Princeton University
Spring 2017
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”

None of these capture the true breadth of the field
My Definition

Cryptography is about using secrets to solve interesting tasks

 stil doesn’t capture everything)
Cryptography Is Everywhere
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
COS 433

Inherent to the study of crypto
- Working knowledge of fundamentals is crucial
- Cannot discern security by experimentation
- Proofs, reductions, probability are necessary
What you should expect to learn:

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

Bonus:

• Debunking some Hollywood crypto
• 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 (mzhandry@pr)
TAs: Udaya Ghai (udayaghai@gm)
Qipeng Liu (qipengl@pr)

Lectures: MW 1:30-2:50pm, Friend 008

Webpage: cs.princeton.edu/~mzhandry/2018-Spring-COS433/

Office Hours: please fill out HW0 Doodle poll
Piazza

https://piazza.com/class/jb0zp9b0blf3o0

Main channel of communication
• Course announcements
• Discuss homework problems with other students
• Find project/study groups
• Ask content questions to instructors, other students
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 per week
• No dropped/late hws, but “extra credit”
• Only typed solutions, submitted via CS Dropbox
• Collaboration encouraged, but write up own solutions

30% Projects
• More details at the end of class today

30% Take-home Final
• Individual
Classroom Policies

Please stop me if you have any questions

Please come to class to be engaged and to learn
• Notes for each lecture will be added to the webpage
• I don’t take attendance
• Don’t be on Facebook, working on assignments, etc

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
Today
“Pre-modern” Crypto Part I: Substitution 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

\[ c = \text{Enc}(k, m) \]

\[ m = \text{Dec}(k, c) \]

For our discussions, assume \text{Enc}, \text{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

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 ($k = 0, \ldots, 25$)

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**

<table>
<thead>
<tr>
<th>Key</th>
<th>Plaintext</th>
<th>Key</th>
<th>Plaintext</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>HJETG HTRGTI BTHHPVT</td>
<td>13</td>
<td>UWRGT UGETGV OGUUCIG</td>
</tr>
<tr>
<td>1</td>
<td>IKFUH IUSHUJ CUIIQWU</td>
<td>14</td>
<td>VXSHU VHFUHW PHVVDJH</td>
</tr>
<tr>
<td>2</td>
<td>JLGVI JVTIVK DVJJRXV</td>
<td>15</td>
<td>WYTIV WIGVIX QIWWEKI</td>
</tr>
<tr>
<td>3</td>
<td>KMHWJ KWUJWL EWKKSYW</td>
<td>16</td>
<td>XZUJW XJHWJY RJXXFLJ</td>
</tr>
<tr>
<td>4</td>
<td>LNIKK LXVKXM FXLLTZX</td>
<td>17</td>
<td>YAVKK YKIXKZ SKYYGMK</td>
</tr>
<tr>
<td>5</td>
<td>MOJYL MYWLYN GYMMUAY</td>
<td>18</td>
<td>ZBWLY ZLJYLA TLZZHN</td>
</tr>
<tr>
<td>6</td>
<td>NPKZM NZXMOZ HZNNVBZ</td>
<td>10</td>
<td>ACXMZ AMKZMB UMAAIOM</td>
</tr>
<tr>
<td>7</td>
<td>OQLAN OAYNAP IAOOWCA</td>
<td>20</td>
<td>BDYNA BNLANC VNBBJP</td>
</tr>
<tr>
<td>8</td>
<td>PRMBO PBZOBQ JBPPXDE</td>
<td>21</td>
<td>CEZOB COMBOD WOCCKQ</td>
</tr>
<tr>
<td>9</td>
<td>QSNJP QCAPCR KCQYYEC</td>
<td>22</td>
<td>DFAPC DPNCPE XPDDLR</td>
</tr>
<tr>
<td>10</td>
<td>RTODQ RDBQDS LDRZZFD</td>
<td>23</td>
<td>EGBQD EQODQF QEEEMSR</td>
</tr>
<tr>
<td>11</td>
<td><strong>SUPER SECRET MESSAGE</strong></td>
<td>24</td>
<td>FHCRE FRPERG ZRFFFNT</td>
</tr>
<tr>
<td>12</td>
<td>TVQFS TDFSUF NFTTBHF</td>
<td>25</td>
<td>GIDSF GSQFSH ASGGOUS</td>
</tr>
</tbody>
</table>
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

Example:

plaintext: super secret message
ciphertext: ARQYV AYSVYX EYAAAFJY

Number of possible keys?

26! ≈ 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 NRFNYCYP R 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:
\( e \rightarrow R, \ t \rightarrow C, \ a \rightarrow Z, \ o \rightarrow O \)
Example

BoFt HNe a NHMNtHyHtYoF KYIVeG to eFKoBe
NefNYtYPEe Bata ePeF tVoHXV tVeE aGe
GenYtYFt to a MGHte WoGKe attaKU.
YFBeeB, ME KOHFTYFX TetteGN aFB KoDIaGYFX
to tEiYKaN teQt, EoH KaF GeKoPeG tVe
ITayFteQt an LETT an tVe Uee

Maybe “data”? Maybe “attack”? Probably “the”
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

“as”?  “and”?  “encode”?  “are”?
Example

don't use a scheme to encode sensitive data, even though the experts are resistant to a brute force attack.
Indeed, even the experts and coders to recover the plaintext as the key.

“indeed”? “even”? “force”? “recover”?
Example

don't use a substitution cipher to encode sensitive data, even though the are resistant to a brute force attack. Indeed, ME countinX Tletters and codINarIX to tEIicaT teQt, Eou can recover the ITainteQt as LeTT as the key
Example

don't 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.

abcdefghijklmnopqrstuvwxyz

ABCDEFGHIJKLMNOPQRSTUVWXYZ
Problem with Substitution

Differing letter frequencies reveal a lot
Polybius Square

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td>e</td>
</tr>
<tr>
<td>2</td>
<td>f</td>
<td>g</td>
<td>h</td>
<td>i</td>
<td>j</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>m</td>
<td>n</td>
<td>o</td>
<td>p</td>
</tr>
<tr>
<td>4</td>
<td>q</td>
<td>r</td>
<td>s</td>
<td>t</td>
<td>u</td>
</tr>
<tr>
<td>5</td>
<td>v</td>
<td>w</td>
<td>x</td>
<td>y</td>
<td>z</td>
</tr>
</tbody>
</table>

plaintext: super secret message

ciphertext: 4345351542 431513421544 32154343112215
Keyed Polybius Square

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>y</td>
<td>n</td>
<td>r</td>
<td>b</td>
<td>f</td>
</tr>
<tr>
<td>2</td>
<td>d</td>
<td>l</td>
<td>w</td>
<td>o</td>
<td>g</td>
</tr>
<tr>
<td>3</td>
<td>s</td>
<td>p</td>
<td>a</td>
<td>t</td>
<td>k</td>
</tr>
<tr>
<td>4</td>
<td>h</td>
<td>v</td>
<td>i</td>
<td>j</td>
<td>x</td>
</tr>
<tr>
<td>5</td>
<td>q</td>
<td>u</td>
<td>z</td>
<td>e</td>
<td>m</td>
</tr>
</tbody>
</table>

plaintext: super secret message

ciphertext: 3152325413 315445135434 55543131332554
Frequency of Polybius?

1
2
3
4
5
Frequency of Polybius?
General Alphabets

Pttx and cttx need not be the same
• cttx symbols can be letters, (tuples of) numbers, etc.
• pttx symbols can also numbers, bits, bytes

In general, changing cttx alphabet doesn’t affect security of cipher
• Keyed Polybius = Un-keyed Polybius + Substitution

Other reasons to change ciphertext alphabet?
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%
  • 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

```
<table>
<thead>
<tr>
<th>Y</th>
<th>N</th>
<th>R</th>
<th>B</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>L</td>
<td>W</td>
<td>O</td>
<td>G</td>
</tr>
<tr>
<td>S</td>
<td>P</td>
<td>A</td>
<td>T</td>
<td>K</td>
</tr>
<tr>
<td>H</td>
<td>V</td>
<td>IJ</td>
<td>X</td>
<td>C</td>
</tr>
<tr>
<td>Q</td>
<td>U</td>
<td>Z</td>
<td>E</td>
<td>M</td>
</tr>
</tbody>
</table>
```
Polygraphic Substitution

Example: Playfair cipher
• Invented by Sir Charles Wheatstone in 1854
• Used by British until WWII
• To encode, choose opposite corners of rectangle
Polygraphic Substitution

Example: Playfair cipher
• Invented by Sir Charles Wheatstone in 1854
• Used by British until WWII

- 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 $> 26^d$ 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
Homophonic Substitution
Homophonic Substitution

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

⟹ Thwarts vanilla frequency analysis

However, still possible to cryptanalyze
• Frequency analysis on tuples of letters will no longer be uniform
Homophonic Substitution

Example: “Grand Chiffre” (Great Cipher)
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
• 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: Kasiski examination, superposition
Kasiski Examination

Published 1863, apparently known to Babbage as early as 1840’s

Example:

key:  cryptocryptocryptocryptocryptocryptocryptocryptocryptocryptocryptocryptocryptocrypt
ptxt:  acannercanacanacanacanacanacanacanacanacanacanacanacanacanacans
ctxt:  CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG

All RED/PURPLE chunks are multiples of 6 apart
• Good indication that the key length is 1, 2, 3, or 6
Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example: shift by 1

CTYCGSTTYCVOPRQBTBATYCLUDRAPGBGIAPGQCEAPGG
CTYCGSTTYCVOPRQBTBATYCLUDRAPGBGIAPGQCEAPGG
Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example: shift by 2

CTYCGSTTTYCVOPRQBTBATYCLUDRAPGBGIAPGQCEAPGG
   CTYCGSTTTYCVOPRQBTBATYCLUDRAPGBGIAPGQCEAPGG
Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example: shift by 3

CTYCGSTTTYCVOPRQBTBATYCLUDAPGBGIAPGQCEAPGG

CTYCGSTTTYCVOPRQBTBATYCLUDAPGBGIAPGQCEAPGG
Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example: shift by 4

CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG
CTYCGSTTYCVOPRQBTBATYCLOURAPGBGIAPGQCEAPGG

1 2 3 4
Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example: shift by 5

CTYCGSTTYCVOPRQBTBATYCLUDRAPGBGIAPGQCEAPGG

CTYCGSTTYCVOPRQBTBATYCLUDRAPGBGIAPGQCEAPGG
Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example: shift by 6

CTYCGSTTYCVOPRQBTBATYCLOURAPGBG1APGQCEAPGG
CTYCGSTTYCVOPRQBTBATYCLOURAPGBG1APGQCEAPGG

1 2 3 4 5 6

Histogram showing distribution of matches for different shifts.
Superposition

Compare shifts of ciphertext, looking for shifts containing many matches

Example: shift by 7

CTYCGSTTYCVOPRQBTB4ATYCL6OURAPGBGIIAPGQCEAPCGG
CTYCGSTTYCVOPRQBTB4ATYCL6OURAPGBGIIAPGQCEAPCGG

1 2 3 4 5 6 7
Superposition

Why does it work?

For shifts that are multiples of key size:
• Both bottom and top ciphertexts encrypted with same key
• \(#(ctxt \text{ matches}) = #(ptxt \text{ matches})\)
  \[ \approx |ptxt| \times \text{col. prob. for English} \]
  \[ \approx |ptxt| \times 0.065 \]
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 $\frac{1}{26} \approx 0.038$
• $\#(\text{ctxt matches}) \approx |\text{ptxt}| * 0.038$
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
Advantages

1945: Claude Shannon proved that if:
- A truly random key is used
- The key is only used to encrypt only one message
- And the key is longer than the message

Then the scheme is *perfectly secure*
Notation

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

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

In this case, we write

$$X \overset{d}{=} Y$$
Definition: A scheme \((\text{Enc},\text{Dec})\) has **perfect secrecy** if, for any two messages \(m_0, m_1 \in M\)

\[
\text{Enc}(K, m_0) \overset{d}{=} \text{Enc}(K, m_1)
\]

Random variable corresponding to uniform distribution over \(K\)

Random variable corresponding to encrypting \(m_1\) using a uniformly random key
Perfect Secrecy of One-time Pad

For concreteness, use XOR (Vernam cipher); applies equally well to other variants of one-time pad

Key space $K = \{0,1\}^n$
Message space $M = \{0,1\}^n$
Ciphertext space $C = \{0,1\}^n$

$\text{Enc}(k, m) = k \oplus m$
$\text{Dec}(k, c) = k \oplus c$
Perfect Secrecy of One-time Pad

**Theorem:** For any message $m \in \{0,1\}^n$ and ciphertext $c \in \{0,1\}^n$,

$$\Pr[\text{Enc}(k, m) = c] = 2^{-n}$$

**Proof:**

$$\Pr[\text{Enc}(k, m) = c] = \Pr[ k \oplus m = c ]$$

$$= \Pr[ k = c \oplus m ]$$

$$= 2^{-n}$$
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 2018 dollars)

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

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

Try to encrypt two messages, security will fail

\[
\text{Enc}(k,m_0) \oplus \text{Enc}(k,m_1) \\
= (k \oplus m_0) \oplus (k \oplus m_1) \\
= m_0 \oplus m_1
\]

Enough redundancy in English text to usually recover messages from XOR
Project 1: Cryptanalysis
Project 1: Cryptanalysis

**Setup:** you’re an intern at a super secret intelligence agency, which is trying to decrypt a batch of documents

**What you know:**
- All pencil-and-paper ciphers
- All based on schemes we’ll see this week

**Your task:**
- Use what you’ve learned to decrypt the documents
Part 0: Form Teams

Due: Friday February 9th

Instructions:
• Teams of 2-3 people
• Sign up on Google spreadsheet
• Use Piazza team-finding feature if necessary

Documents will be released to teams by Feb 10th
Part 1: Basic Analysis (15%)

Due: Tuesday February 20th

Instructions: tell us as much as possible about each document
• Which documents encrypted by same means?
• What cipher used?
• Parameters of cipher (key length, etc)

Main purpose is to give early feedback
Part 2: Cryptanalysis (85%)

Due: Tuesday March 6\textsuperscript{th}

Instructions: Actually decrypt the documents
• Also, give thorough write-up on your methodology
• Also, report on intelligence gathered

For both parts 1 and 2, you should definitely make use of computers to perform analysis
• Please submit your code
Competition

Submit any (partially) decrypted documents early and often

Every Monday morning, teaching staff will test how well you’ve done so far
• Most successful team will receive 2 bonus points
• Runner up will receive 1 bonus point

Bonus dates:
   Feb 19th, Feb 26th, March 5th
Reminders

By Friday Feb 9\textsuperscript{th}:
\begin{itemize}
  \item HW0: Fill out OH Doodle poll
  \item Find Teams for Project 1
\end{itemize}

Due Tuesday Feb 13\textsuperscript{th}:
\begin{itemize}
  \item HW1, on course webpage