COMP547

Claude Crépeau

INTRODUCTION TO

MODERN CRYPTOGRAPHY

Second Edition

Jonathan Katz •Yehuda Lindell

Chapter 1 : Introduction

1.1 Cryptography and Modern Cryptography

- The Concise Oxford Dictionary (2006) defines "*cryptography*" as

"the art of writing or solving codes".

- With "*codes*" elsewhere defined as

"a system of pre-arranged signals, especially used to ensure secrecy in transmitting messages".

1.1 Cryptography and Modern Cryptography

- Historically accurate, but it does not capture the essence of modern cryptography.
- Focuses solely on the problem of secret communication.

1.1 Cryptography and Modern Cryptography

- Until the 20th century, cryptography was an art.
  - Constructing good codes, or breaking existing ones, relied on creativity and personal skill.
  - Very little theory that could be relied upon
  - Not even a well-defined notion of a *good code*.
  - Late 20th century: rich theory emerged,
  - Rigorous cryptography as a science.

1.1 Cryptography and Modern Cryptography

- Cryptography now encompasses much more than secret communication.
  - message authentication, digital signatures,
  - protocols for exchanging secret keys, authentication protocols,
  - electronic auctions and elections, digital cash,
  - any distributed computation that may come under internal or external attack.

1.1 Cryptography and Modern Cryptography

- A definition of *modern cryptography*:

  “the mathematical study of techniques for securing digital information, transactions, and distributed computations against adversarial attacks”.

1.1 Cryptography and Modern Cryptography

Historically, the major consumers of cryptography were military and intelligence organizations.

Today, cryptography is everywhere!

- Security mechanisms: integral part of almost any computer system.
- Users (often unknowingly) every time they access a secured website.
- Cryptographic methods enforce access control in multi-user operating systems, and prevent thieves from extracting trade secrets from stolen laptops.
- Software protection methods employ encryption, authentication, and other tools to prevent copying.

1.1 Cryptography and Modern Cryptography

- Cryptography has gone from

  “an art form that dealt with secret communication for the military”

  to

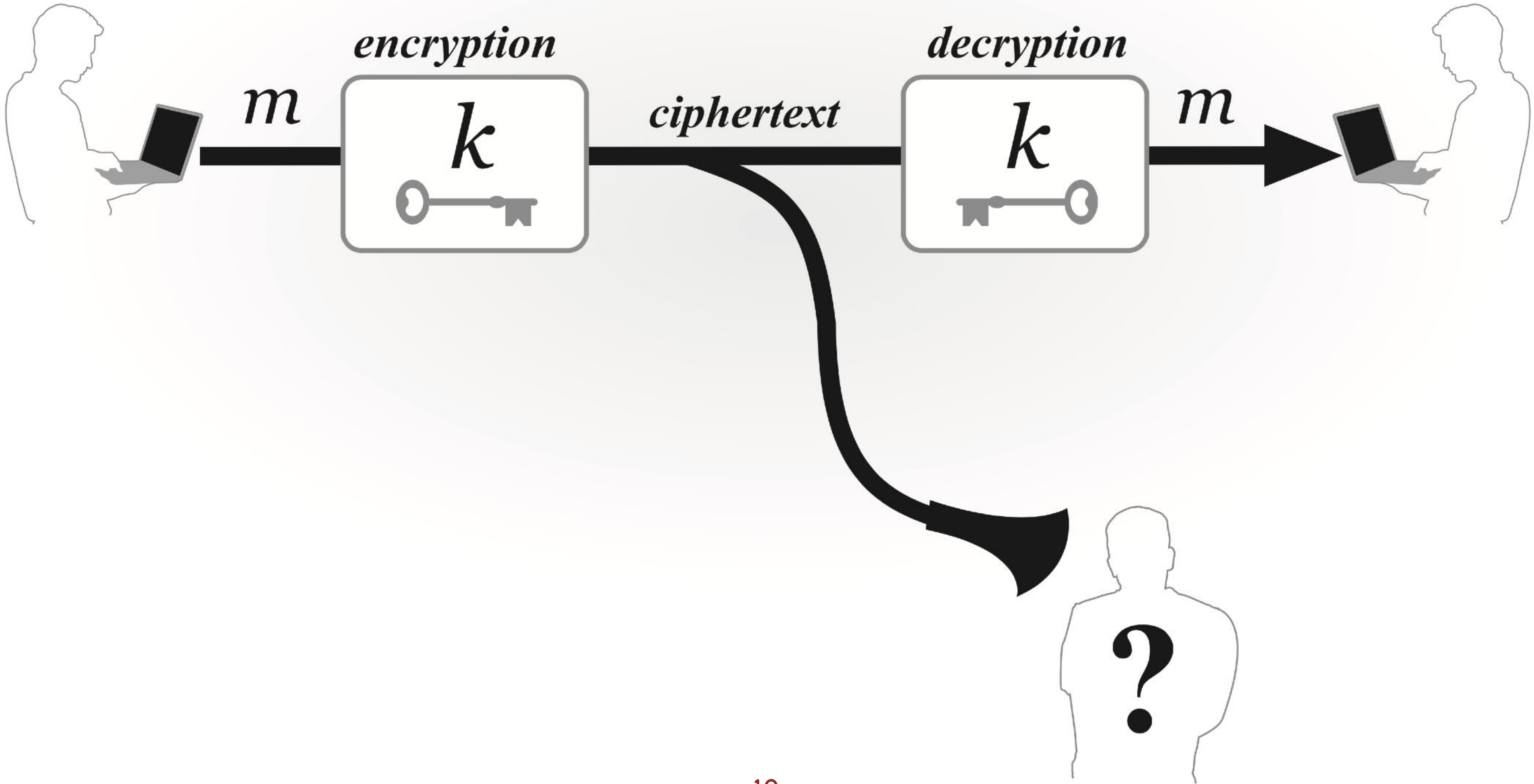
  “a science to secure systems for ordinary people all across the globe”.

- Cryptography is becoming a more and more central topic within computer science.
- Focus of this course/book is *modern cryptography*.

1.1 Cryptography and Modern Cryptography

- Begin our study by examining the state of cryptography before the changes above.
- Provides an understanding of where cryptography has come from so that we later appreciate how much it has changed.
- The study of "classical cryptography" (ad-hoc constructions, simple ways to break them)serves as motivation for more rigorous approach.

1.2 The Setting of Private-Key Encryption



Syntax of encryption.

- Private-key encryption scheme:

1. Key-generation algorithm **Gen** :
   probabilistic algorithm that outputs a key ***k*.**
   (chosen according to distribution determined by the scheme)

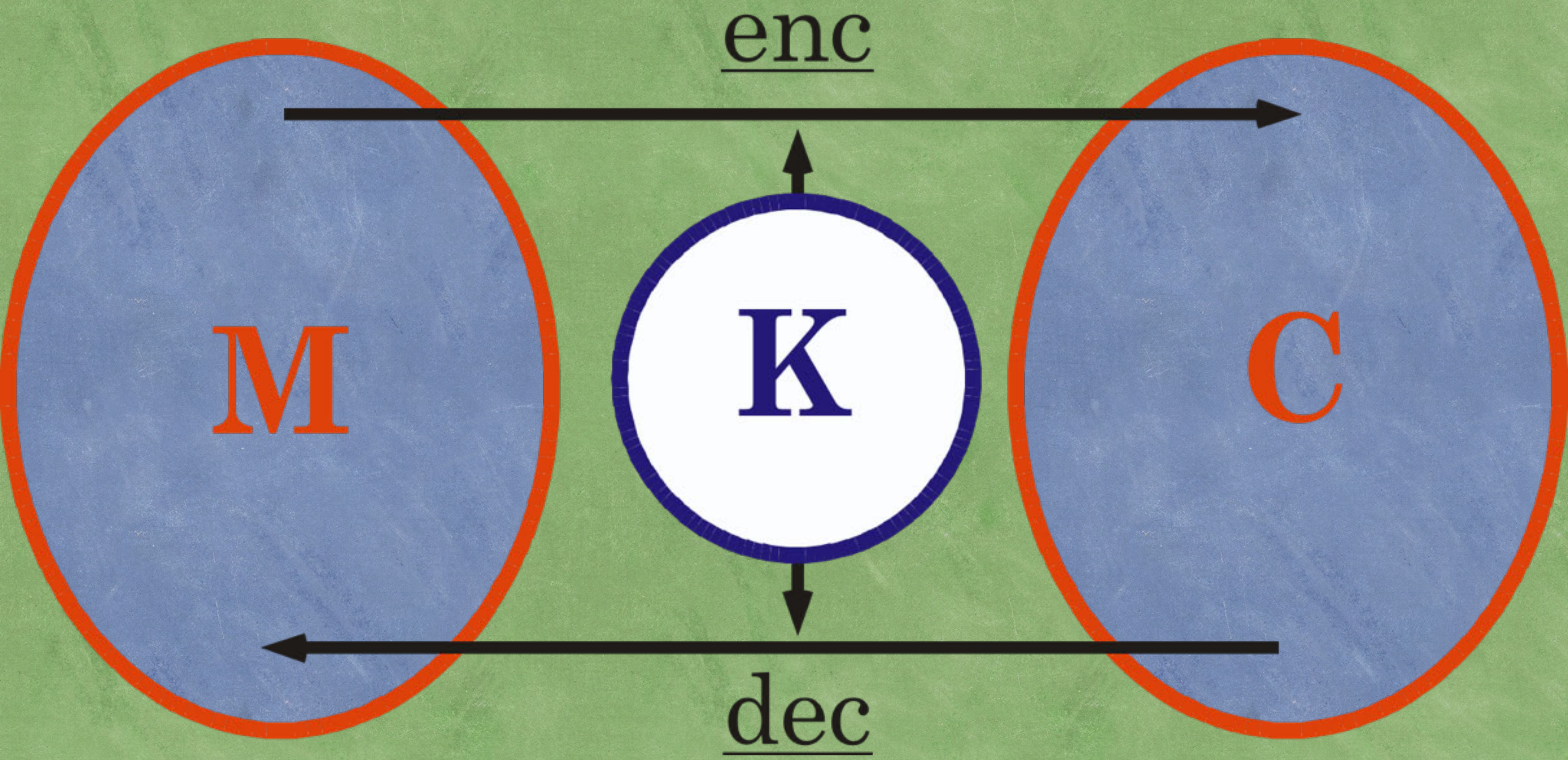
2. Encryption algorithm **Enc** :
   inputs a key ***k*** and a plaintext message **m** and outputs a ciphertext **c**.

   (**$\mathbf{Enc}_{k}(m)$** = encryption of ***m*** using key ***k***.)

3. Decryption algorithm **Dec** :
   inputs a key ***k*** and a ciphertext ***c*** and
   outputs a plaintext ***m***.

   (**$\mathbf{Dec}_{k}(c)$ =** decryption of ***c*** using the key ***k***.)

symmetric encryption



Syntax of encryption.

- Key space ($\mathcal{K}$):
  set of all possible keys output by the key-generation algorithm.
- Almost always, **Gen** simply chooses a key uniformly at random from $\mathcal{K}$.
- Plaintext (or message) space ($\mathcal{M}$):
  set of all "legal" messages
  (i.e., those supported by the encryption algorithm).

Syntax of encryption.

- Since any ciphertext is obtained by encrypting some plaintext under some key, the sets $\mathcal{K}$ and $\mathcal{M}$ together define a set of all possible ciphertexts denoted by C.
- An encryption scheme is fully defined by specifying the three algorithms **(Gen, Enc, Dec)** and the plaintext space $\mathcal{M}$.

Syntax of encryption.

- ***Correctness*** requirement of any encryption scheme is that for all ***k*** output by **Gen** and for all ***m*** $\in \mathcal{M}$, it holds that

$$\mathbf{Dec}_{\boldsymbol{k}}(\mathbf{Enc}_{\boldsymbol{k}}(\boldsymbol{m})) = \boldsymbol{m}.$$

- In words,
  decrypting a ciphertext (using the appropriate key) yields the original message that was encrypted.

Syntax of encryption.

- First, **Gen** is run to obtain a key ***k*** that the parties share.
- When one party wants to send a plaintext ***m*** to the other,
  - he computes $\boldsymbol{c} := \mathbf{Enc}_{\boldsymbol{k}}(\boldsymbol{m})$
  - he sends the resulting ciphertext ***c*** over the public channel to the other party.
- Upon receiving ***c***, the other party computes $\boldsymbol{m} := \mathbf{Dec}_{\boldsymbol{k}}(\boldsymbol{c})$ to recover the original plaintext.

Keys and Kerckhoffs' principle

- An adversary who knows the algorithm **Dec** and the key ***k*** shared by the two communicating parties, is able to decrypt all communication between these parties.

- It is for this reason that the communicating parties must share the key ***k*** secretly, and keep ***k*** completely secret from everyone else.

- Maybe they should keep the decryption algorithm **Dec** a secret, too?

Kerckhoffs' principle

Kerckhoffs’ principle

- The plaintext space M is typically assumed to be known, e.g., it may consist of English sentences.
- Perhaps all the algorithms constituting the encryption scheme (i.e., **Gen, Enc, Dec**) should be kept secret ?
- In the late 19th century, Auguste Kerckhoffs gave his opinion on this matter in a paper outlining important design principles for military ciphers.

Kerckhoffs’ principle

- One of the most important of these principles (now known simply as Kerckhoffs’ principle) is :

  “The cipher method must not be required to be secret, and it must be able to fall into the hands of the enemy without inconvenience.”

- In other words, the encryption scheme itself should not be kept secret, and so only the key should constitute the secret information.

Kerckhoffs' principle

Three primary arguments in favor of Kerckhoffs.

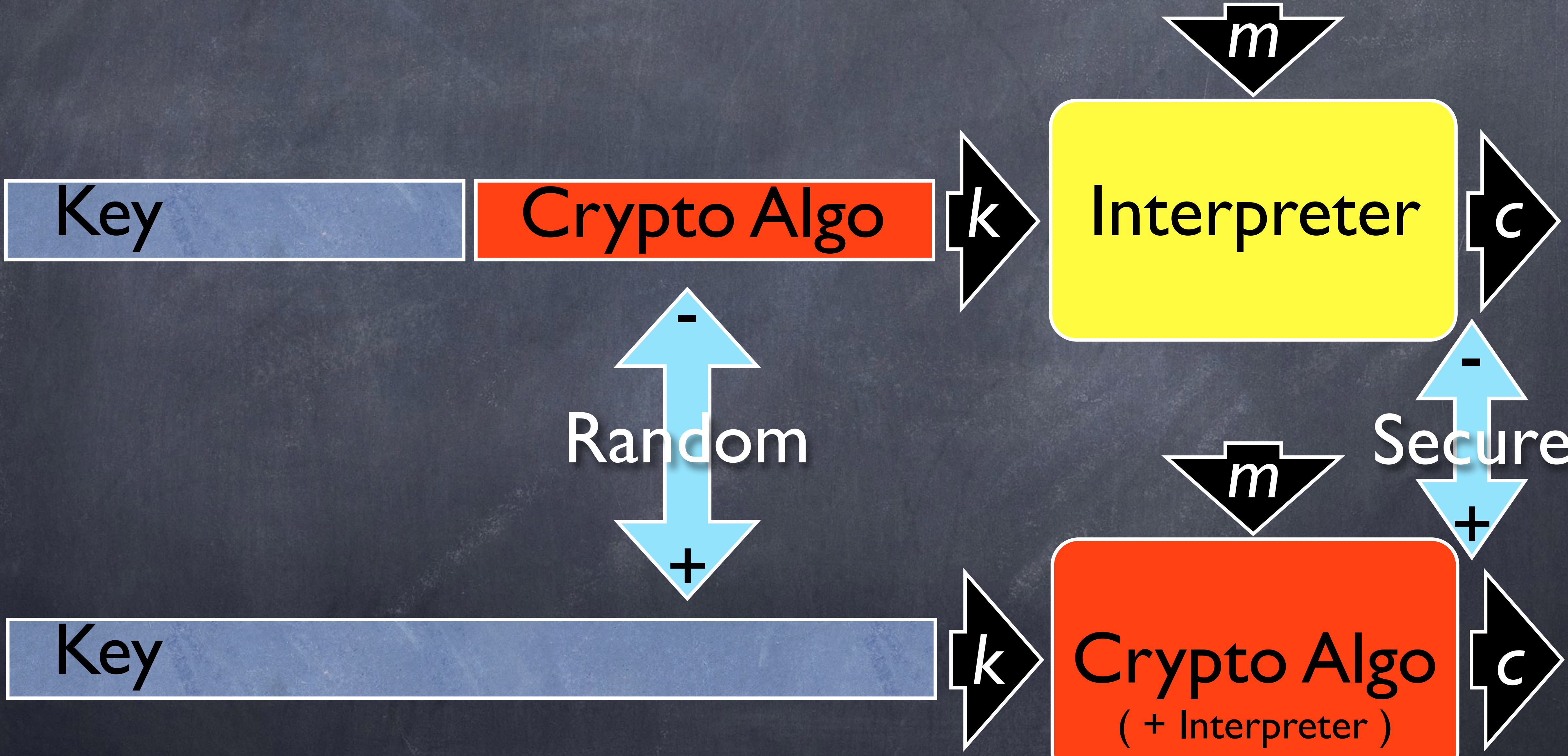
1. Much easier for the parties to maintain secrecy of a short key than to maintain secrecy of an algorithm.

   Easier to share a short (say, 100-bit) string and store this string securely than it is to share and securely store a program that is much larger.

   Furthermore, details of an algorithm can be leaked (by insider?) or learned through reverse engineering.

1/3

Kerckhoffs' principle



1/3

Kerckhoffs' principle

Three primary arguments in favor of Kerckhoffs.

2. In case the key is exposed, it will be much easier for the honest parties to change the key than to replace the algorithm being used.

   It is good security practice to refresh a key frequently even when it has not been exposed.

2/3

Kerckhoffs' principle

Three primary arguments in favor of Kerckhoffs.

3. In case many pairs of people (say, within a company) need to encrypt their communication, it will be significantly easier for all parties to use the same algorithm/program, but different keys, than for everyone to use a different program.

3/3

“security by obscurity” vs “open cryptographic design”

Today, Kerckhoffs’ principle is that

“*all algorithms be made public.*”

1. Published designs undergo public scrutiny and are therefore likely to be stronger.

   Many years of experience have demonstrated that it is very difficult to construct good encryption schemes.

   Our confidence in the security of a scheme is much higher if it has been extensively studied and no weaknesses have been found.

1/3

“security by obscurity” vs “open cryptographic design”

Today, Kerckhoffs’ principle is that

“all algorithms be made public.”

2. It is better for security flaws, if they exist, to be revealed by “ethical hackers” (leading, hopefully, to the system being fixed) rather than having these flaws be known only to malicious parties.

2/3

"security by obscurity" vs "open cryptographic design"

Today, Kerckhoffs' principle is that

"*all algorithms be made public.*"

3. If the security of the system relies on the secrecy of the algorithm, then reverse engineering of the code (or leakage by industrial espionage) poses a serious threat to security.

   This is in contrast to the secret key which is not part of the code, and so is not vulnerable to reverse engineering.

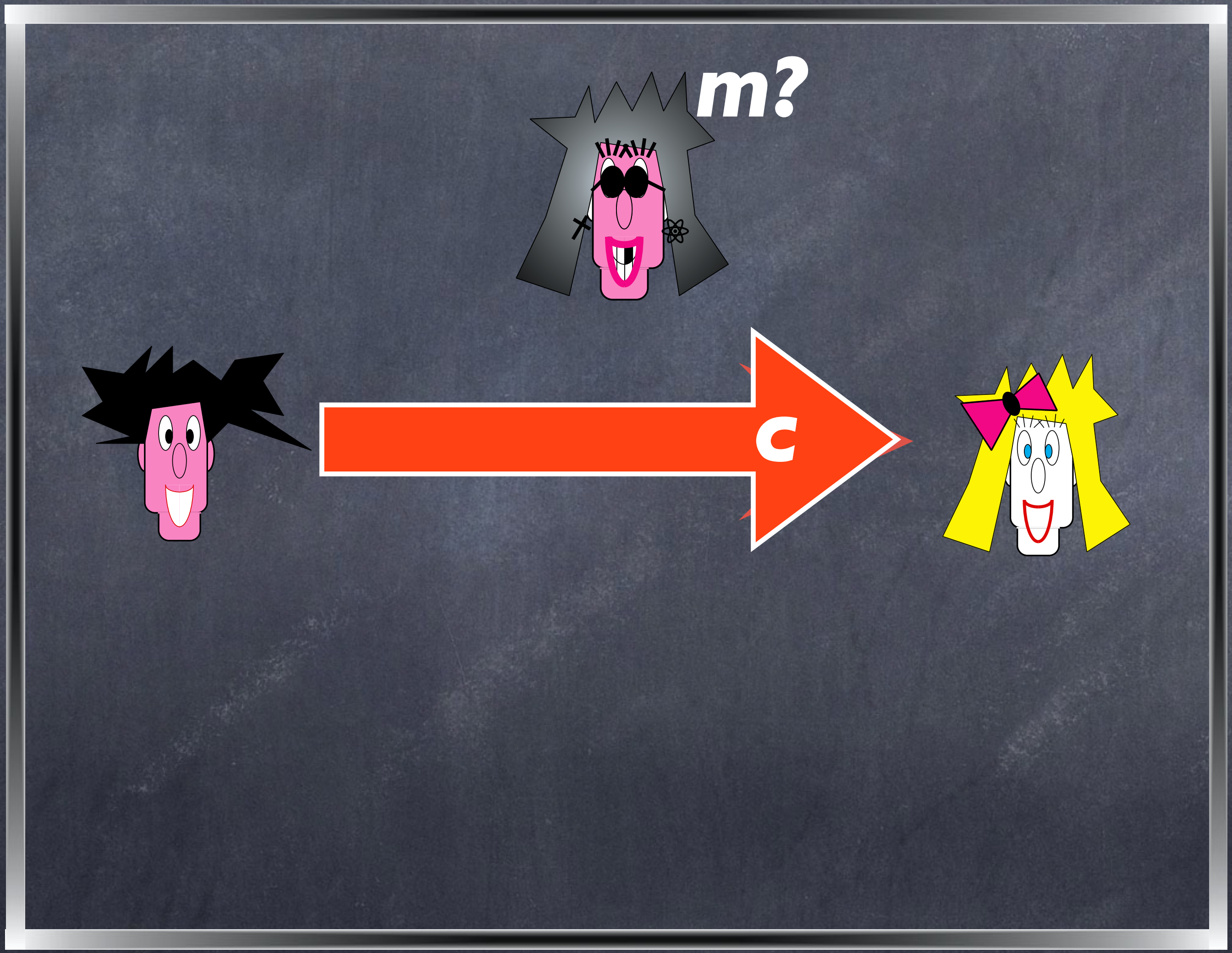
3/3

Kerckhoffs’ principle

- As simple and obvious as it may sound, the principle of open cryptographic design **is ignored** over and over again with disastrous results.
- Very **dangerous** to use a proprietary algorithm (i.e., a non-standardized algorithm that was designed in secret by someone), and only publicly tried and tested algorithms should be used.
- Enough **good algorithms** that are standardized but not patented, so that there is no reason whatsoever today to use something else.

Attack scenarios

- **Ciphertext-only attack:**

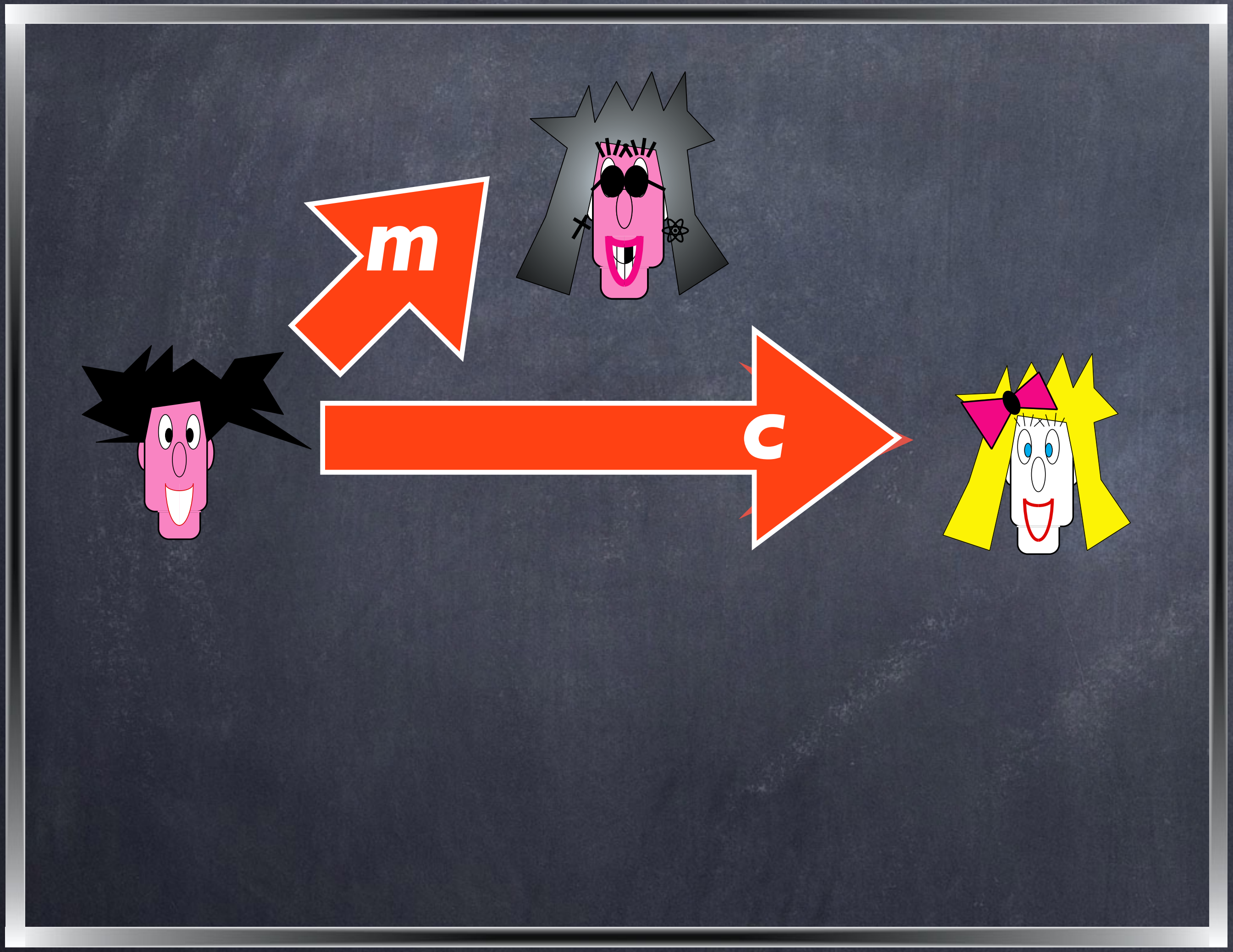


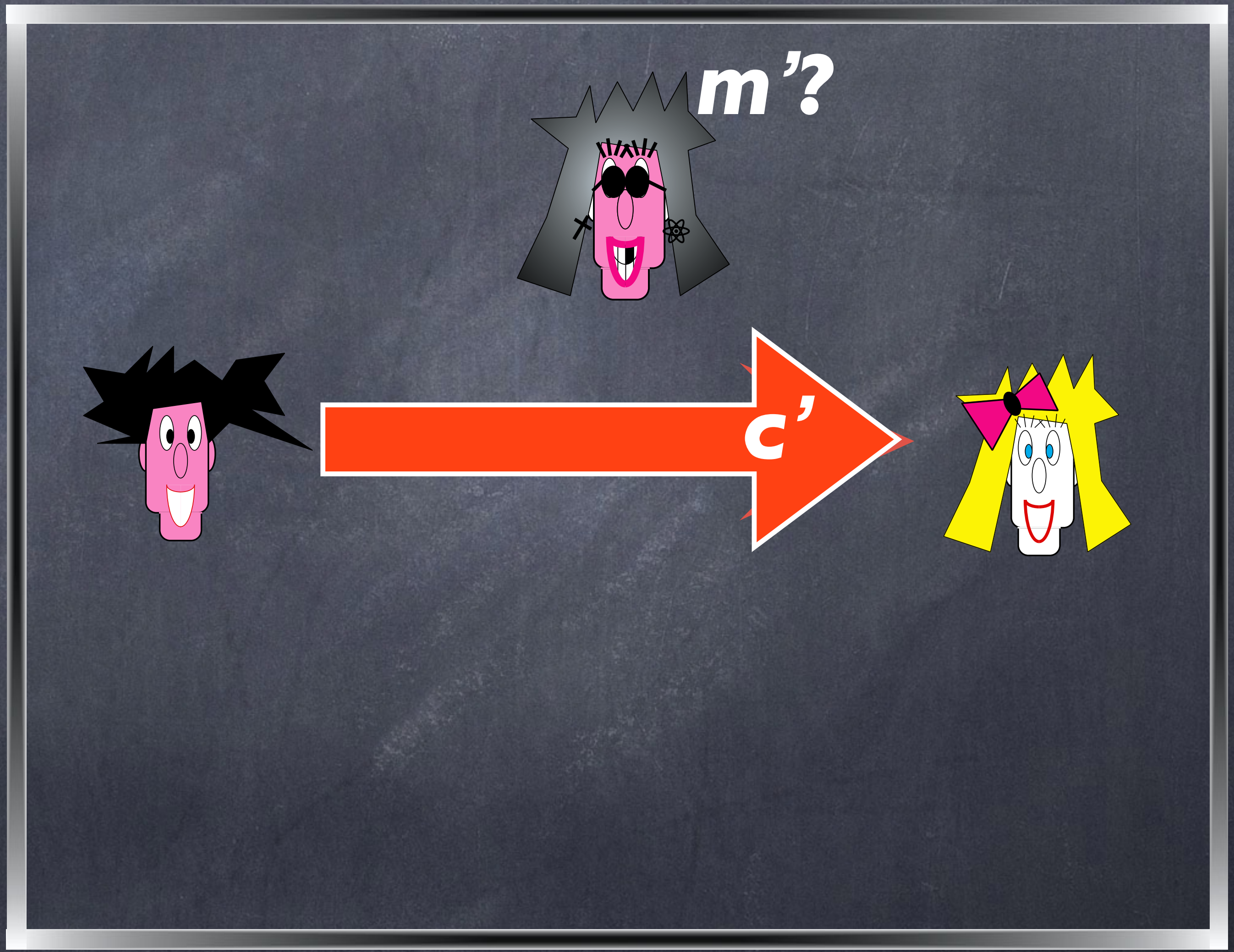
Attack scenarios

- **Ciphertext-only attack:** This is the most basic type of attack and refers to the scenario where the adversary just observes a ciphertext (or multiple ciphertexts) and attempts to determine the underlying plaintext (or plaintexts).

Attack scenarios

- **Known-plaintext attack:**



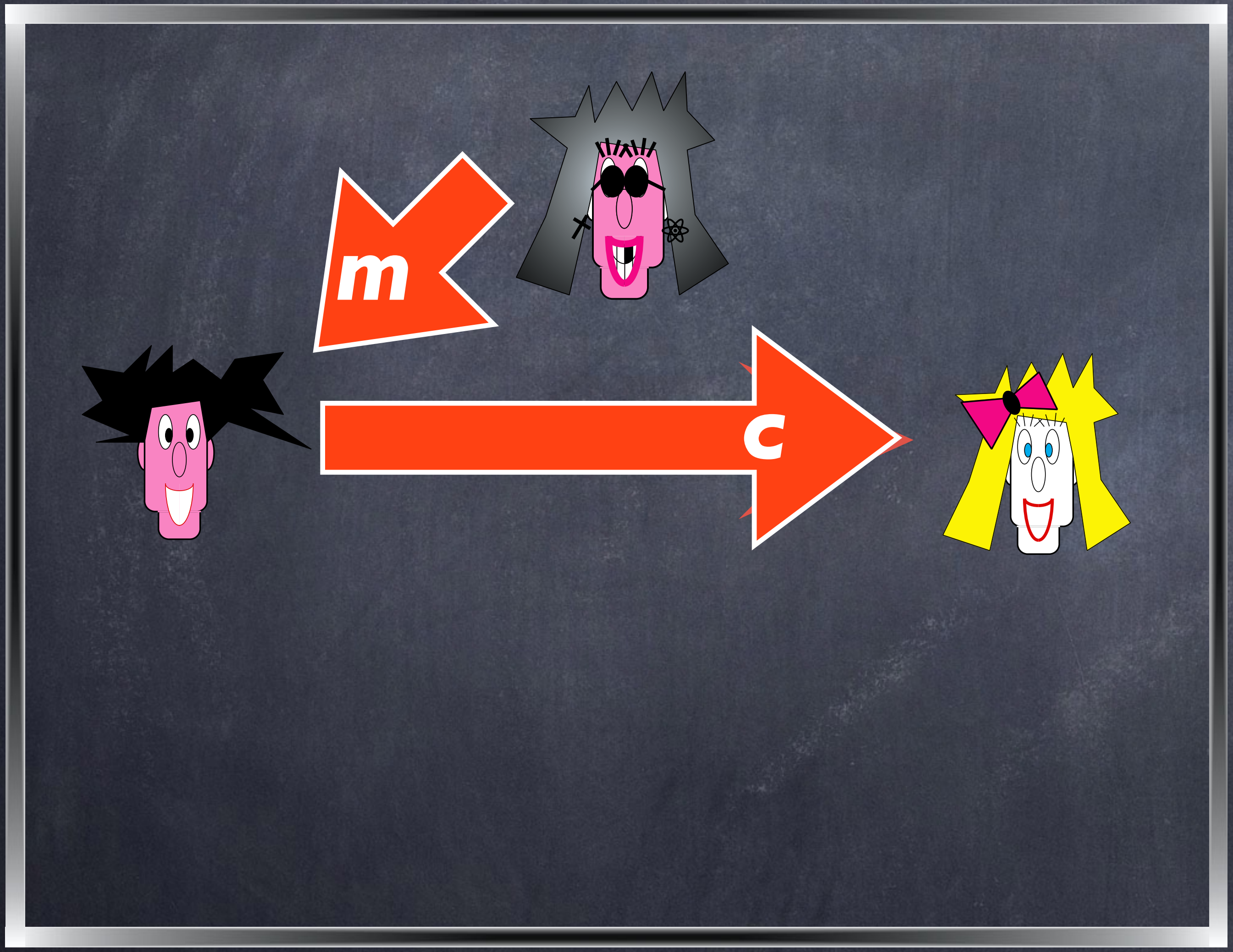


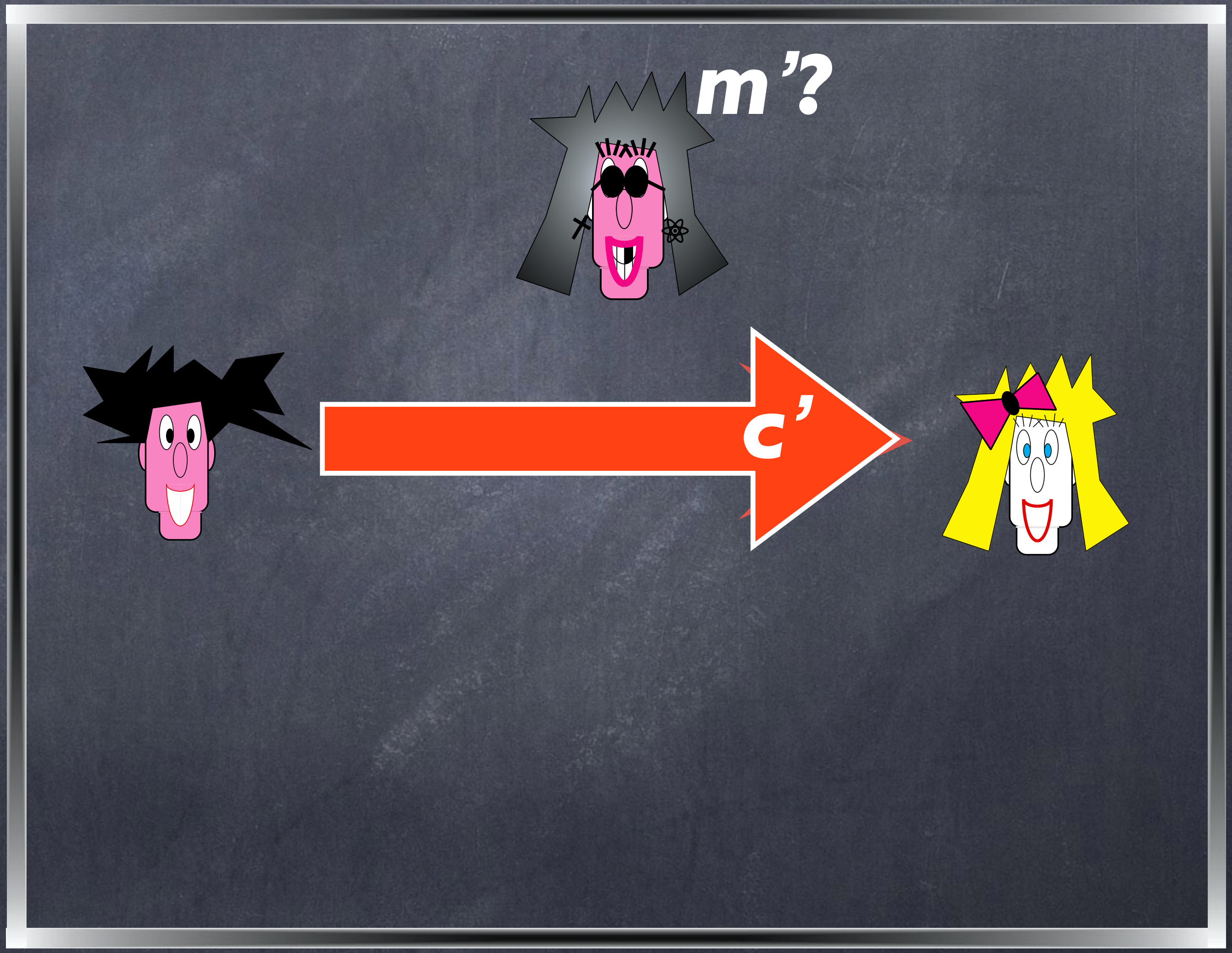
Attack scenarios

- **Known-plaintext attack:** The adversary learns one or more pairs of plaintexts/ciphertexts encrypted under the same key. The aim is to determine the plaintext that was encrypted in some other ciphertext.

Attack scenarios

- **Chosen-plaintext attack:**



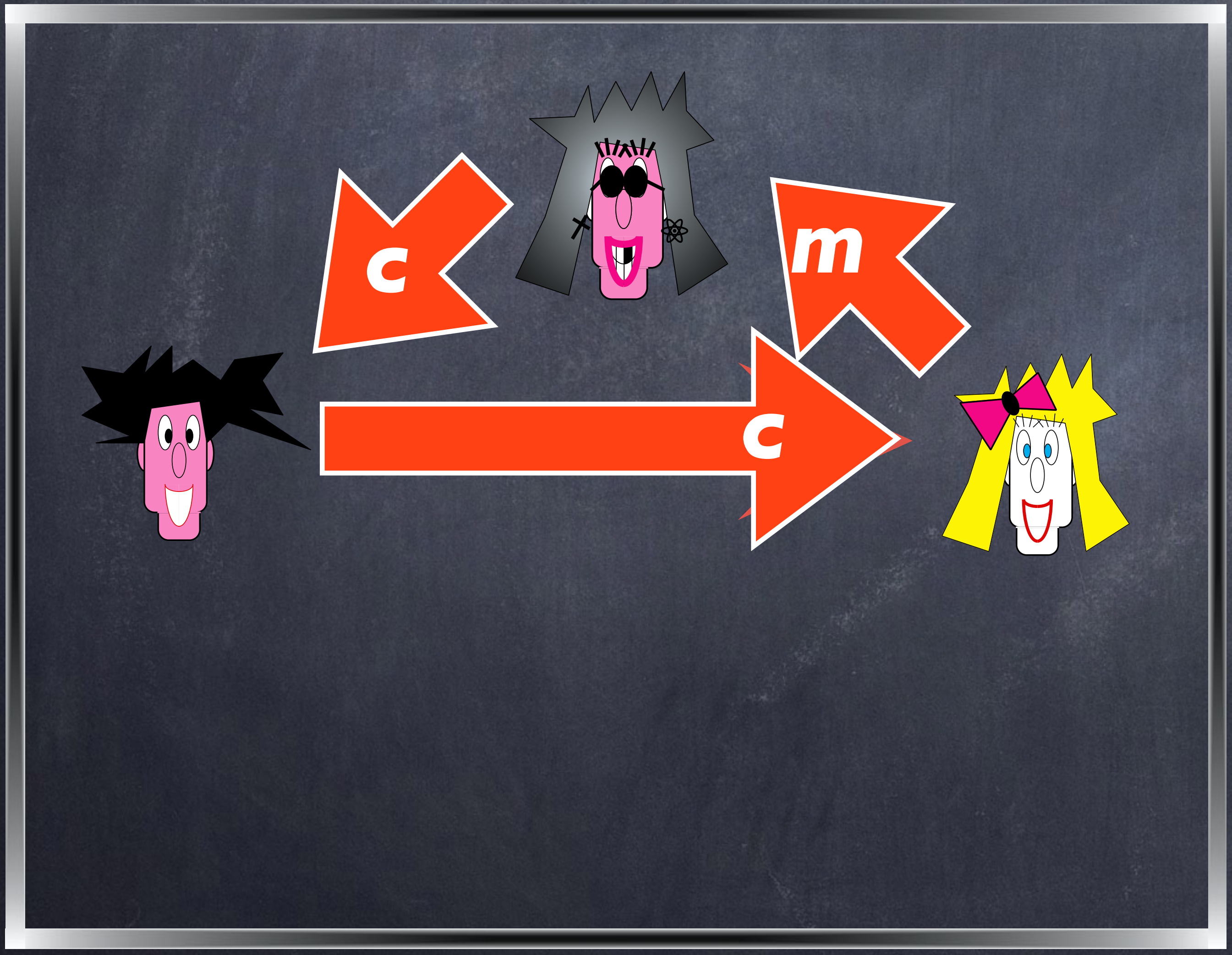


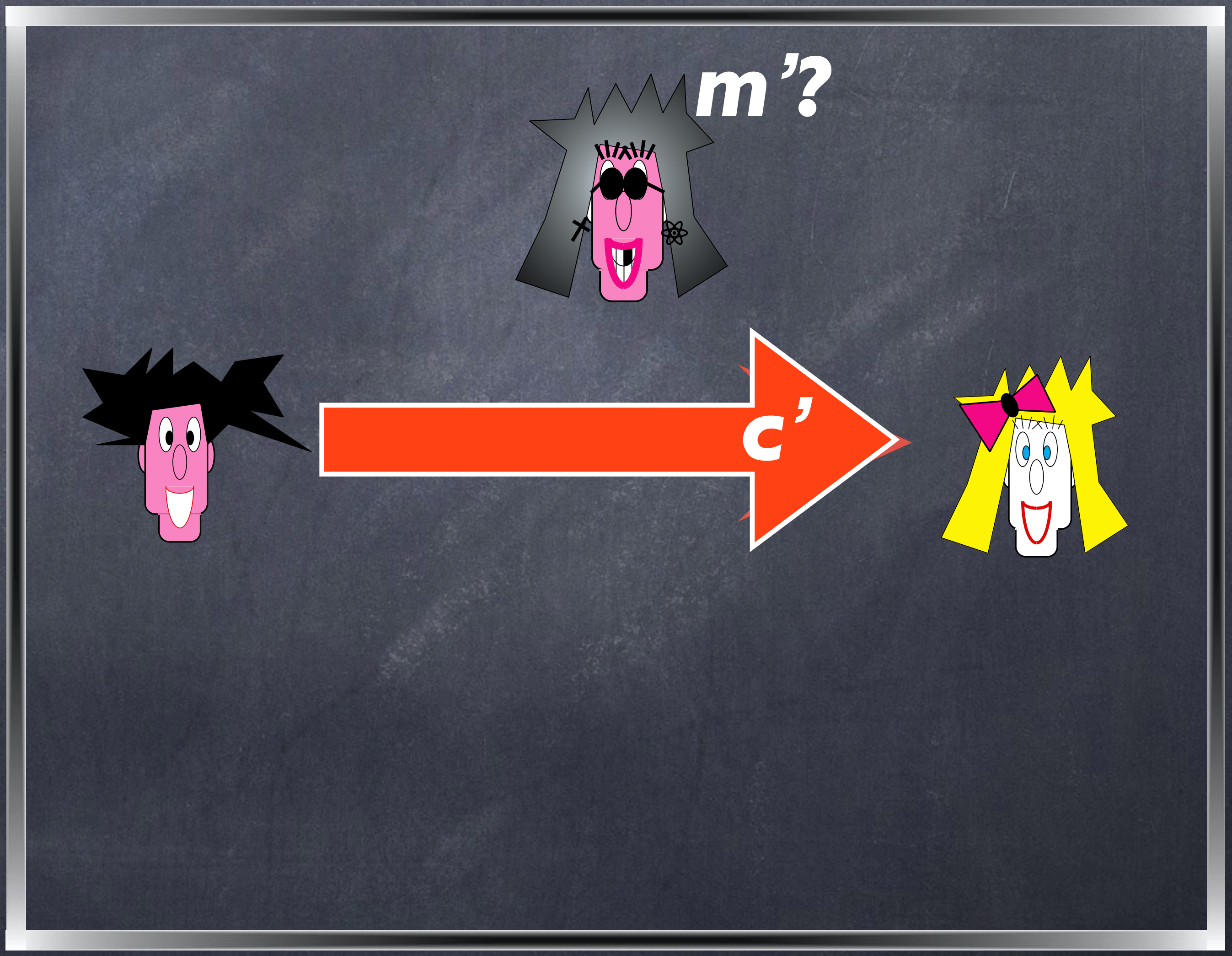
Attack scenarios

- **Chosen-plaintext attack:** The adversary has the ability to obtain the encryption of plaintexts of its choice. It then attempts to determine the plaintext that was encrypted in some other ciphertext.

Attack scenarios

- **Chosen-ciphertext attack:**





Attack scenarios

- **Chosen-ciphertext attack:** The adversary is even given the capability to obtain the decryption of ciphertexts of its choice. The adversary's aim, once again, is to determine the plaintext that was encrypted in some other ciphertext.

INTRODUCTION TO
MODERN
CRYPTOGRAPHY
Second Edition
Jonathan Katz • Yehuda Lindell

Chapter 1 : Introduction

1.3 Historical Ciphers and Their Cryptanalysis

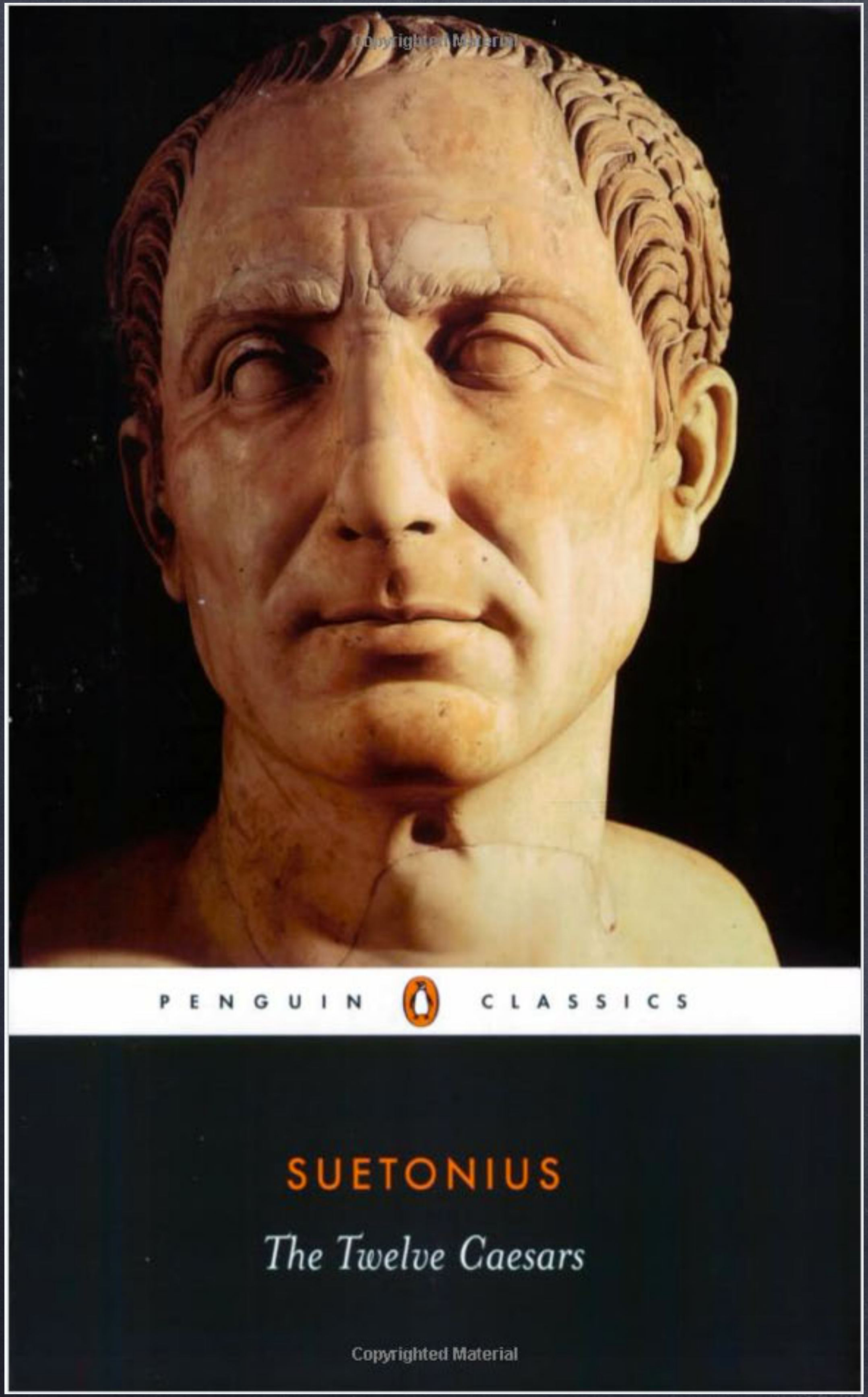
- Cæsar’s cipher.
- The shift cipher and sufficient key space principle.
- Mono-alphabetic substitution.
- An improved attack on the shift cipher.
- The Vigenère (poly-alphabetic shift) cipher.
- Breaking the Vigenère cipher.
- Ciphertext length and cryptanalytic attacks.

1.3 Historical Ciphers and Their Cryptanalysis

- In this section (and this section only), plaintexts are written in lower case and ciphertexts are written in UPPER CASE.
- When describing attacks on schemes, we always apply Kerckhoffs' principle and assume that the scheme is known to the adversary (but the key being used is not).

Cæsar's cipher

LVI. He left memoirs too of his deeds in the Gallic war and in the civil strife with Pompeius; for the author of the Alexandrian, African, and
Hispanic Wars is unknown; some think it was Oppius, others Hirtius, who also supplied the final book of the Gallic War, which Cæsar left
unwritten. With regard to Cæsar's memoirs Cicero, also in the *Brutus* speaks in the following terms: "He wrote memoirs which deserve the
highest praise; they are naked in their simplicity, straightforward stripped of all rhetorical adornment, as of a garment; but while
his purpose was to supply material to others, on v ry might draw, he haply gratified silly folk, who will
try to use the curling-irons on his narrative, but h ching the subject." Of these same memoirs Hirtius
uses this emphatic language: "They are so highly ra seems to have deprived writers of an opportunity,
rather than given them one; yet our admiration fo s; for they know how well and faultlessly he wrote,
while we know besides how easily and rapidly he that they were put together somewhat carelessly
and without strict regard for truth; since in many e the accounts which others gave of their actions,
and gave a perverted account of his own, either c ess; and he thinks that he intended to rewrite and
revise them. He left besides a work in two volum *nti-Catones* ['Against Cato'], in addition to a poem,
entitled *Iter* ['The Journey']. He wrote the first of t d returning to his army from Gallia Citerior, where
he heard lawsuits; the second about the time of th course of a twenty-four days' journey from Rome
to Hispania Ulterior. Some letters of his to the se to have been the first to reduce such documents
to pages and the form of a note-book [*i.e.*, to boo nd generals sent their reports written right across
the sheet [*i.e.*, without columns or margins, but ac on]. There are also letters of his to Cicero, as
well as to his intimates on private affairs, and in the latter, if he had anything confidential to say, he wrote it in cipher, that is,
by so changing the order of the letters of the alphabet, that not a word could be made out. If anyone wishes to decipher
these, and get at their meaning, he must substitute the fourth letter of the alphabet, namely D, for A, and so with the others.
We also have mention of certain writings of his boyhood and early youth, such as the *Laudes Herculis* ["Praises of Hercules"], a tragedy
Oedipus, and a *Dicta Collectanea* ["Collection of Apophthegms"]; but Augustus forbade the publication of all these minor works in a very
brief and frank letter sent to Pompeius Macer, whom he had selected to set his libraries in order.


PENGUIN CLASSICS

SUETONIUS

The Twelve Caesars

Cæsar's cipher

One of the oldest recorded ciphers, known as Cæsar's cipher, is described in

"De Vita Cæsarum, Divus Iulius"

("The Lives of the Cæsars, The Deified Julius"),

written in approximately 110 A.D.

De Vita Cæsarum, Divus Iulius

- There are also letters of his to Cicero, as well as to his intimates on private affairs, and in the latter, if he had anything confidential to say, he wrote it in cipher, that is, by so changing the order of the letters of the alphabet, that not a word could be made out. If anyone wishes to decipher these, and get at their meaning, he must substitute the first letter of the alphabet, namely **a**, for the fourth **D**, and so with the others.

Cæsar's cipher

- Julius Cæsar encrypted by rotating the letters of the alphabet by 3 places: **`a`** was replaced with **`D`**, **`b`** with **`E`**, and so on. Of course, at the end of the alphabet, the letters wrap around and so **`x`** was replaced with **`A`**, **`y`** with **`B`**, and **`z`** with **`C`**.

- For example, "**`begin the attack now`**", with spaces removed, would be encrypted as:

  `EHJLQ WKH DWWDFN QRZ`

  making it unintelligible.

Cæsar's cipher / ROT-13

- **ROT-13** (where the shift is 13) is widely used nowadays in various online forums.
- It is understood that this does not provide any cryptographic security.
- **ROT-13** is used merely to ensure that the text (say, a movie spoiler) is unintelligible unless the reader of a message consciously chooses to decrypt it.

Shift cipher + sufficient key space principle

- The shift cipher is similar to Cæsar’s cipher, but a secret key is introduced.
- The shift cipher the key ***k*** is a number between 0 and 25.
- To **encrypt**, letters are rotated by ***k*** places as in Cæsar’s cipher.
- Algorithm **Gen** outputs a random number ***k*** in the set { 0, . . . , 25 };

Shift cipher + sufficient key space principle

- Algorithm **Enc** takes a key ***k*** and a plaintext written using English letters and shifts each letter of the plaintext forward ***k*** positions (wrapping around from **z** to **a**);
- Algorithm **Dec** takes a key ***k*** and a ciphertext written using English letters and shifts every letter of the ciphertext backward ***k*** positions (wrapping around from **a** to **z**).
- The plaintext message space M is defined to be all finite strings of characters from the English alphabet.

Shift cipher

Using mathematical notation,

- The message space $\mathcal{M}$ is defined to be any finite sequence of integers that lie in the range $\{0, \ldots, 25\}$.
- Encryption of a plaintext character $\boldsymbol{m_i}$ with the key $\boldsymbol{k}$ gives the ciphertext character $[(\boldsymbol{m_i}+\boldsymbol{k}) \bmod 26]$,
- Decryption of a ciphertext character $\boldsymbol{c_i}$ with the key $\boldsymbol{k}$ is defined by $[(\boldsymbol{c_i}-\boldsymbol{k}) \bmod 26]$.

Sufficient key space principle

- Is the shift cipher secure?
- Try to decrypt the following message that was encrypted using the shift cipher and a secret key ***k*** (whose value we will not reveal):

  OVDTHUFWVZZPISLRLFZHYLAOLYL.

- Is it possible to decrypt this message without knowing ***k*** ?

Sufficient key space principle

- It is completely trivial! The reason is that there are only 26 possible keys.
- Easy to try every key, and see which key decrypts the ciphertext into a plaintext that "makes sense".
- Brute-force attack or exhaustive search attack. Clearly, any secure encryption scheme must not be vulnerable to such a brute-force attack; otherwise, it can be completely broken, irrespective of how sophisticated the encryption algorithm is.

Sufficient key space principle

This brings us to a trivial, yet important, principle called the "sufficient key space principle":

"*Any secure encryption scheme must have a key space that is not vulnerable to exhaustive search*".

Sufficient key space principle

- In today's age, an **exhaustive search** may use very powerful computers, or many thousands of PC's that are distributed around the world. Thus, the number of possible keys must be very large (at least 2^{60} or 2^{70}).
- **Necessary** condition for security, but not a **sufficient** one.
- Next we see an encryption scheme that has a very large key space but which is still insecure.

Mono-alphabetic substitution

- The idea behind mono-alphabetic substitution is to map each plaintext character to a different ciphertext character in an arbitrary, but fixed, manner, subject only to the fact that the mapping must be one-to-one in order to enable decryption.
- The key space thus consists of all permutations of the alphabet, meaning that the size of the key space is

$$26! \sim 2^{88}$$

if we are working with the English alphabet.

Mono-alphabetic substitution

- As an example, the key

```
abcdefghijklmnopqrstuvwxyz
XEUADNBKVMROCQFSYHWGLZIJPT
```

  in which **`a`** maps to **`X`**, etc, would encrypt the message **`tellhimaboutme`** to **`GDOOKVCXEFLGCD`**.

- A brute force attack on the key space for this cipher takes much longer than a lifetime, even using the most powerful computer known today.

Mono-alphabetic substitution

- However, this does not necessarily mean that the cipher is secure.

- Assume that English-language text is being encrypted (i.e., the text is grammatically-correct English writing, not just text written using characters of the English alphabet).

- It is then possible to attack the mono-alphabetic substitution cipher by utilizing statistical patterns of the English language (of course, the same attack works for any language).

Mono-alphabetic substitution

The two properties of this cipher that are utilized in the attack are as follows:

1. In this cipher, the mapping of each letter is fixed, and so if **e** is mapped to **D**, then every appearance of **e** in the plaintext will result in the appearance of **D** in the ciphertext.
2. The probability distribution of individual letters in the language is known. That is, the average frequency counts of the different letters are quite invariant over different texts. Of course, the longer the text, the closer the frequency counts will be to the average. However, even relatively short texts (consisting of tens of words) have distributions that are "close enough" to the average.

Mono-alphabetic substitution

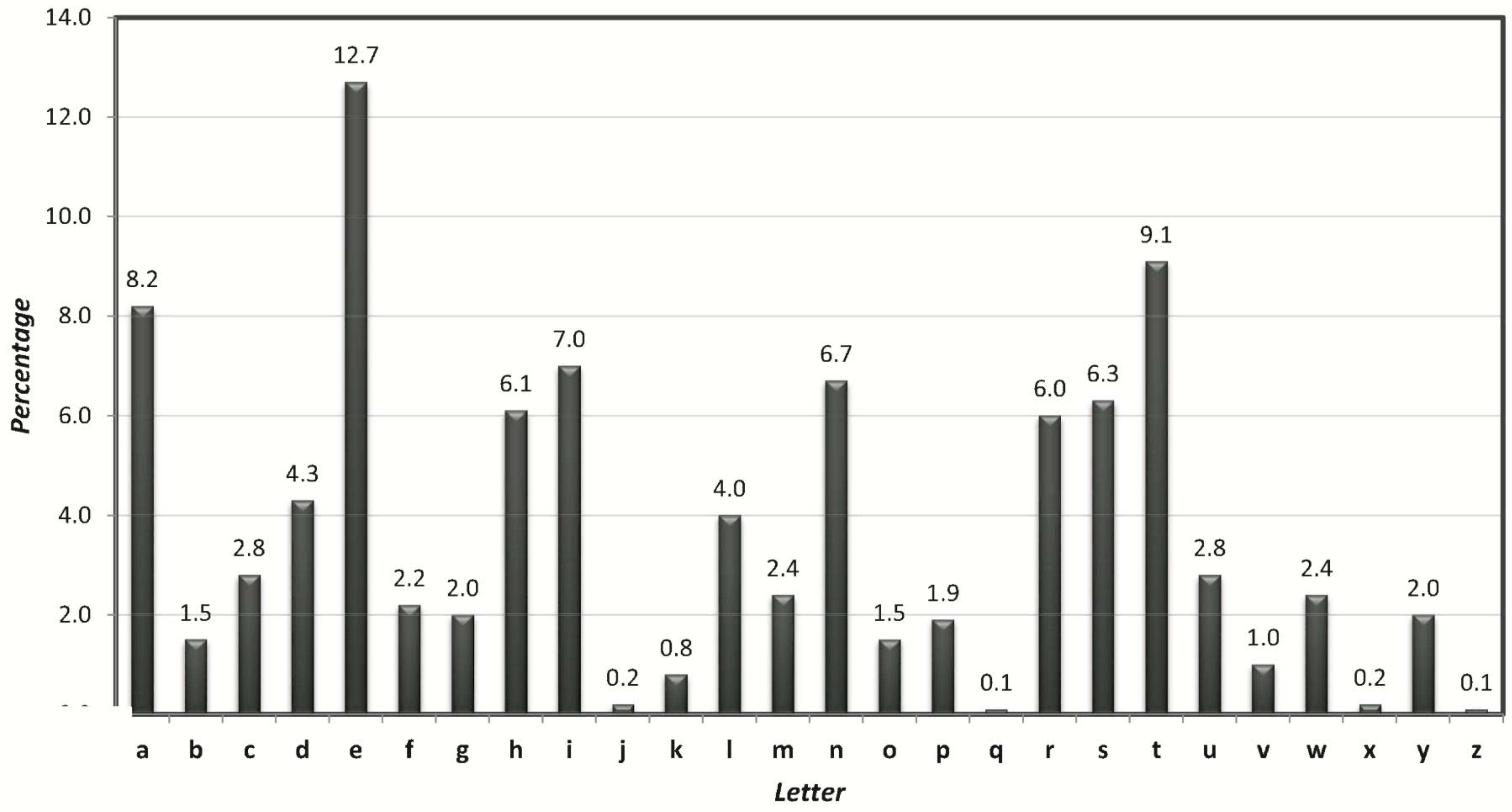


FIGURE 1.3: Average letter frequencies for English-language text.

Mono-alphabetic substitution

- Make an initial guess of the mapping defined by the key based on the frequency counts. For example, since e is the most frequent letter in English, we will guess that the most frequent character in the ciphertext corresponds to the plaintext character e, and so on.
- Unless the ciphertext is very long, some guesses are likely to be wrong.

Mono-alphabetic substitution

Even for quite short ciphertexts the guesses will be good enough to enable relatively quick decryption utilizing other knowledge of the English language

- between `t` and `e`, the character `h` is likely to appear
- `u` generally follows `q`, etc

Mono-alphabetic substitution

- Actually, it should not be very surprising that the mono-alphabetic substitution cipher can be quickly broken, since puzzles based on this cipher appear in newspapers (and are solved by some people before their morning coffee)!
- Although mono-alphabetic cipher has a very large key space, it is still completely insecure.

An improved attack on the shift cipher

- A drawback of brute-force approach is that it is difficult to automate, since it is difficult for a computer to check whether some plaintext “makes sense”.
- We can use character frequency tables to give an improved attack on the shift cipher.

An improved attack on the shift cipher

- Let p_i , for $0 \leq i \leq 25$, denote the probability of the ith letter in normal English text. A simple calculation using known values of p_i gives

$$\sum_{i=0}^{25} p_i^2 \approx 0.065 \,.$$

- Now, say we are given some ciphertext and let q_i denote the probability of the ith letter in this ciphertext (q_i is simply the number of occurrences of the ith letter divided by the length of the ciphertext).

An improved attack on the shift cipher

- If the key is ***k***, then we expect that q_{i+k} should be roughly equal to p_i for every ***i***.
- If we compute for each value of $j \in \{ 0, \ldots, 25 \}$,

$$I_j \overset{\text{def}}{=} \sum_{i=0}^{25} p_i \cdot q_{i+j}$$

then we expect to find that $I_k \approx 6.5\%$ where ***k*** is the key that is actually being used (whereas I_j for $j \neq k$ is expected to be smaller).

2.96%
3.19%
3.39%
3.40%
3.43%
3.60%
3.76%
3.90%
3.93%
3.95%
4.13%
4.52%
4.53%
6.54%

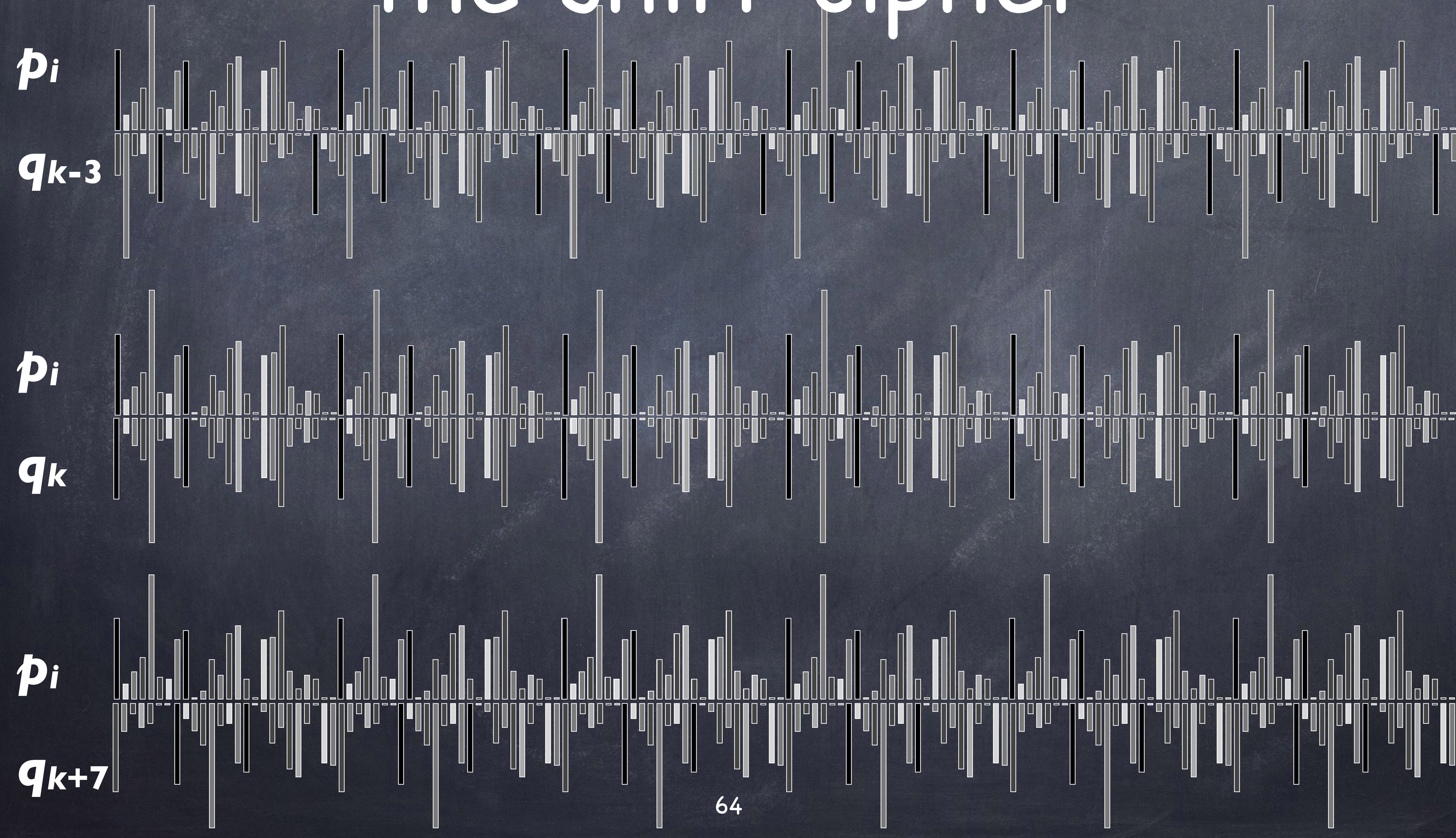
An improved attack on the shift cipher

- Key-recovery attack that is easy to automate: compute I_j for all j, and then output the value k for which I_k is closest to 6.5%.

p_i	**abcdefghijklmnopqrstuvwxyz**
...	
q_{k-1}	**zabcdefghijklmnopqrstuvwxy**
q_k	**abcdefghijklmnopqrstuvwxyz**
q_{k+1}	**bcdefghijklmnopqrstuvwxyza**
...	

2.96%
3.19%
3.39%
3.40%
3.43%
3.60%
3.76%
3.90%
3.93%
3.95%
4.13%
4.52%
4.53%
6.54%

An improved attack on the shift cipher



The Vigenère cipher

The Vigenère cipher (poly-alphabetic shift)

- Attacks on mono-alphabetic ciphers can be thwarted by mapping different instances of the same plaintext character to different ciphertext characters.
- This has the effect of “smoothing out” the probability distribution of characters in the ciphertext.

The Vigenère cipher

- For example, consider the case that `e` is sometimes mapped to **`G`**, sometimes to **`P`**, and sometimes to **`Y`**.
- The ciphertext letters **`G`**, **`P`**, and **`Y`** do not stand out as more frequent, because less frequent characters are also be mapped to them.
- Character frequencies do offer much information about the mapping.

The Vigenère cipher

- The Vigenère cipher works by applying multiple shift ciphers in sequence.
- A short, secret word is chosen as the key, and then the plaintext is encrypted by "adding" each plaintext character to the next character of the key (as in the shift cipher), wrapping around in the key when necessary.

The Vigenère cipher

- For example, an encryption of the message `tellhimaboutme` using the key `cafe` :

Plaintext:	**`tellhimaboutme`**
Key:	**`cafecafecafeca`**
Ciphertext:	**`VEQPJIREDOZXOE`**

- Notice that **`l`** is mapped once to **`Q`** and once to **`P`**. The ciphertext character **`E`** is sometimes obtained from **`e`** and sometimes from **`a`**.

The Vigenère cipher

- If the key is a sufficiently-long word (chosen at random), then cracking this cipher seems to be a daunting task.
- Indeed, it was considered by many to be an unbreakable cipher.
- Although it was invented in the 16th century, a systematic attack on the scheme was only devised hundreds of years later.

Breaking the Vigenère cipher.

- If the length of the key is known, then the task is relatively easy.
- Specifically, say the length of the key is ***t*** (this is sometimes called the period). Then the ciphertext can be divided into ***t*** parts where each part can be viewed as being encrypted using a single instance of the shift cipher.

Breaking the Vigenère cipher.

- Let $\boldsymbol{k = k_1, \ldots, k_t}$ be the key
(each $\boldsymbol{k_i}$ is a letter of the alphabet)

- let $\boldsymbol{c_1, c_2, \ldots}$ be the ciphertext characters.

- For every $\boldsymbol{j}$ ($\boldsymbol{1 \leq j \leq t}$) the set of characters

$$\boldsymbol{c_j, c_{j+t}, c_{j+2t}, \ldots}$$

were encrypted by a shift cipher using key $\boldsymbol{k_j}$.

Breaking the Vigenère cipher.

- All that remains is therefore to determine, for each ***j***, which of the 26 possible keys is the correct one.

- This is not as trivial as in the case of the shift cipher, because by guessing a single letter of the key it is no longer possible to determine if the decryption "makes sense".

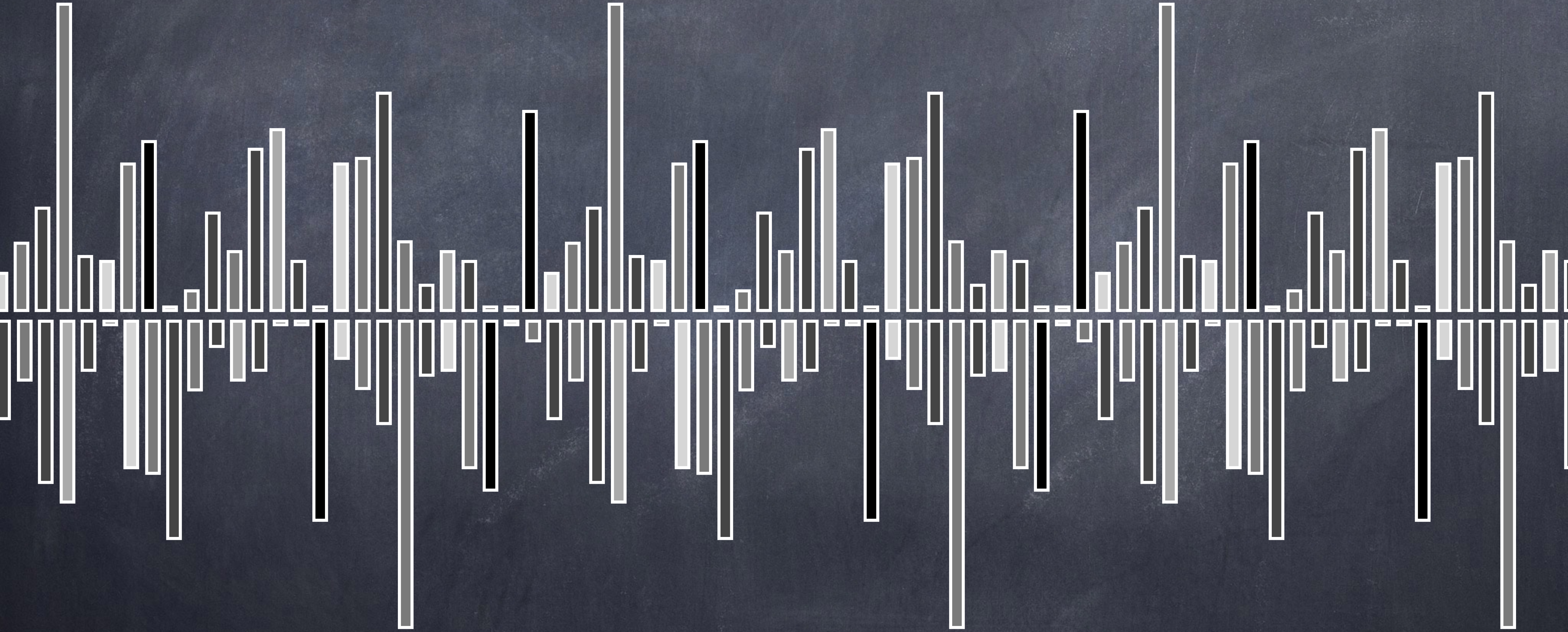
Breaking the Vigenère cipher.

- Furthermore, checking for all values of ***j*** simultaneously would require a brute force search through 26^t different possible keys (which is infeasible for ***t*** greater than, say, 15).
- We can still use the statistical method described earlier.

Breaking the Vigenère cipher.

- For every set of ciphertext characters relating to a given key, it is possible to tabulate the frequency of each ciphertext character and then check which of the 26 possible shifts yields the "right" distribution.
- This can be done separately for each key, the attack can be carried out very quickly; all that is required is to build ***t*** frequency tables and compare them to the real distribution.

Breaking the Vigenère cipher.

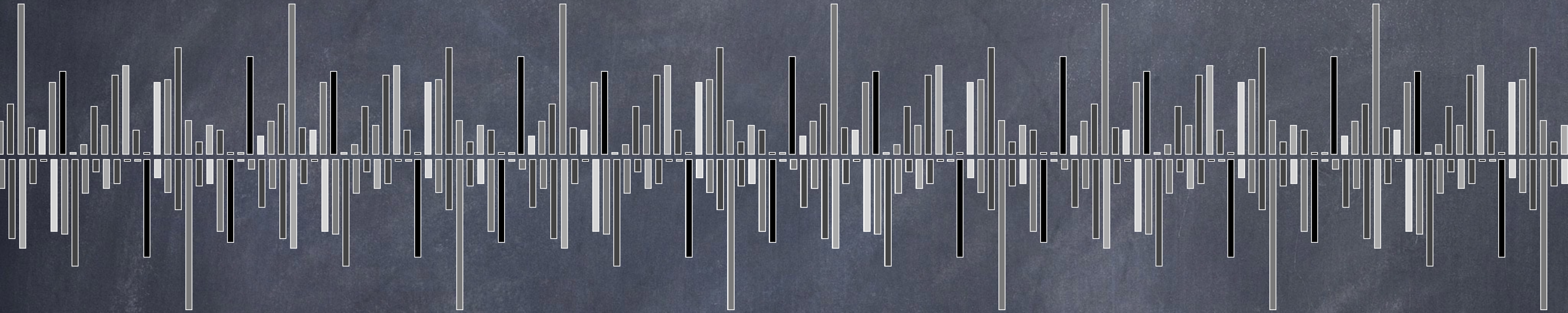


Breaking the Vigenère cipher.

- An alternate, somewhat easier approach, is to use the improved method for attacking the shift cipher that we showed earlier.
- Attack does not rely on checking for a plaintext that "makes sense", but only relies on the underlying probability distribution of characters in the plaintext.

2.96%
3.19%
3.39%
3.40%
3.43%
3.60%
3.76%
3.90%
3.93%
3.95%
4.13%
4.52%
4.53%
6.54%

Breaking the Vigenère cipher.



- Either of the approaches give successful attacks when the key length is known.

2.96%	3.19%	3.39%	3.40%
3.43%	3.60%	3.76%	3.90%
3.93%	3.95%	4.13%	4.52%
	4.53%	6.54%	

Kasiski/Babbage's method

It was first published by Friedrich Kasiski in 1863, but seems to have been discovered by Charles Babbage as early as 1846.

Kasiski's method

- It remains to show how to determine the length of the key.
- Kasiski's method, published in the mid-19th century, gives one approach for solving this problem.
- The first step is to identify repeated patterns of length 2 or 3 in the ciphertext.

Kasiski's method

These are likely to be due to certain bigrams or trigrams that appear very often in English.

- Consider the word "`the`" that appears very often in English text. Clearly, "`the`" will be mapped to different ciphertext characters, depending on its position in the text.
- But in a long text, there is a good chance that "`the`" will be mapped repeatedly to the same ciphertext characters.

Kasiski's method

- If it appears twice in the same relative position, then it will be mapped to the same ciphertext characters.
  - In positions $\boldsymbol{t + j}$ and $\boldsymbol{2t + i}$ $\boldsymbol{(i \neq j)}$ then it will be mapped to different characters.
  - In positions $\boldsymbol{t + j}$ and $\boldsymbol{2t + j}$, then it will be mapped to the same ciphertext characters.

the period is 5 and the distance between the two appearances of NII is 40, which is 8 times the period.

- Consider the following concrete example with the key beads:

```
M: the man and the woman retrieved the letter from the post office
K: bea dsb ead sbe adsbe adsbeadsb ean sdeads bead sbe adsb eadbea
C: ULE PSO ENG NII WREBR RHLSMEYWE XHR DHXTHJ GVOP NII PRKU SFIJGE
```

- The word `the` is mapped to **`ULE`**, to **`NII`** and to **`XHR`**. It is mapped twice to **`NII`**, and in a long enough text it is likely that it would be mapped multiple times to each of the possibilities.

Kasiski’s method

- The main observation of Kasiski is that the distance between such multiple appearances (except for some coincidental ones) is a multiple of the period length.
- Therefore, the greatest common divisor of all the distances between the repeated sequences should yield the period **t** or a multiple thereof.

Kasiski's method

- If the key-length is ***t***, then the ciphertext characters

  $$c_1, c_{1+t}, c_{1+2t}, \ldots$$

  are encrypted using the same shift.

- The frequencies of the characters in this sequence are expected to be identical to the character frequencies of standard English text except in some shifted order.

index of coincidence method

- Let q_i denote the frequency of the ith English letter in the previous sequence.
- If the shift used here is k_1, we expect q_{i+k_1} to be roughly equal to p_i for all i, where p_i is the frequency of the ith letter in English.
- This means that the sequence $p_0, \dots, p_{25}$ is just the sequence $q_0, \dots, q_{25}$ shifted by k_1 places. As a consequence, we expect that

$$\sum_{i=0}^{25} q_i^2 = \sum_{i=0}^{25} p_i^2 \approx 0.065 .$$

index of coincidence method

- For $\tau = 1, 2, \ldots,$ look at the sequence of ciphertext characters $c_1, c_{1+\tau}, c_{1+2\tau}, \ldots$ and tabulate $q_0, \ldots, q_{25}$ for this sequence.

  Then compute

$$S_\tau \overset{\text{def}}{=} \sum_{i=0}^{25} q_i^2 .$$

  When $\tau = t$ we expect to see $S_\tau \approx 0.065$ as discussed earlier.

index of coincidence method

- On the other hand, for **$\tau \neq t$** we expect that all characters will occur with roughly equal probability in the sequence **c_1, $c_{1+\tau}$, $c_{1+2\tau}$, ...,** and so we expect **$q_i \approx {}^1/_{26}$** for all **i**. In this case we will obta

$$S_\tau \approx \sum_{i=0}^{25} \left(\frac{1}{26}\right)^2 \approx 0.038,$$

which is sufficiently different from 0.065 for this technique to work.

Ciphertext length and cryptanalytic attacks.

- The above attacks on the Vigenère cipher require a longer ciphertext than for previous schemes.
- For example, a large ciphertext is needed for determining the period if Kasiski’s method is used.

Ciphertext length and cryptanalytic attacks.

- Statistics are needed for ***t*** different parts of the ciphertext, and the frequency table of a message converges to the average as its length grows (and so the ciphertext needs to be approximately ***t*** times longer than in the case of the shift cipher).

This phenomenon is not coincidental, and relates to the size of the key space for each encryption scheme.

INTRODUCTION TO
MODERN
CRYPTOGRAPHY
Second Edition
Jonathan Katz •Yehuda Lindell

Chapter 1 : Introduction

1.4 The Basic Principles of Modern Cryptography

Principle 1 — the first step in solving any cryptographic problem is the formulation of a rigorous and precise definition of security.

1.4 The Basic Principles of Modern Cryptography

Principle 2 — when the security of a cryptographic construction relies on an unproven assumption, this assumption must be precisely stated.

Furthermore, the assumption should be as minimal as possible.

1.4 The Basic Principles of Modern Cryptography

Principle 3 — cryptographic constructions should be accompanied by a rigorous proof of security with respect to a definition formulated according to **Principle 1**, and relative to an assumption (if any) stated as in **Principle 2**.

<u>1.4.1 Principle 1</u> – Formulation of Exact Definitions

<u>Importance for design:</u>

- enables us to better direct our design efforts
- evaluate the quality of what we build
- define what is needed first rather than to come up with a post facto definition
- achieves more than is needed (less efficient)

Formulation of Exact Definitions

Importance for usage:

encryption scheme within some larger system

- suffices for our application?
- define the security we desire in our system
- look for a scheme satisfying this definition
- a weaker notion of security may suffice

Formulation of Exact Definitions

Importance for study:

how can we compare encryption schemes?

- efficiency
- level of security
- trade-off

An example: secure encryption.

Answer 1 — an encryption scheme is secure if no adversary can find the *secret key* when given a ciphertext.

An example: secure encryption.

Answer 2 — an encryption scheme is secure if no adversary can find the *plaintext* that corresponds to the ciphertext.

An example: secure encryption.

Answer 3 — an encryption scheme is secure if no adversary can determine *any character* of the plaintext that corresponds to the ciphertext.

An example: secure encryption.

Answer 4 — an encryption scheme is secure if no adversary can derive any *meaningful information* about the plaintext from the ciphertext.

Definitions of security should suffice for all potential applications.

An example: secure encryption.

The Final Answer — an encryption scheme is secure if no adversary can effectively compute any *function* of the plaintext from the ciphertext.

Formulation of Exact Definitions

In order to fully define security of some cryptographic task, there are two distinct issues that must be explicitly addressed :

- what is considered to be a *break*,
- what is assumed regarding the *power of the adversary*.

Formulation of Exact Definitions

Any definition of security will take the following general form:

-- A cryptographic scheme for a given task is secure if no adversary of a specified power can achieve a specified break. --

1.4.2 Principle 2 – Reliance on Precise Assumptions

- Most modern cryptographic constructions cannot be proven secure unconditionally.
- Indeed, proofs of this sort would require resolving questions in the theory of computational complexity that seem far from being answered today.

Reliance on Precise Assumptions

1. *Validation of the assumption:*

- Assumptions are statements that are not proven but are rather conjectured to be true.
- In order to strengthen our belief in some assumption, it is necessary for the assumption to be studied.

Reliance on Precise Assumptions

- The more the assumption is examined and tested without being refuted, the more confident we are that the assumption is true.
- Study of an assumption can provide positive evidence of its validity by showing that it is implied by another, widely believed, assumption.
- If the assumption being relied upon is not precisely stated and presented, it cannot be studied and (potentially) refuted.

Reliance on Precise Assumptions

2. *Comparison of schemes:*

- Presented with two schemes that can both be proven to satisfy some definition but each with respect to a different assumption, assuming both schemes are equally efficient, which scheme should be preferred?

Reliance on Precise Assumptions

- If the assumption on which one scheme is based is *weaker* than the assumption on which the second scheme is based (i.e., the second assumption implies the first), then the first scheme is to be preferred .
- If the assumptions used by the two schemes are incomparable, then the general rule is to prefer the scheme that is based on the better-studied assumption, or the assumption that is simpler.

Reliance on Precise Assumptions

3. Facilitation of proofs of security:

Modern cryptographic constructions are presented together with proofs of security.

If the security of the scheme cannot be proven unconditionally and must rely on some assumption, then a proof that

"a construction is secure if an assumption is true"

can only be provided if there is a precise statement of what the assumption is.

Reliance on Precise Assumptions

- One observation is that it is always possible to just assume that a construction *itself* is secure.
- Of course, this is not accepted practice in cryptography for a number of reasons :

Reliance on Precise Assumptions

1. An assumption that has been tested over the years is preferable to a new assumption that is introduced just to prove a given construction secure.

Reliance on Precise Assumptions

2. There is a general preference for assumptions that are simpler to state, since they are easier to study and to refute.

   For example, an assumption of the type that some mathematical problem is hard to solve is simpler to study and work with than an assumption that an encryption scheme satisfies a complex definition.

Reliance on Precise Assumptions

3. These low-level assumptions can typically be shared amongst a number of constructions.

   If a specific instantiation of the assumption turns out to be false, it can simply be replaced by a *different* instantiation of that assumption.

1.4.3 Principle 3 – Rigorous Proofs of Security

- Modern crypto stresses the importance of proofs of security for proposed schemes.
- The fact that exact definitions and precise assumptions are used means that such a proof of security is possible.
- However, why is a proof necessary? The main reason is that the security of a construction or protocol cannot be checked in the same way that software is typically checked.

Rigorous Proofs of Security

- For example, the fact that the ciphertext looks garbled, does not mean that a sophisticated adversary is unable to break the scheme.
- Without a *proof* that no specified adversary can break the scheme, we are left only with our intuition that this is the case.
- Experience has shown that intuition in cryptography and computer security is disastrous.

Rigorous Proofs of Security

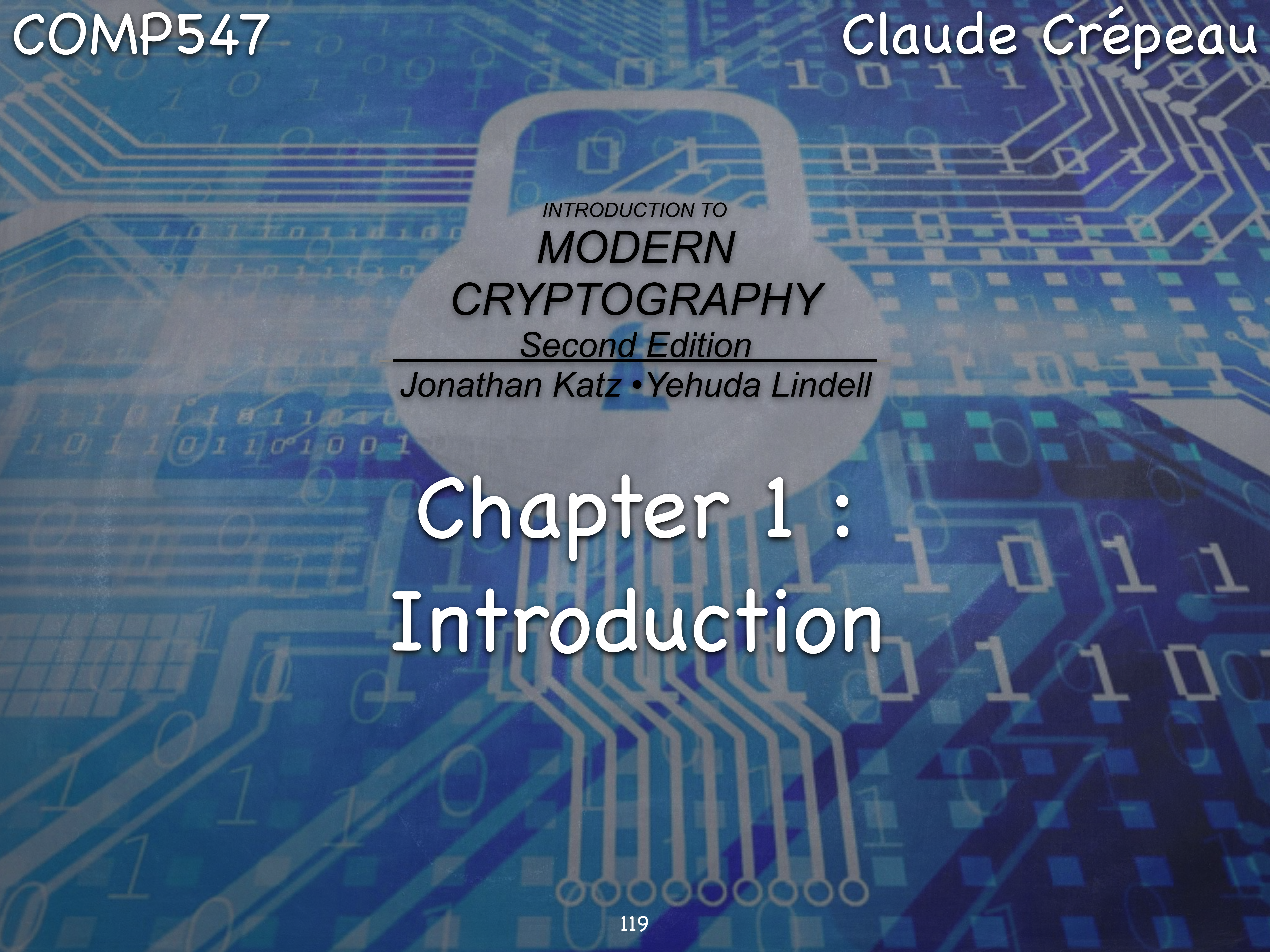
- Although software bugs can sometimes be very costly, the potential damage that may result from someone breaking the security mechanisms of a bank is huge.
- Although many bugs exist in software, things basically work due to the fact that typical users do not try to make their software fail. In contrast, attackers use amazingly complex and intricate means to attack security mechanisms with the clear aim of breaking them.

The reductionist approach.

Given a theorem of the form

"Given that Assumption X is true, Construction Y is secure according to the given definition",

a proof typically shows how to *reduce* the problem given by Assumption X to the problem of breaking Construction Y. More to the point, the proof will typically show (via a constructive argument) how any adversary breaking Construction Y can be used as a sub-routine to violate Assumption X.



INTRODUCTION TO

MODERN CRYPTOGRAPHY

Second Edition

Jonathan Katz •Yehuda Lindell

Chapter 1 : Introduction