Cryptography
CS 555

Week 1:

- Course Overview & What is Cryptography
- Historical Ciphers (& How to Break Them)
- Perfect Secrecy
- Computational Security

Readings: Katz and Lindell Chapter 1-2 + Appendix A.3 (background)

Spring 2021

Course Resources

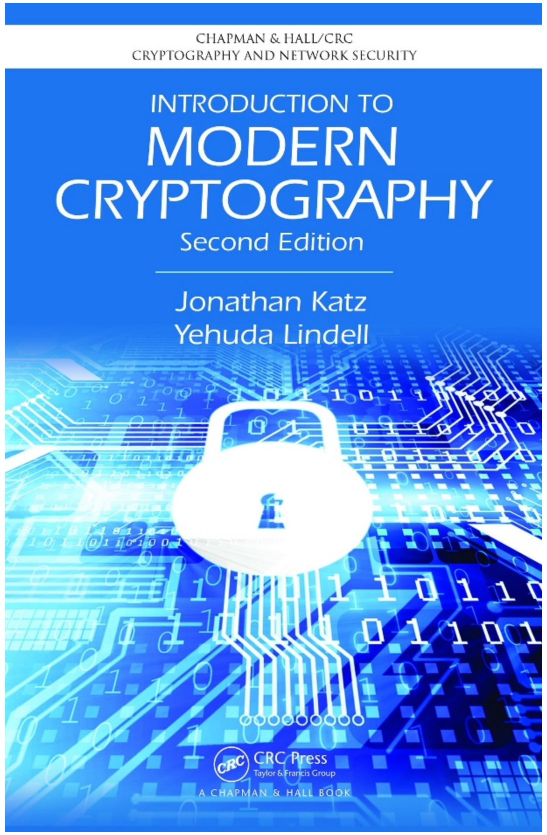
Instructor: Jeremiah Blocki

Office Hours: Thursdays from 2-4PM

TA: Mohammad Hassan Ameri

Office Hours: TBD

Course Web Page: Slides, homeworks and schedule
https://www.cs.purdue.edu/homes/jblocki/courses/555_Spring21/



Technology

- **Brightspace**
  - Syllabus (You are responsible for reading and understanding course policies)
  - Recorded Lectures
  - Quizzes
- **Gradescope**
  - Submit homework assignments
  - View Graded Assignments and Exams
- **Piazza**
  - Course Discussion Board
  - Announcements/Questions
  - Preferred method of communication

Grades

- Course Participation: 5%
- Homework: 35%
- Midterm Exam: 20%
- Final Exam: 25%
- Online Quizzes: 15%

Collaboration is permitted on homework assignments, but you completely understand your solutions and you must write the solutions entirely in your own words.

No collaboration on quizzes/exams

Expected Background

- Basic Probability Theory
- Algorithms and Complexity
    - Most security proofs involve reductions
- General Mathematical Maturity
    - Quantifiers/Predicate Logic
    - Understand what is (is not) a proper definition
    - Know how to write a proof

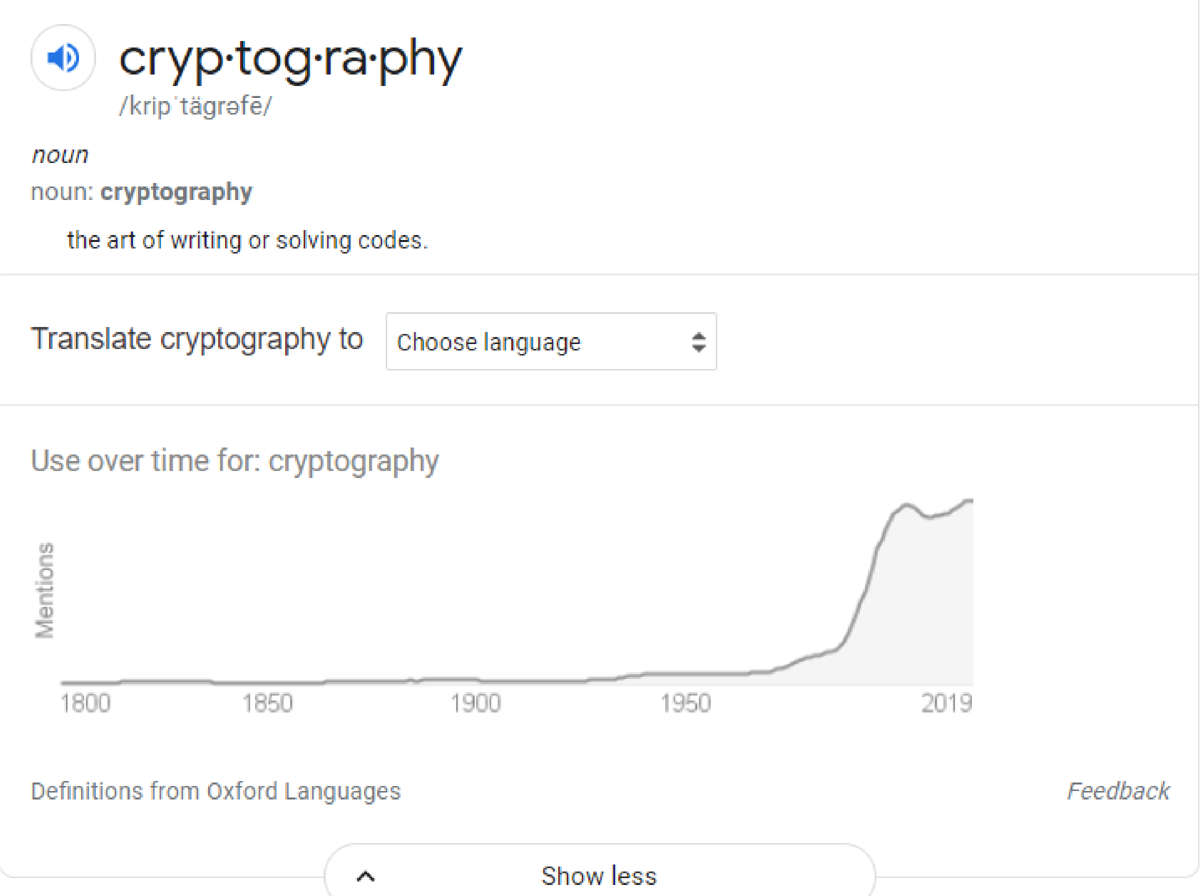
Course Goals

- Understand the mathematics underlying cryptographic algorithms and protocols
- Understand the power (and limitations) of common cryptographic tools
- Understand the formal approach to security in modern cryptography

Topic 1: Course Overview & What is Cryptography

What is Cryptography?

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



cryp·tog·ra·phy

/kripˈtägrəfē/

noun

noun: **cryptography**

the art of writing or solving codes.

Translate cryptography to Choose language

Use over time for: cryptography

Definitions from Oxford Languages

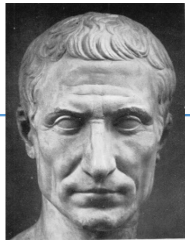
Feedback

What is Cryptography?

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

“The study of mathematical techniques for *securing digital information*, systems and distributed computation against adversarial attacks.”

-- Intro to Modern Cryptography



Art

Late 20th century

Science

What is Cryptography?

- Precise Mathematical Security Definitions
- Specific Algorithmic Assumptions
- Formal Security Reductions/Proofs

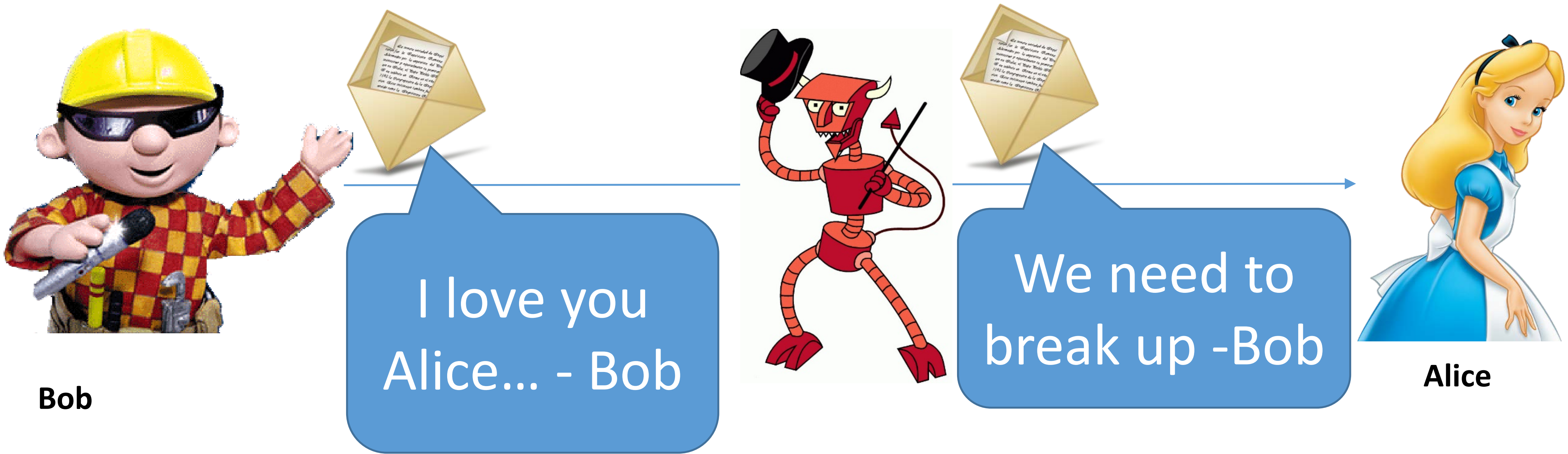
- Experience
- Intuition
- Creativity

What Does It Mean to "Secure Information"

- Confidentiality (Security/Privacy)
  - Only intended recipient can see the communication

What Does It Mean to “Secure Information”

- Confidentiality (Security/Privacy)
  - Only intended recipient can see the communication
- Integrity (Authenticity)
  - The message was actually sent by the alleged sender



Bob

I love you Alice… - Bob

We need to break up -Bob

Alice

Two Attacker Models

- Passive Attacker (Eve)
  - Attacker can eavesdrop
  - Protection Requires?
    - Confidentiality

- Active Attacker (Mallory)
  - Has full control over communication channel
  - Protection Requires?
    - Confidentiality & Integrity

Steganography vs Cryptography

- Steganography
  - Goal: Hide existence of a message
    - Invisible Ink, Tattoo Underneath Hair, …

  - Assumption: Method is secret

Steganography vs Cryptography

- Steganography
  - **Goal:** Hide existence of a message
    - Invisible Ink, Tattoo Underneath Hair, …
  - **Assumption:** Method is secret
- Cryptography
  - **Goal:** Hide the meaning of a message
  - Depends only on secrecy of a (short) key
  - **Kerckhoff's Principle:** Cipher method should not be required to be secret.

FOR ADDED SECURITY, AFTER WE ENCRYPT THE DATA STREAM, WE SEND IT THROUGH OUR NAVAJO CODE TALKER.
ÁLA'IH, DO'NEHLINI, ÁLA'IH, DO'NEHLINI, ÁLA'IH, DO'NEHLINI, DO'NEHLINI, DO'NEHLINI, DO'NEHLINI, ÁLA'IH, DO'NEHLINI, DO'NEHLINI, ÁLAIH, ÁLA'IH, DO'NEH'LINI, . . .
... IS HE JUST USING NAVAJO WORDS FOR "ZERO" AND "ONE"?
WHOA, HEY, KEEP YOUR VOICE DOWN!

Symmetric Key Encryption

- What cryptography has historically been all about (Pre 1970)
- Two parties (sender and receiver) share secret key

- Sender uses key to encrypt (“scramble”) the message before transmission
- Receiver uses the key to decrypt (“unscramble”) and recover the original message

Encryption: Basic Terminology

- Plaintext
  - The original message m
- Plaintext Space (Message Space)
  - The set $\mathcal{M}$ of all possible plaintext messages
  - Example 1: $\mathcal{M} = \{ \ 'attack','retreat', \ 'hold\ current\ position'\}$
  - Example 2: $\mathcal{M} = \{0,1\}^n$ --- all n-bit messages
- Ciphertext $\mathrm{c} \in \mathcal{C}$
  - An encrypted (“scrambled”) message c ∈ $\mathcal{C}$ (ciphertext space)
- Key/Keyspace $\mathrm{k} \in \mathcal{K}$

Private Key Encryption Syntax

- Message Space: $\mathcal{M}$
- Key Space: $\mathcal{K}$
- Three Algorithms $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$
  - $\mathrm{Gen}(R)$ (Key-generation algorithm)
    - **Input:** Random Bits R
    - **Output:** Secret key $\mathrm{k} \in \mathcal{K}$
  - $\mathrm{Enc}_{\mathrm{k}}(m)$ (Encryption algorithm)
    - **Input:** Secret key $\mathrm{k} \in \mathcal{K}$ and message $\mathrm{m} \in \mathcal{M}$
    - **Output:** ciphertext *c*
  - $\mathrm{Dec}_{\mathrm{k}}(c)$ (Decryption algorithm)
    - **Input:** Secret key $\mathrm{k} \in \mathcal{K}$ and a ciphertex c
    - **Output:** a plaintext message $\mathrm{m} \in \mathcal{M}$
- **Invariant:** $\mathrm{Dec}_k(\mathrm{Enc}_k(\mathrm{m}))=\mathrm{m}$

Typically picks $\mathrm{k} \in \mathcal{K}$ uniformly at random

Trusted Parties (e.g., Alice and Bob) must run Gen in advance to obtain secret k.

Assumption: Adversary does not get to see output of Gen

Example: Shift Cipher

- Key Space: $\mathcal{K}$={0,1,…,25}
- Message Space: $\mathcal{M}$={a,b,c,…,z}*
- Right Shift Operation
  - RS_1(a) = b
  - RS_1(b) = c
  - ...
  - RS_1(z) = ?
  - RS_{i+1}(a)=RS_i(b)

Shift Cipher

- Key Space: $\mathcal{K}$={0,1,…,25}
- Message Space: $\mathcal{M}$={a,b,c,…,z}*
- Right Shift Operation
    - $RS_1(a) = b$
    - $RS_1(b) = c$
    - ...
    - $RS_1(z) = a$
    - $RS_{i+1}(a)=RS_i(b)$
- $\mathrm{Enc_k}(m_1 \circ \cdots \circ m_n) = RS_k(m_1) \circ \cdots \circ RS_k\ (m_n)$
    - Each letter in plaintext message $\mathrm{m} = m_1 \circ \cdots \circ m_n$ is right shifted k times RS_k
- **Question:** what is ciphertext space $\mathcal{C}$?

Example: Shift Cipher (Multiple Characters)

- Key Space: $\mathcal{K}$={0,1,…,25}
- Message Space: $\mathcal{M}$={a,b,c,…,z}*

$$\mathrm{Enc_k}(m_1 \circ \cdots \circ m_n) = RS_k(m_1) \circ \cdots \circ RS_k\ (m_n)$$
$$\mathrm{Dec_k}(c_1 \circ \cdots \circ c_n) = LS_k(c_1) \circ \cdots \circ LS_k\ (c_n)$$

- **Note:**

$$\mathrm{Dec_k}(\mathrm{Enc_k}(m_1 \circ \cdots \circ m_n)) = m_1 \circ \cdots \circ m_n$$

since

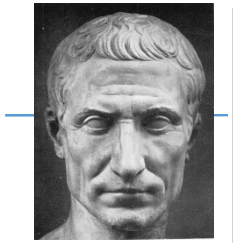
$$LS_k\big(RS_k(m_i)\big) = m_i$$

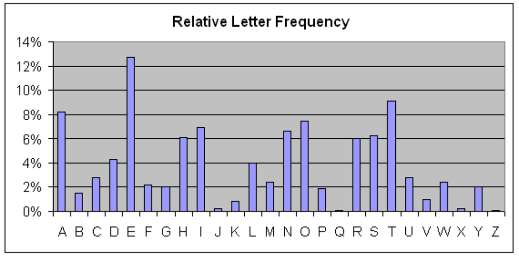
Topic 2: Historical Ciphers (& How to Break Them)

Cryptography History

- 2500+ years
- Ongoing battle
  - Codemakers and codebreakers

Caesar Shift Cipher (50 BC)

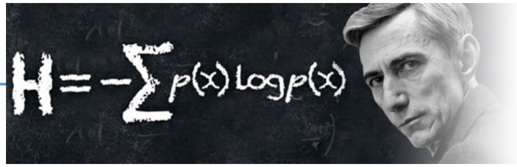



Relative Letter Frequency
14%
12%
10%
8%
6%
4%
2%
0%
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Frequency Analysis

Cipher Machines (1900s)

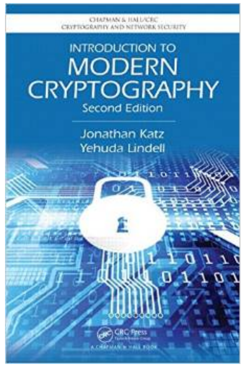
Shannon Entropy/Perfect Secrecy (~1950)



1970s

Public Key Crypto/RSA

Formalization of Modern Crypto (1976+)



Who Uses Cryptography

- Traditionally: Militias
- Modern Times: Everyone!

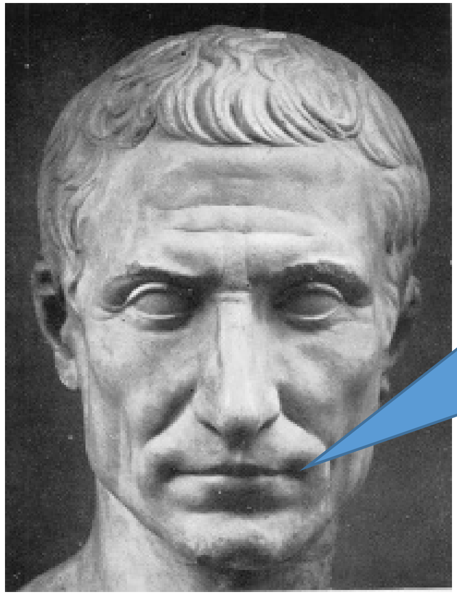
Caesar Shift Cipher (50 BC)

Revolutionary War

Modern Crypto

One if by Land
A Massachusetts Number Book
amazon

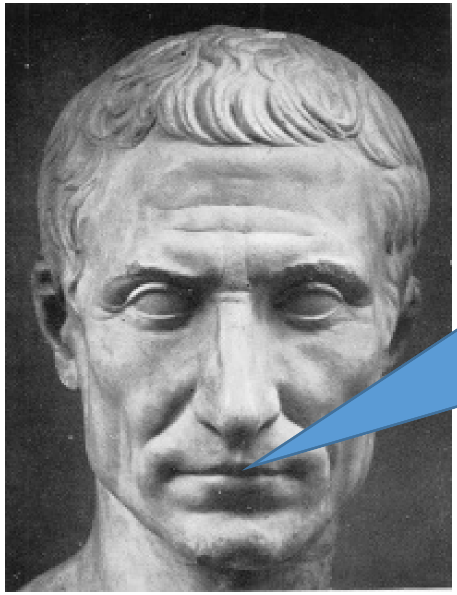
Caesar Cipher



Three shall be the number of thy shifting and the number of thy shifting shall be three. Four shalt thou not shift, neither shift thou two, excepting that thou then proceed to three. Five is right out.....

Caesar adopted the shift cipher with secret key k=3

Caesar Cipher (Example)



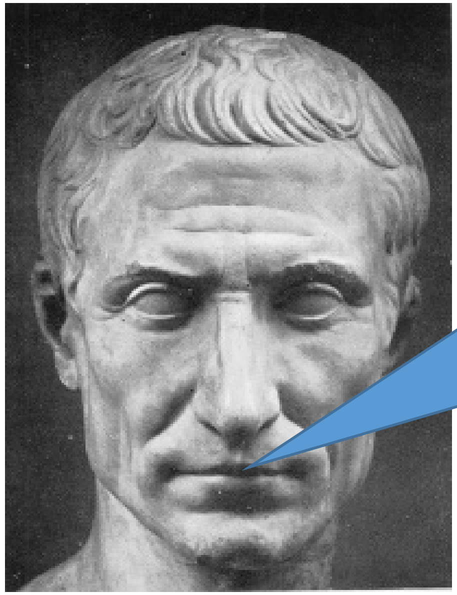
BEGINTHEATTACKNOW

→

EHJLQWKHDWWDFNQRZ

Caesar adopted the shift cipher with secret key k=3

Caesar Cipher (Example)



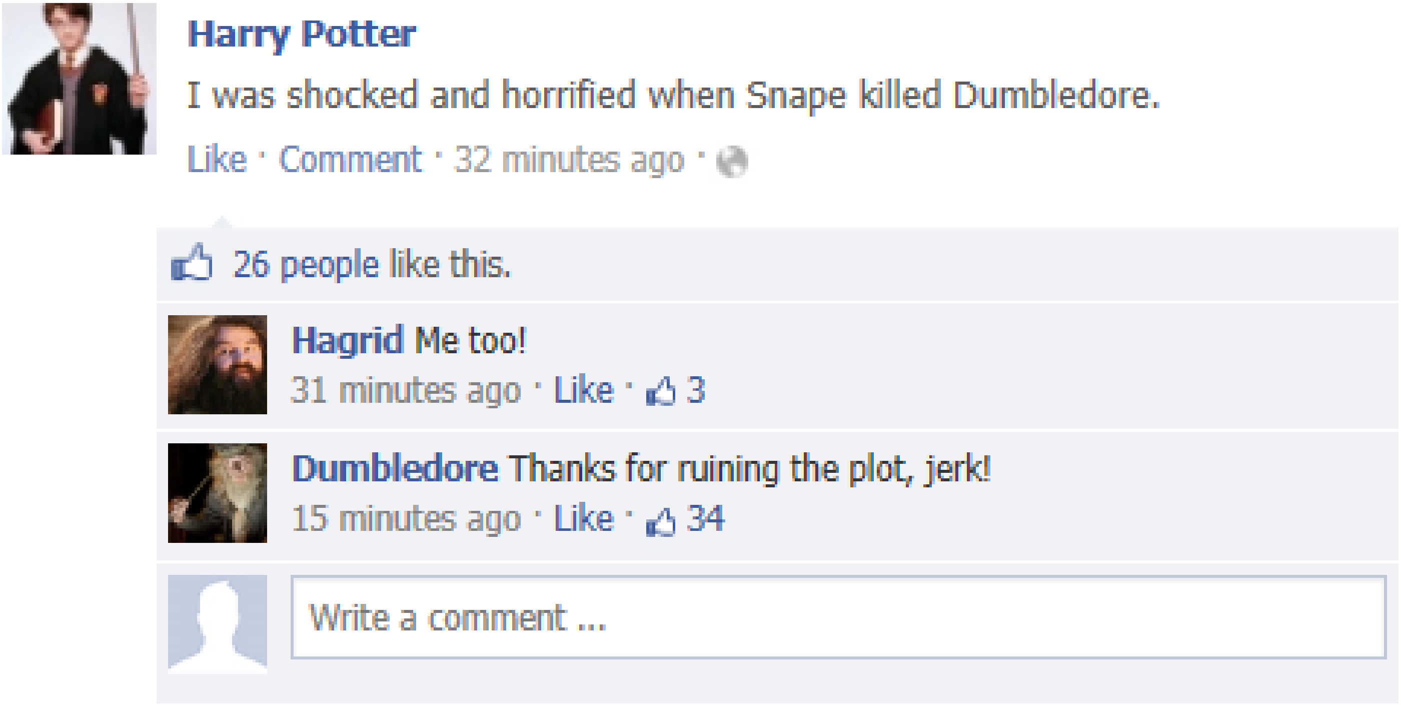
BEGINTHEATTACKNOW

→

EHJLQWKHDWWDFNQRZ

Immediate Issue: anyone who knows method can decrypt (since k=3 is fixed)

Modern Application: Avoid Spoilers (ROT13)



Harry Potter

I was shocked and horrified when Snape killed Dumbledore.

Like · Comment · 32 minutes ago ·

26 people like this.

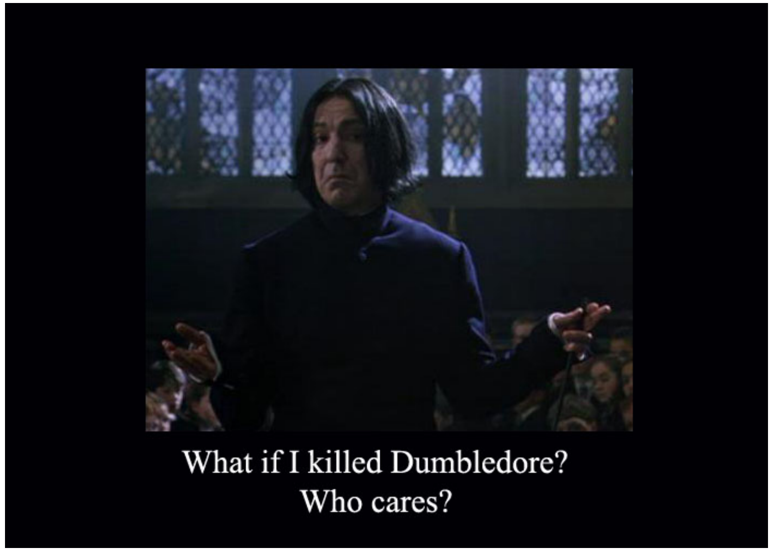
Hagrid Me too!

31 minutes ago · Like · 3

Dumbledore Thanks for ruining the plot, jerk!

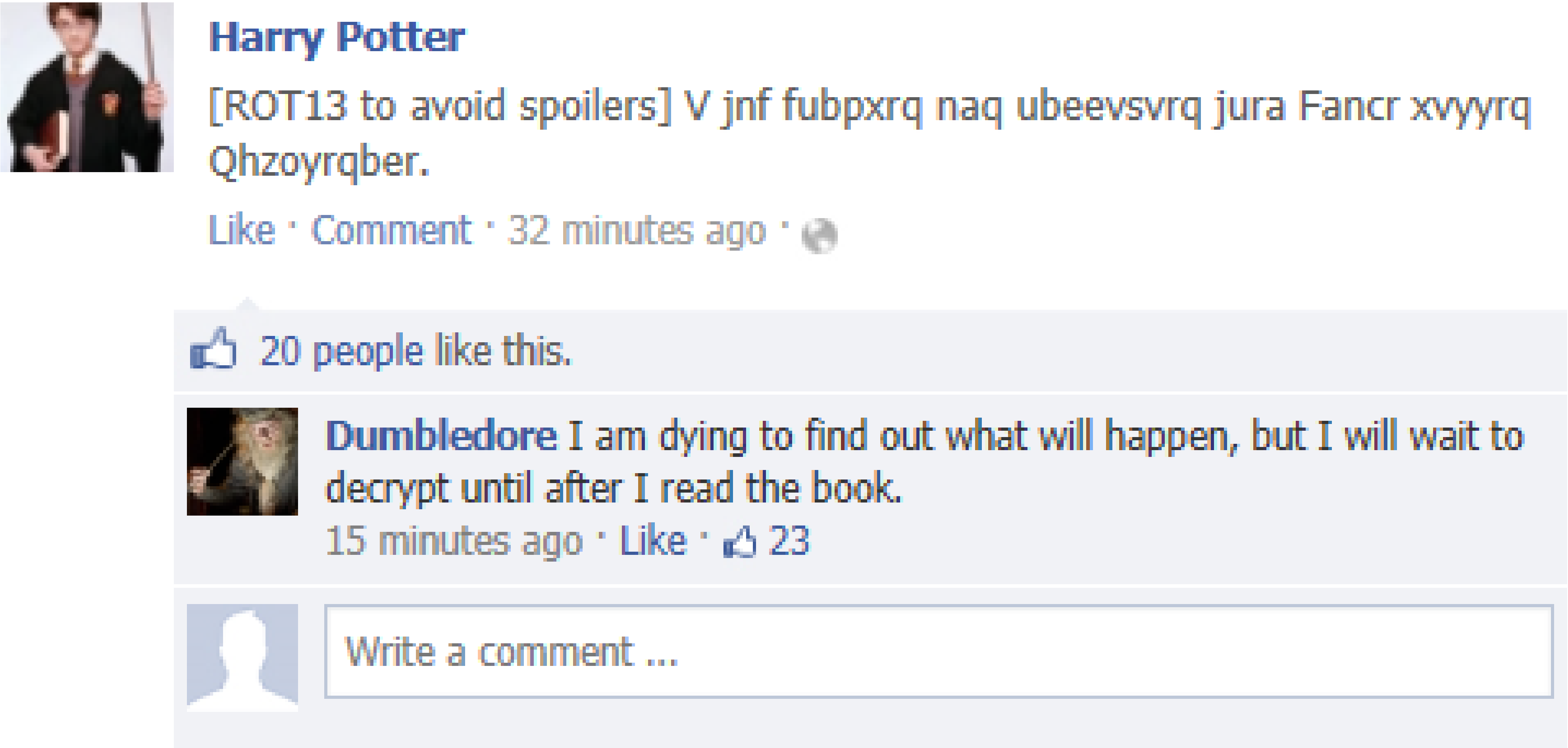
15 minutes ago · Like · 34

Write a comment ...



What if I killed Dumbledore?
Who cares?

Modern Application: Avoid Spoilers (ROT13)



Harry Potter

[ROT13 to avoid spoilers] V jnf fubpxrq naq ubeevsvrq jura Fancr xvyyrq Qhzoyrqber.

Dumbledore I am dying to find out what will happen, but I will wait to decrypt until after I read the book.

Shift Cipher: Brute Force Attack

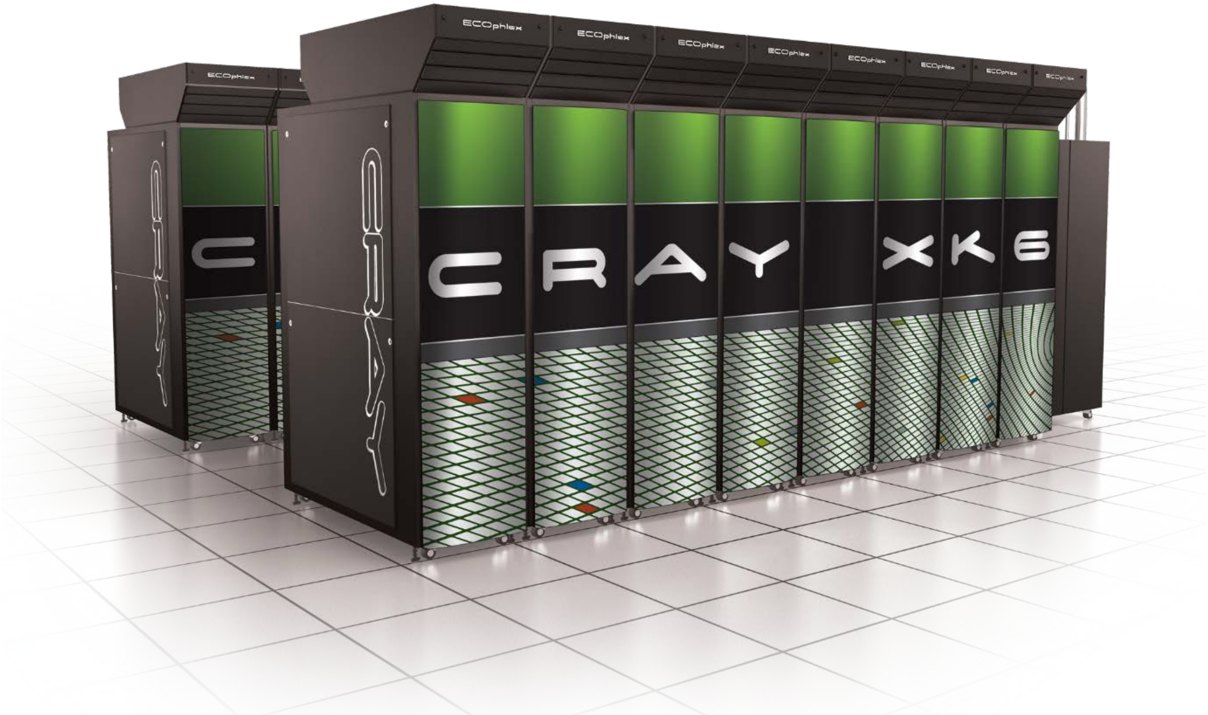
- Ciphertext: “lwxrw ztn sd ndj iwxcz xh gxvwi?”
  - k=1 → m = “mxysx auo te oek jxyda yi hywxj?”
  - k=2 → m=“nyzty bvp uf pfl kyzeb zj izxyk?”
  - k=3 → m=“ozauz cwq vg qgm lzafc ak jayzl?”
  - k=4 → m = “pabva dxr wh rhn mabgd bl kbzam?”
  - k=5 → m=“qbcwb eys xi sio nbche cm lcabn?”
  - k=6 → m=“rcdxc fzt yj tjp ocdif dn mdbco?”

Shift Cipher: Brute Force Attack

- Ciphertext: “lwxrw ztn sd ndj iwxcz xh gxvwi?”
  - …
  - k=7 → m=“sdeyd gau zk ukq pdejg eo necdp?”
  - k=8 → m=“tefze hbv al vlr qefkh fp ofdeq?”
  - k=9 → m = “ufgaf icw bm wms rfgli gq pgefr?”
  - k=10 → m=“vghbg jdx cn xnt sghmj hr qhfgs?”
  - k=11→ m= “which key do you think is right?”
  - k=12→ m= “xijdi lfz ep zpv uijol jt sjhiu?”

Sufficient Key Space Principle

"Any secure encryption scheme *must* have a key space that is sufficiently large to make an exhaustive search attack infeasible."



Sufficient Key Space Principle

"Any secure encryption scheme *must* have a key space that is sufficiently large to make an exhaustive search attack infeasible."

Question 1: How big is big enough? Complicated question….

Question 2: If the key space is large is the encryption scheme necessarily secure?

Substitution Cipher

- Secret key K is permutation of the alphabet
  - Example:
    - 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
    - X E U A D N B K V M R O C Q F S Y H W G L Z I J P T

- **Encryption:** apply permutation K to each letter in message
  - TELLHIMABOUTME → GDOOKVCXEFLGCD

- **Decryption:** reverse the permutation

Substitution Cipher

- Secret key K is a permutation of the alphabet
    - Example:
        - 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
        - X E U A D N B K V M R O C Q F S Y H W G L Z I J P T

- **Question:** What is the size of the keyspace $\mathcal{K}$?

$$|\mathcal{K}| = 26! \approx 2^{88}$$

Tuesday's Crypto Answers

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

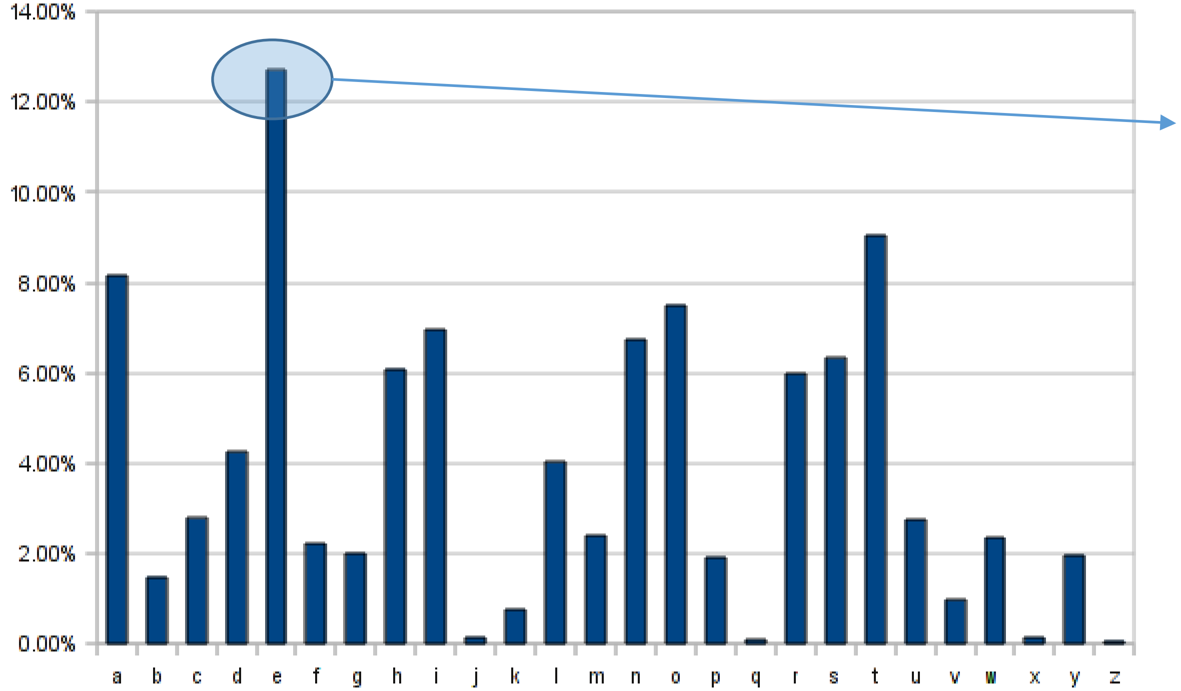
THE	ONLY	TIME	YOU	RUN	OUT
25 17 18	3 7 5 20	25 19 22 18	20 3 13	6 13 7	3 13 25

OF	CHANCES	IS	WHEN	YOU
3 11	23 17 2 7 23 18 15	19 15	26 17 18 7	20 3 13

STOP	TAKING	THEM.
15 25 3 14	25 2 4 19 7 1	25 17 18 22

Frequency Analysis

- **Observation 1:** If e is mapped to d then every appearance of e in the plaintext results in the appearance of a d in the ciphertext
- **Observation 2:** Some letters occur much more frequently in English.
- **Observation 3:** Texts consisting of a few sentences tend to have a distribution close to average.



Step 1: Find letter in ciphertext that occurs with frequency > 11%. This letter is probably e…

Vigenère Cipher

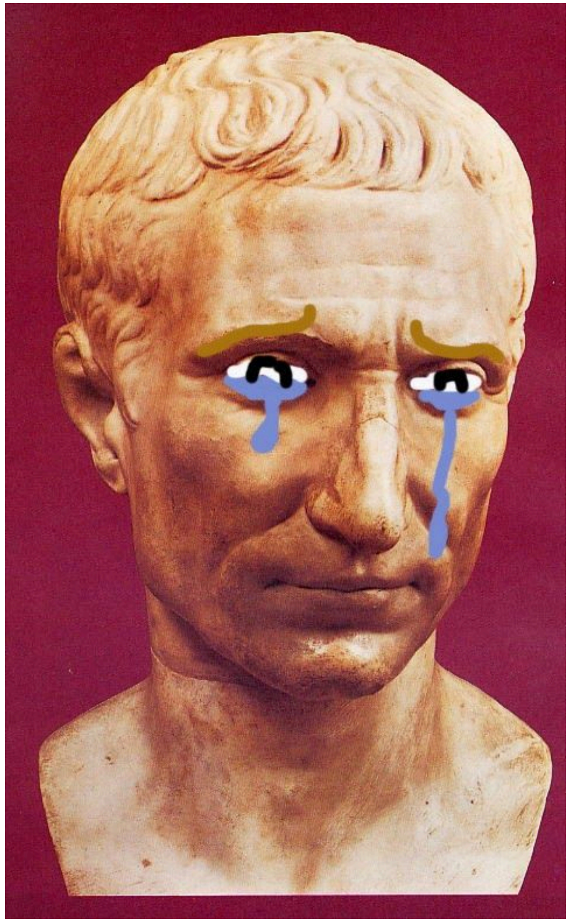
- Generalizes Shift Cipher
- $K=k_1,\ldots,k_t$
- $Enc_K(m)$
    - Shift first letter right k_1 times
    - Shift second letter right k_2 times
    - …
    - Shift t^{th} letter right k_t times
    - Shift $t+1^{st}$ letter right k_1 times
    - …
- **Question:** Size of key-space?
- Answer: 26^t (brute force may not be useful)

Vigenère Cipher

- Still vulnerable to frequency analysis
- Good guess: Select $K=k_1,\ldots,k_t$ to maximize number of e’s in resulting ciphertext
  - See Katz and Lindell 1.3 for even more sophisticated heuristics.
- Attack works when the initial message m is sufficiently long
- Vigenère is “perfectly secret” if the message m is at most t letters long.

Conclusions

- Designing secure ciphers is hard
- Vigenère remained “unbroken” for a long time
- Complex schemes are not secure
- All historical ciphers have fallen



Homework 1 Released

- Due: Thursday, February 4th at 11:59 PM on Gradescope (2 weeks)
- Solutions should be typeset (preferably in Latex)
- You may collaborate with classmates, but you must write up your own solution and you *must understand* this solution
- Ask clarification questions on Piazza or during office hours

Topic 3: Perfect Secrecy + One-Time-Pads

Principles of Modern Cryptography

- Need formal definitions of "security"

  If you don't understand what you want to achieve, how can you possibly know when (or if) you have achieved it?

  - Attempt 1: Impossible/infeasible for attacker to recover secret key K
    - $\mathrm{Enc}_{\mathrm{k}}(\mathrm{m}) = \mathrm{m}$
  - Attempt 2: Impossible for attacker to recover entire plaintext from ciphertext?
    - Ok to decrypt 90% of message?
  - Attempt 3: Impossible for attacker to figure out any particular character of the plaintext from the ciphertext?
    - [Too Weak] Does employee make more than $100,000 per year?
    - [Too Strong] Lucky guess? Prior Information? (e.g., letters always begin "Dear ….")

Principles of Modern Cryptography

- Need formal definitions of “security”

  If you don’t understand what you want to achieve, how can you possibly know when (or if) you have achieved it?

  - Final Attempt: Regardless of information an attacker *already* has, a ciphertext should leak no *additional information* about the underlying plaintext.
    - This is the “*right*” approach
    - Still need to *formalize* mathematically

- Security definition includes *goal* and *threat-model*

Principles of Modern Cryptography

- Proofs of Security are critical
  - Iron-clad guarantee that attacker will not succeed (relative to definition/assumptions)
- Experience: intuition is often misleading in cryptography
  - An “intuitively secure” scheme may actually be badly broken.

- Before deploying in the real world
  - Consider definition/assumptions in security definition
  - Does the threat model capture the attackers true abilities?

Perfect Secrecy Intuition

- Regardless of information an attacker *already* has, a ciphertext should leak no *additional information* about the underlying plaintext.

- We will formalize this intuition
    - And show how to achieve it

Private Key Encryption Syntax

- Message Space: $\mathcal{M}$
- Key Space: $\mathcal{K}$
- Three Algorithms $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$
  - $\mathrm{Gen}(R)$ (Key-generation algorithm)
    - **Input:** Random Bits R
    - **Output:** Secret key $\mathrm{k} \in \mathcal{K}$.
  - $\mathrm{Enc}_{\mathrm{k}}(m)$ (Encryption algorithm)
    - **Input:** Secret key $\mathrm{k} \in \mathcal{K}$ and message $\mathrm{m} \in \mathcal{M}$
    - **Output:** ciphertext *c*
  - $\mathrm{Dec}_{\mathrm{k}}(c)$ (Decryption algorithm)
    - **Input:** Secret key $\mathrm{k} \in \mathcal{K}$ and a ciphertex c
    - **Output:** a plaintext message $\mathrm{m} \in \mathcal{M}$

- **Invariant:** $\mathrm{Dec}_k(\mathrm{Enc}_k(m))=m$

Typically picks $\mathrm{k} \in \mathcal{K}$ uniformly at random

Trusted Parties (e.g., Alice and Bob) must run Gen in advance to obtain secret k.

Assumption: Adversary does not get to see output of Gen

An Example

- Enemy knows that Caesar likes to fight in the rain and it is raining today

$$\Pr[m = wait] = 0.3$$
$$\Pr[m = attack] = 0.7$$

- Suppose that Caesar sends c=Enc$_K$(m) to generals and that the attacker intercepts the ciphertext c and calculates

$$\Pr[m = wait\ |\text{c=EncK(m)}] = 0.2$$
$$\Pr[m = attack\ |\text{c=EncK(m)}] = 0.8$$

- Did the attacker learn anything useful?

Perfect Secrecy

Definition 1: An encryption scheme $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ with message space $\mathcal{M}$ is perfectly secret if for *every* probability distribution $\mathcal{D}$ over $\mathcal{M}$, every message $\mathrm{m} \in \mathcal{M}$ and every ciphertext $\mathrm{c} \in \mathcal{C}$ for which $\Pr[C = c] > 0$:

$$\Pr[M = m | C = c] = \Pr[M = m].$$

(where $M \leftarrow \mathcal{D}$, $K = Gen(R)$ and $C = \mathrm{Enc_K}(M)$)

Definition 2: For every $\mathrm{m}, \mathrm{m}' \in \mathcal{M}$ and $\mathrm{c} \in \mathcal{C}$

$$\Pr[\mathrm{Enc_K}(m) = c] = \Pr[\mathrm{Enc_K}(m') = c].$$

(where the probabilities are taken over the randomness of Gen and Enc)

Lemma 2.4: The above definitions are equivalent.

Perfect Secrecy

Definition 1: An encryption scheme $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ with message space $\mathcal{M}$ is perfectly secret if for *every* probability distribution $\mathcal{D}$ over $\mathcal{M}$, every message $\mathrm{m} \in \mathcal{M}$ and every ciphertext $\mathrm{c} \in \mathcal{C}$ for which $\Pr[C = c] > 0$:

$$\Pr[M = m | C = c] = \Pr[M = m].$$

(where $M \leftarrow \mathcal{D}$, $K = Gen(R)$ and $C = \mathrm{Enc}_{\mathrm{K}}(M)$)

Definition 2: For every $\mathrm{m}, \mathrm{m}' \in \mathcal{M}$ and $\mathrm{c} \in \mathcal{C}$

$$\Pr[\mathrm{Enc}_{\mathrm{K}}(m) = c] = \Pr[\mathrm{Enc}_{\mathrm{K}}(m') = c].$$

(where the probabilities are taken over the randomness of Gen and Enc)

Lemma 2.4: The above definitions are equivalent.

Definition 1 is more compelling as a security definition (attacker gains no information).

Easier to prove an encryption scheme satisfied definition 2.

Proof (Def 1 ➔ Def 2):

Suppose first that (Gen,Enc,Dec) does not satisfy definition 2. Then there exists $\mathrm{m}, \mathrm{m}' \in \mathcal{M}$ and $\mathrm{c} \in \mathcal{C}$ such that

$$\Pr[\mathrm{Enc}_{\mathrm{K}}(m) = c] \neq \Pr[\mathrm{Enc}_{\mathrm{K}}(m') = c] \ \ (1).$$

We will now prove that definition 1 does not hold. Define $\mathcal{D}$ such that

$$\Pr[M{=}m]{=}\Pr[M{=}m']{=}\frac{1}{2}$$

Assume for the sake of contradiction that Definition 1 were satisfied then we would have

$$\Pr[M = m|C = c] = \Pr[M = m] = \frac{1}{2}, \qquad \text{and}$$

$$\Pr[M = m'|C = c] = \Pr[M = m'] = \frac{1}{2}$$

which implies

$$\boxed{Pr[M = m|C = c] = Pr[M = m'|C = c] \ \ (*)}$$

Proof (Def 1 ➔ Def 2):

Suppose first that (Gen,Enc,Dec) does not satisfy definition 2. Then there exists $\mathrm{m}, \mathrm{m}' \in \mathcal{M}$ and $\mathrm{c} \in \mathcal{C}$ such that

$$\Pr[\mathrm{Enc}_{\mathrm{K}}(m) = c] \neq \Pr[\mathrm{Enc}_{\mathrm{K}}(m') = c] \ \ (1).$$

Define $\mathcal{D}$ such that Pr[M=m]=Pr[M=m']=$\frac{1}{2}$

Bayes Rule (1)

$$\Pr[M = m | \mathrm{Enc}_{\mathrm{K}}(M) = c] = \frac{\Pr[C = c | M = m]\Pr[\mathrm{M{=}m}]}{\Pr[\mathrm{C{=}c}]}$$

$$= \frac{1}{2}\frac{\Pr[\mathrm{Enc}_{\mathrm{K}}(m) = c]}{\Pr[\mathrm{C{=}c}]} \qquad (2)$$

Proof (Def 1 ➔ Def 2):

Suppose first that (Gen,Enc,Dec) does not satisfy definition 2. Then there exists $\mathrm{m}, \mathrm{m}' \in \mathcal{M}$ and $\mathrm{c} \in \mathcal{C}$ such that

$$\Pr[\mathrm{Enc}_{\mathrm{K}}(m) = c] \neq \Pr[\mathrm{Enc}_{\mathrm{K}}(m') = c] \ \ (1).$$

Define $\mathcal{D}$ such that Pr[M=m]=Pr[M=m']=$\frac{1}{2}$

Bayes Rule (2)

$$\Pr[M = m' | \mathrm{Enc}_{\mathrm{K}}(M) = c] = \frac{\Pr[C = c | M = m'] \Pr[\mathrm{M=m'}]}{\Pr[\mathrm{C=c}]}$$

$$= \frac{1}{2} \frac{\Pr[\mathrm{Enc}_{\mathrm{K}}(m') = c]}{\Pr[\mathrm{C=c}]} \qquad (3)$$

Proof (Def 1 ➔ Def 2):

Suppose first that (Gen,Enc,Dec) does not satisfy definition 2. Then there exists $\mathrm{m}, \mathrm{m}' \in \mathcal{M}$ and $\mathrm{c} \in \mathcal{C}$ such that

$$\Pr[\mathrm{Enc}_{\mathrm{K}}(m) = c] \neq \Pr[\mathrm{Enc}_{\mathrm{K}}(m') = c] \ \ (1).$$

Define $\mathcal{D}$ such that Pr[M=m]=Pr[M=m']=$\frac{1}{2}$

Combining equations (2) and (3), Bayes Rule implies that

$$\Pr[M = m' | \mathrm{Enc}_{\mathrm{K}}(M) = c] = \frac{\frac{1}{2}\Pr[\mathrm{Enc}_{\mathrm{K}}(m') = c]}{\Pr[C=c]}$$

$$\neq \frac{\frac{1}{2}\Pr[\mathrm{Enc}_{\mathrm{K}}(m) = c]}{\Pr[C=c]} = \Pr[M = m | \mathrm{Enc}_{\mathrm{K}}(M) = c] \ (**)$$

Proof (Def 1 ➔ Def 2):

Thus, Bayes Rule implies that

$$\Pr[M = m' | \mathrm{Enc}_{\mathrm{K}}(M) = c] = \frac{1}{2} \frac{\Pr[\mathrm{Enc}_{\mathrm{K}}(m') = c]}{\Pr[C{=}c]}$$

$$\neq \frac{1}{2} \frac{\Pr[\mathrm{Enc}_{\mathrm{K}}(m) = c]}{\Pr[C{=}c]} = \Pr[M = m | \mathrm{Enc}_{\mathrm{K}}(M) = c] \; (**)$$

We previously showed that definition 2 implies

$$\Pr[M = m | C = c] = \Pr[M = m' | C = c] \;\; (*)$$

Contradiction!

(Still need to prove Def 2 ➔ Def 1 --- See textbook for details)

Proof (Def 2 ➔ Def 1):

Assume that Definition 2 holds then for all messages m,m’ and ciphertexts we have

$$\Pr[\mathrm{Enc}_{\mathrm{K}}(m) = c] = \Pr[\mathrm{Enc}_{\mathrm{K}}(m') = c]$$

Now to show that Definition 1 holds we fix any distribution D and any message m and ciphertext c for which $\Pr[C = c] > 0$

(when $\mathrm{C} = \mathrm{Enc}_{\mathrm{K}}(M)$ for a randomly sampled message M from D)

We need to prove that $\Pr[M = m | C = c] = \Pr[M = m]$

Observation 1: If $\Pr[M = m] = 0$ then $\Pr[M = m | C = c] = 0 = \Pr[M = m]$

Proof (Def 2 ➔ Def 1):

We need to prove that $\Pr[M = m | C = c] = \Pr[M = m]$

Observation 1: If $\Pr[M = m] = 0$ then $\Pr[M = m | C = c] = 0 = \Pr[M = m]$

Otherwise, define $\mathrm{p_c} \coloneqq \Pr[C = c | M = m]$ and notice that

$$\Pr[C = c | M = m] = \Pr[\mathrm{Enc_K}(M) = c | M = m] = \Pr[\mathrm{Enc_K}(m) = c] \quad (1)$$

For any other message m’ we have

$$\Pr[\mathrm{Enc_K}(m') = c] \underset{\substack{\uparrow \\ \text{Def 1}}}{=} \Pr[\mathrm{Enc_K}(m) = c] \underset{\substack{\uparrow \\ \text{Eq 1}}}{=} \Pr[C = c | M = m] \underset{\substack{\uparrow \\ \text{Definition}}}{=} \mathrm{p_c}$$

Proof (Def 2 ➔ Def 1):

$$\boxed{\Pr[M=m|C=c]} = \frac{\Pr[C=c|M=m]\Pr[M=m]}{\Pr[C=c]} \quad \text{(Bayes Theorem)}$$

$$= \frac{\Pr[C=c|M=m]\Pr[M=m]}{\sum_{m'}\Pr[M=m']\Pr[C=c|M=m']}$$

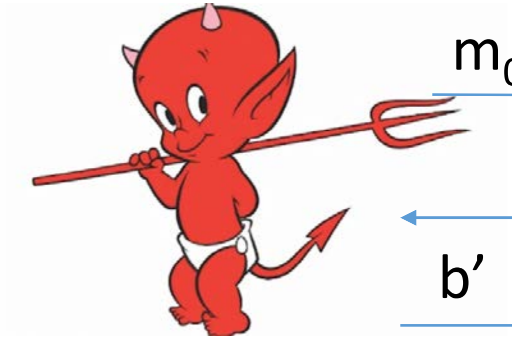
$$= \frac{\mathrm{p_c}\Pr[M=m]}{\sum_{m'}\Pr[M=m']\mathrm{p_c}} = \frac{\Pr[M=m]}{\sum_{m'}\Pr[M=m']}$$

$$= \frac{\Pr[M=m]}{1} = \boxed{\Pr[M=m]}$$

This is what we wanted to prove

QED

Another Equivalent Definition (Game)



m_0, m_1

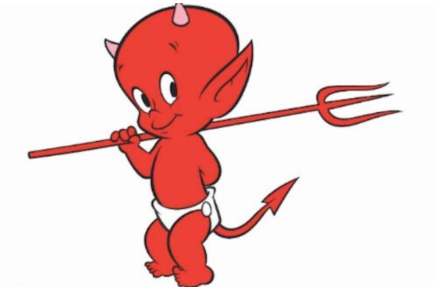
c

b'

Random bit b

K ← Gen(.)

c = Enc$_K$(m$_b$)

$\forall$  $\Pr[$ $Guesses\ b' = b] = \frac{1}{2}$

Another Equivalent Definition (Game)

Formally, let $\Pi = (Gen, Enc, Dec)$ *denote the encryption scheme, and let* A *denote an eavesdropping attacker.*

Call the game the adversarial indistinguishability experiment and define a random variable $PrivK^{eav}_{A,\Pi}$ *as follows*

$$PrivK^{eav}_{A,\Pi} = \begin{cases} 1 & \text{if } b = b' (attacker\ is\ correct) \\ 0 & \text{otherwise}(attacker\ is\ not\ correct) \end{cases}$$

Π *has indistinguishable encryptions in the presence of an eavesdropper if for all eavesdropping adversaries* A *we have*

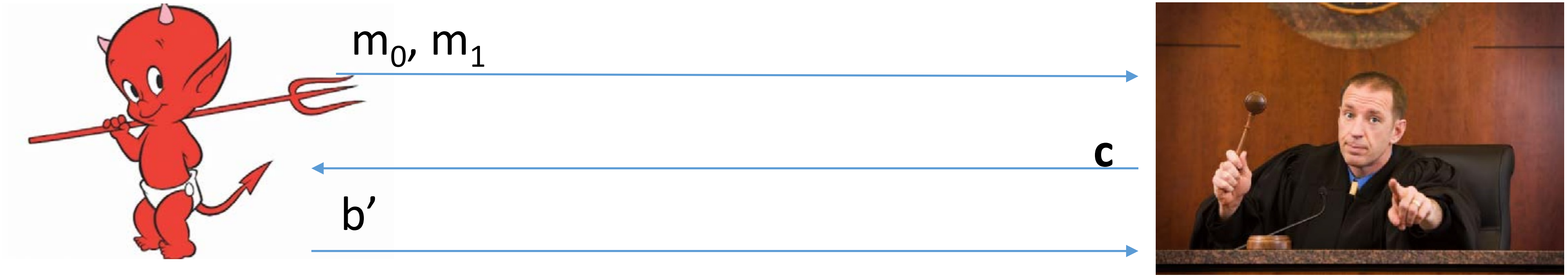
$$\Pr\left[PrivK^{eav}_{A,\Pi} = 1\right] = \frac{1}{2}$$

om bit b

en(.)

$c_K(m_b)$

Another Equivalent Definition (Game)



m_0, m_1

c

b’

Random bit b

K ← Gen(.)

c = $\mathrm{Enc}_K(m_b)$

Suppose we have m,m’,c’ s.t. $\Pr[\mathrm{Enc}_K(m) = c'] > \Pr[\mathrm{Enc}_K(m') = c']$ then the adversary can win the game w.p > ½. How?

What else do we need to establish to prove that the definitions are equivalent?

One Time Pad [Vernam 1917]

$$\mathrm{Enc}_K(m) = K\oplus m \qquad\qquad \mathrm{Dec}_K(c) = K\oplus c$$

$$\mathbf{Example} = \mathbf{1011}\oplus\mathbf{0011} = ???$$

Theorem: The one-time pad encryption scheme is perfectly secret

The following calculation holds for any c, m

$$\Pr[\mathrm{Enc}_K(m)=c] = \Pr[K\oplus m=c] = \Pr[K=c\oplus m] = {}^{1}/_{|\mathcal{K}|}.$$

Thus, for any m, m’, c we have

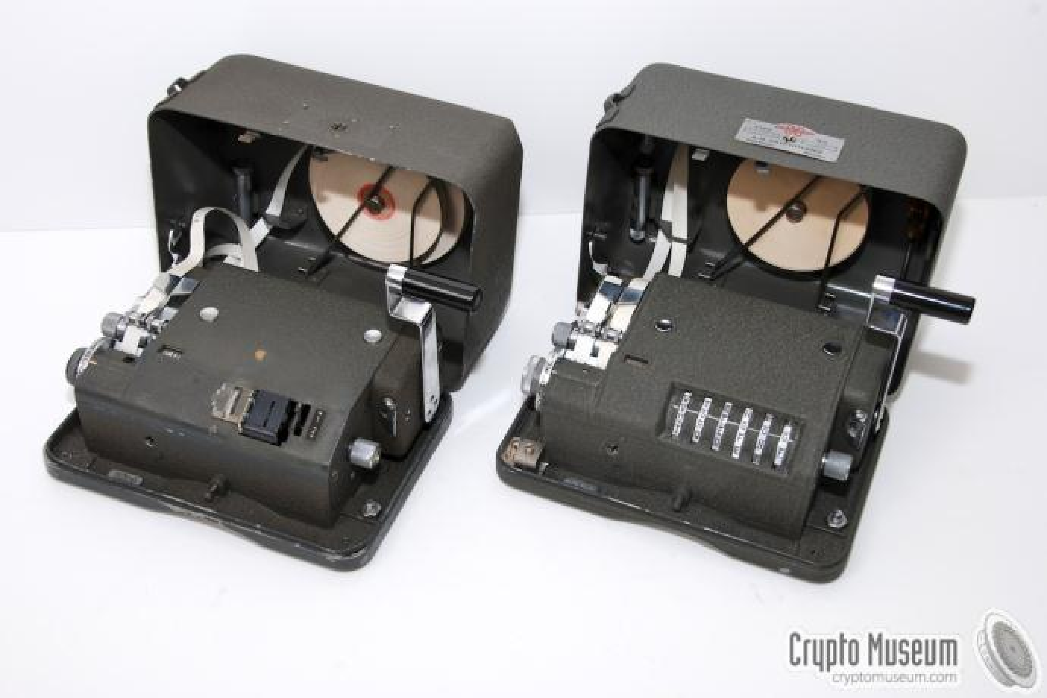
$$\Pr[\mathrm{Enc}_K(m)=c]={}^{1}/_{|\mathcal{K}|} =\Pr[\mathrm{Enc}_K(m')=c].$$

One Time Pad [Vernam 1917]

$$\mathrm{Enc}_K(m) = K \oplus m \qquad\qquad \mathrm{Dec}_K(c) = K \oplus c$$

$\mathbf{Example} = \mathbf{1011} \oplus \mathbf{0011}$ = ???

© D. Rijmenants 2009



Crypto Museum
cryptomuseum.com

One Time Pad

One Time Pad

Perfect Secrecy Limitations

Theorem: If (Gen,Enc,Dec) is a perfectly secret encryption scheme then

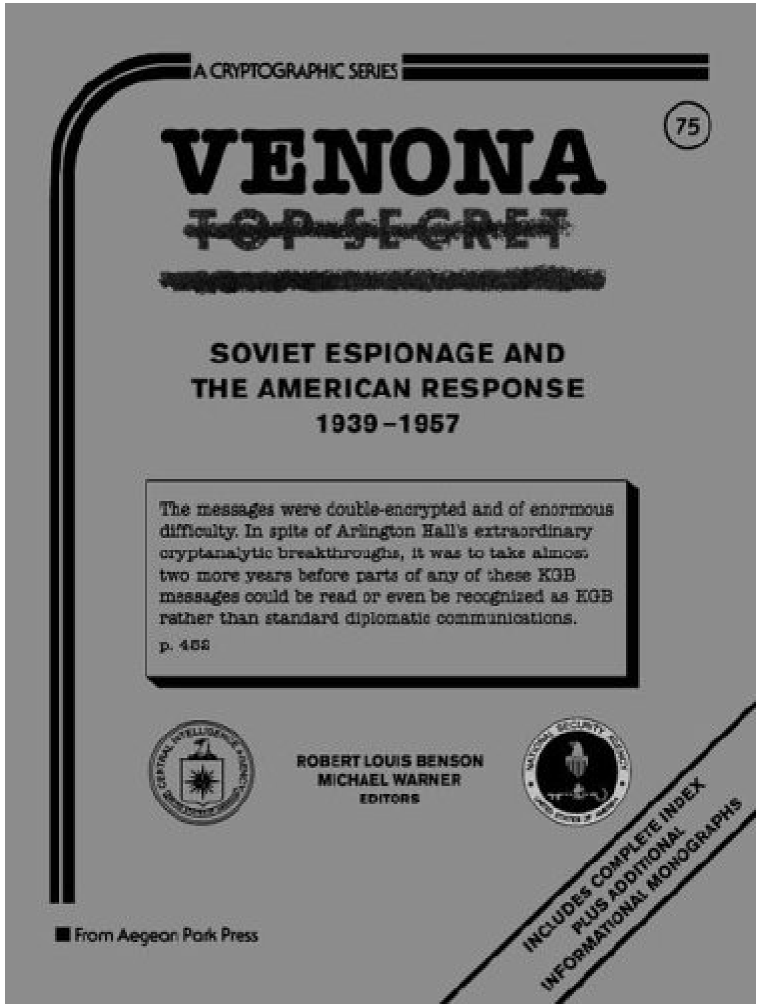
$$|\mathcal{K}| \geq |\mathcal{M}|$$

One Time Pad Limitations

- The key is as long as the message
  - How to exchange long messages?
  - Need to exchange/secure lots of one-time pads!
- OTPs can only be used once
  - As the name suggests
- VENONA project (US + UK)
  - Decrypt ciphertexts sent by Soviet Union which were mistakenly encrypted with portions of the same one-time pad over several decades

$$c \oplus c' = (m \oplus k) \oplus (m' \oplus k) = m \oplus m'$$

VENONA project



A CRYPTOGRAPHIC SERIES

75

VENONA

~~TOP SECRET~~

SOVIET ESPIONAGE AND
THE AMERICAN RESPONSE
1939–1957

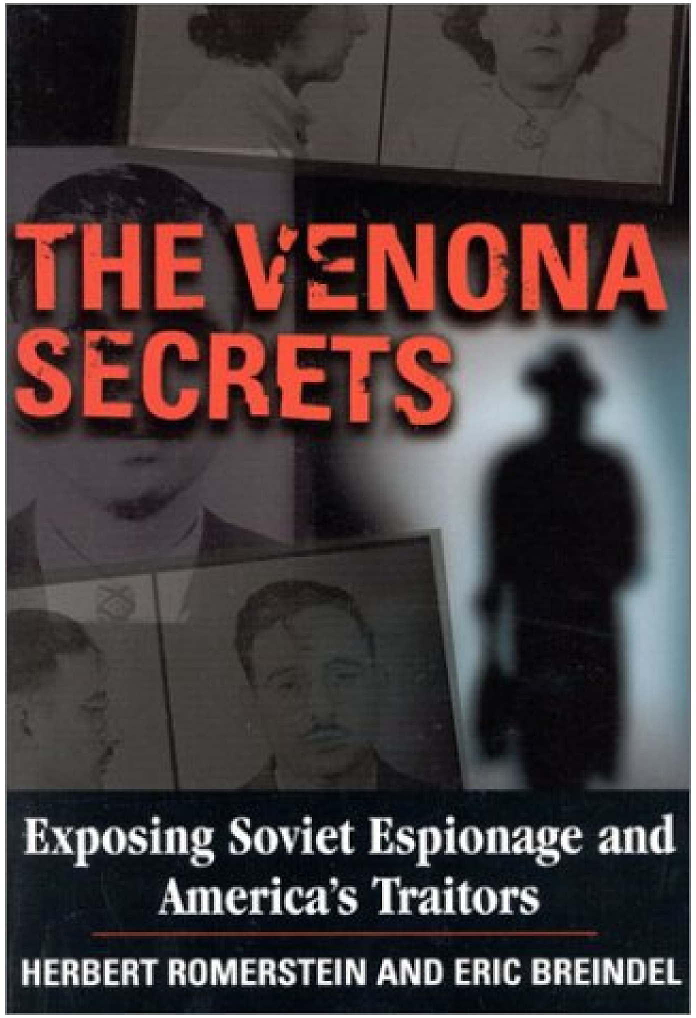
The messages were double-encrypted and of enormous difficulty. In spite of Arlington Hall's extraordinary cryptanalytic breakthroughs, it was to take almost two more years before parts of any of these KGB messages could be read or even be recognized as KGB rather than standard diplomatic communications.

p. 452

ROBERT LOUIS BENSON
MICHAEL WARNER
EDITORS

INCLUDES COMPLETE INDEX
PLUS ADDITIONAL
INFORMATIONAL MONOGRAPHS

■ From Aegean Park Press



THE VENONA SECRETS

Exposing Soviet Espionage and America's Traitors

HERBERT ROMERSTEIN AND ERIC BREINDEL

Shannon's Theorem

Theorem: Let (Gen,Enc,Dec) be an encryption scheme with $|\mathcal{K}| = |\mathcal{M}| = |\mathcal{C}|$. Then the scheme is perfectly secret if and only if:

1. Every key k ∈ $\mathcal{K}$ is chosen with (equal) probability ${}^{1}/_{|\mathcal{K}|}$ by the algorithm Gen, and
2. For every $\mathrm{m} \in \mathcal{M}$ and every c ∈ $\mathcal{C}$ there exists a unique key k ∈ $\mathcal{K}$ such that Enc$_k$(m)=c.

An Important Remark on Randomness

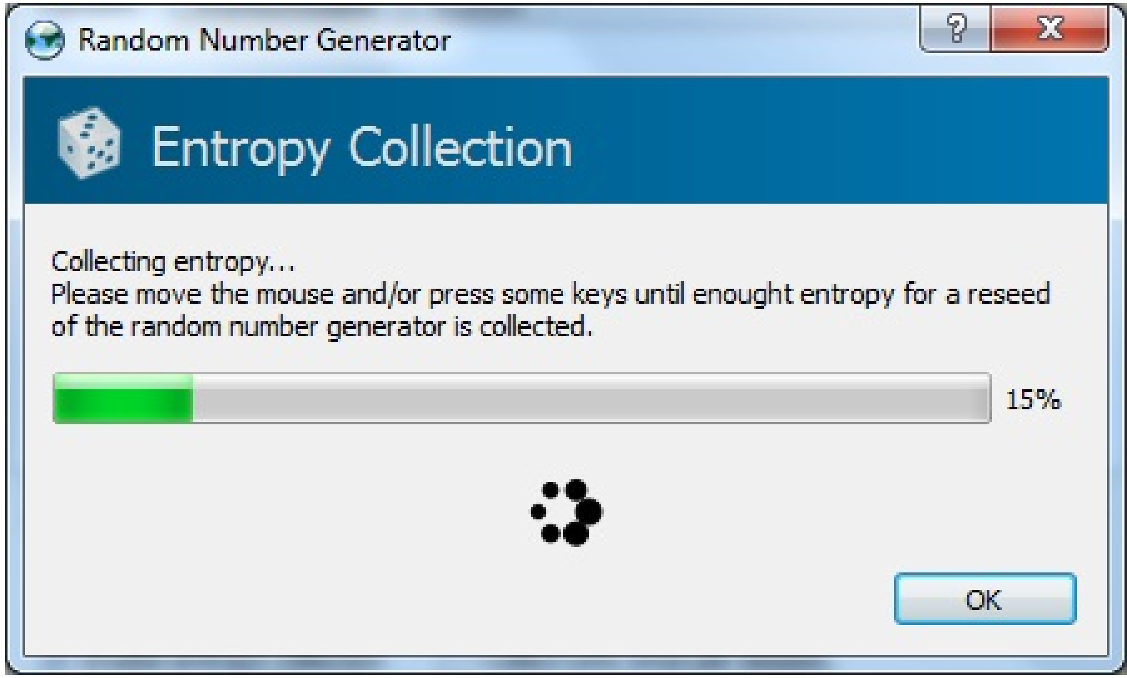
- In our analysis we have made (and will continue to make) a key assumption:
- We have access to true "randomness" to generate a secret key K

  Example: K = one time pad

- Independent Random Bits
  - Unbiased Coin flips
  - Radioactive decay?

In Practice

- Hard to flip thousands/millions of coins



Collecting entropy...
Please move the mouse and/or press some keys until enought entropy for a reseed of the random number generator is collected.

15%

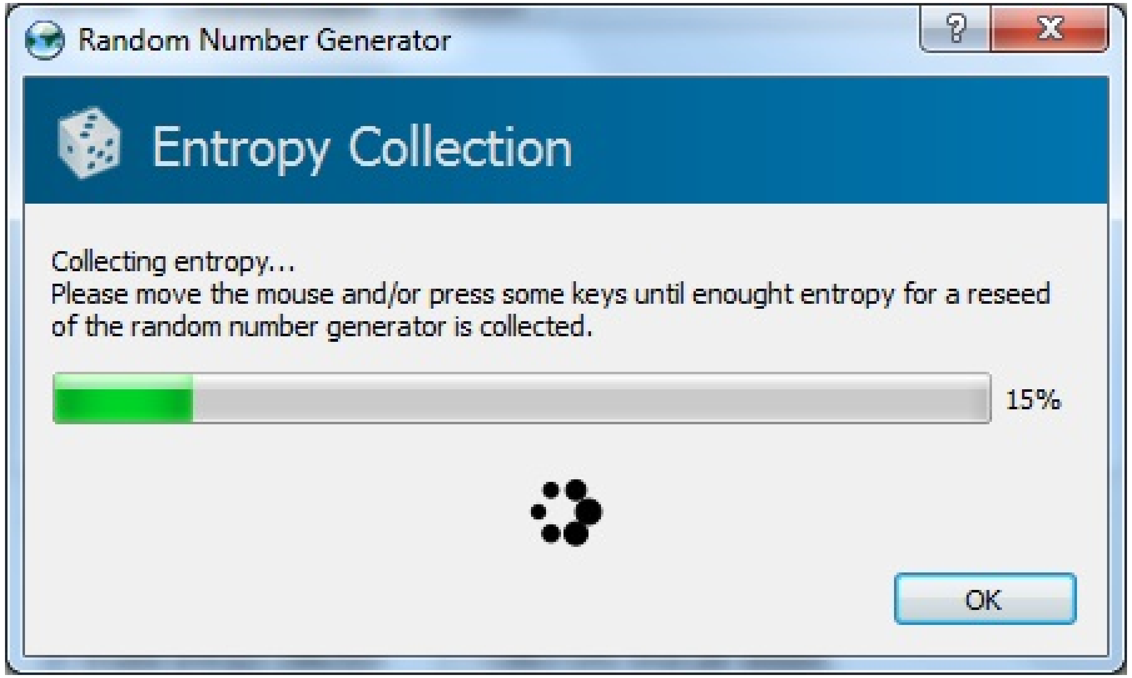
- Mouse-movements/keys
  - Uniform bits?
  - Independent bits?
- Use Randomness Extractors
  - As long as input has high entropy, we can extract (almost) uniform/independent bits
  - Hot research topic in theory

In Practice

- Hard to flip thousands/millions of coins

- Mouse-movements/keys

- Customized Randomness Chip?



Collecting entropy...
Please move the mouse and/or press some keys until enought entropy for a reseed of the random number generator is collected.

Caveat: Don’t do this!

- Rand() in C stdlib.h is no good for cryptographic applications

- Source of many real

world flaws

Perfect Secrecy

- What capabilities do we assume the attacker has?
  - Eavesdropping (Passive Adversary)
  - That’s it!
  - **Implicit Assumption**: No ability to tamper with messages!

Remark on One-Time Pads: If attacker has the ability to tamper with the ciphertext then s/he can easily flip the last bit of the message. How?

Answer: Flip the last bit of the intercepted ciphertext $c = K \oplus m$ to obtain $c' = c \oplus 00 \ldots 01$

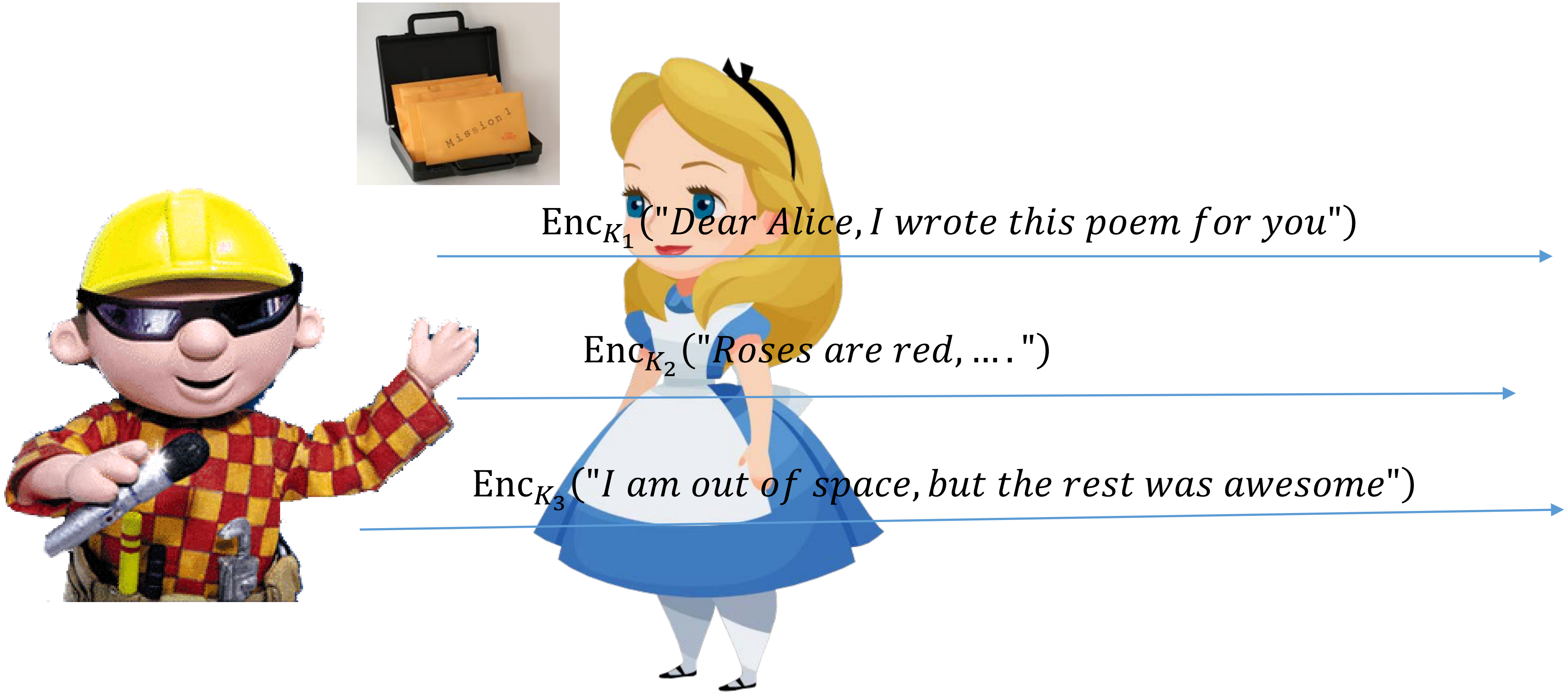
$$\mathrm{Dec}_K(c') = K \oplus c' = (K \oplus c) \oplus 00 \ldots 01 = m \oplus 00 \ldots 01$$

Week 1: Topic 4: Computational Security

What if we want to send a longer message?

K_1, K_2, K_3

K_1, K_2, K_3

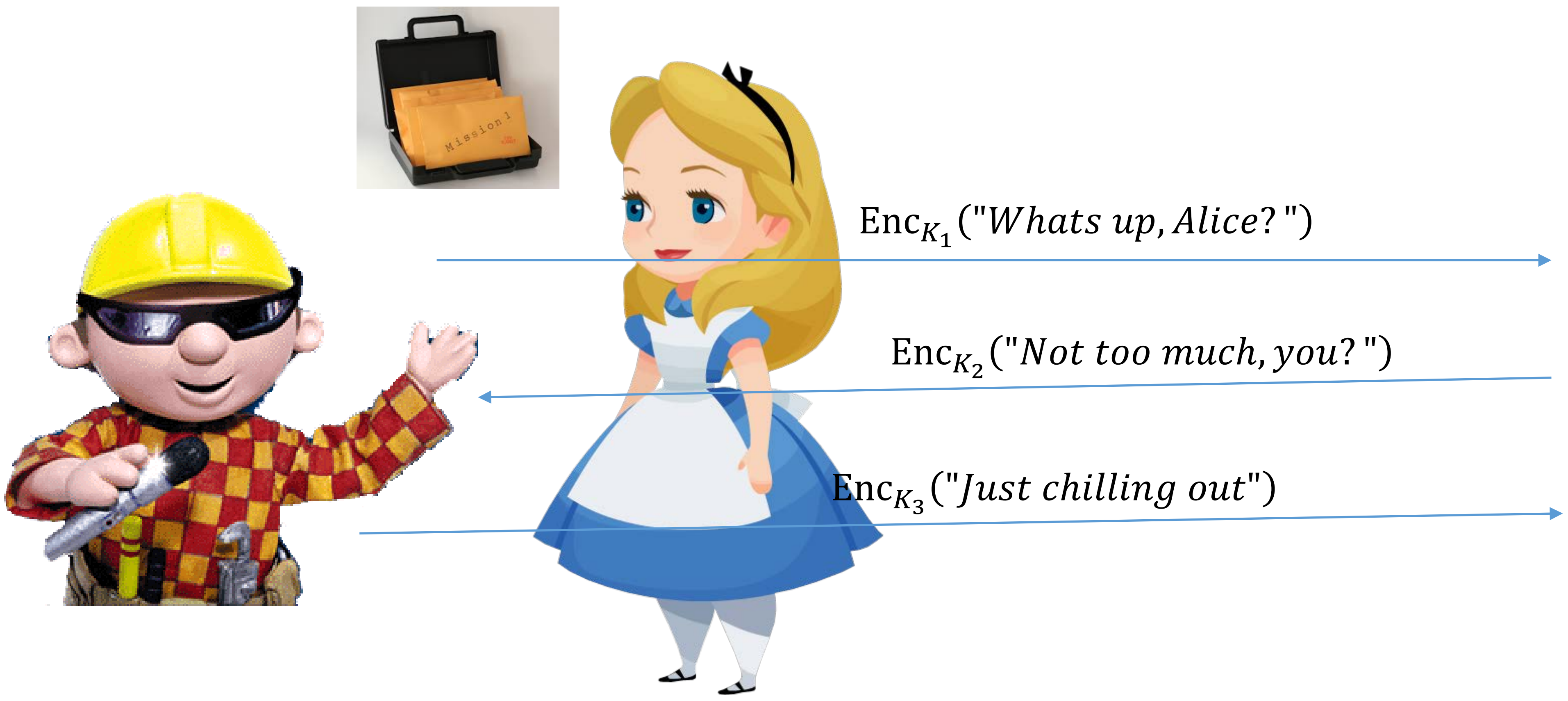


$\mathrm{Enc}_{K_1}(\text{"}Dear\ Alice, I\ wrote\ this\ poem\ for\ you\text{"})$

$\mathrm{Enc}_{K_2}(\text{"}Roses\ are\ red, \ldots.\text{"})$

$\mathrm{Enc}_{K_3}(\text{"}I\ am\ out\ of\ space, but\ the\ rest\ was\ awesome\text{"})$

What if we want to send many messages?



K_1,K_2,K_3

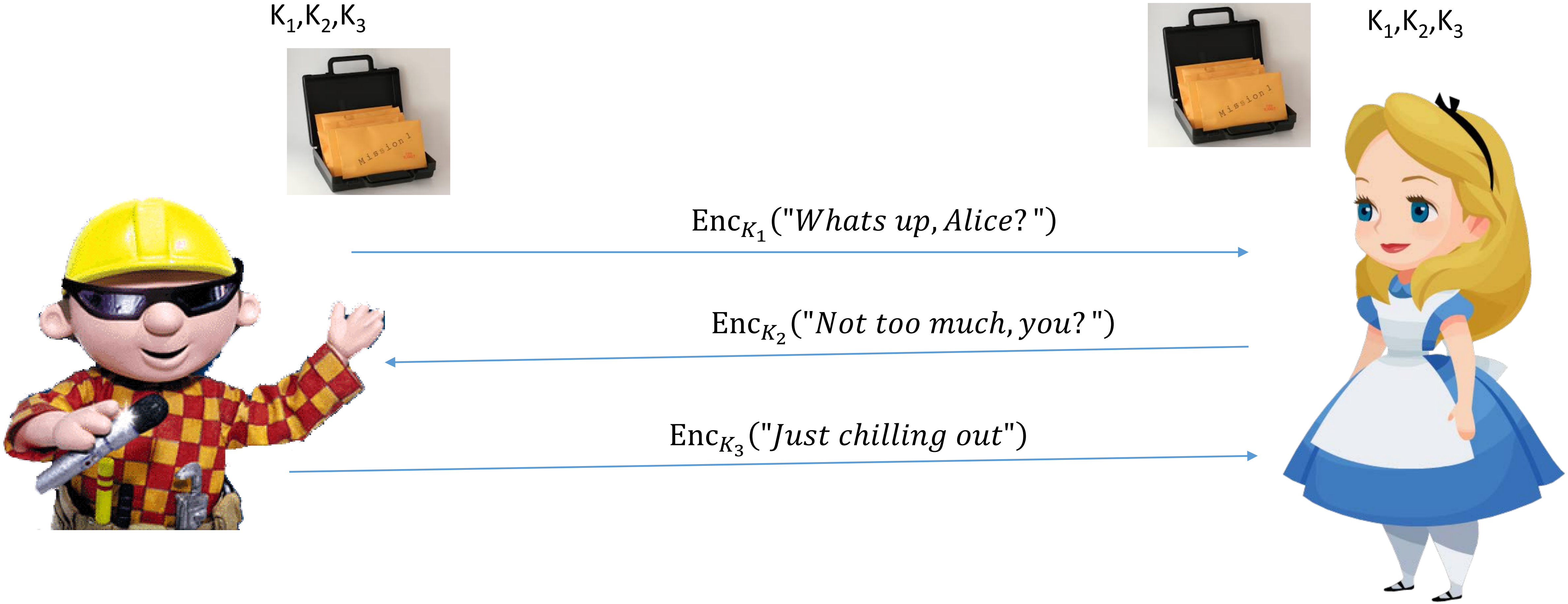
K_1,K_2,K_3

$\mathrm{Enc}_{K_1}(\text{"}Whats\ up, Alice?\text{"})$

$\mathrm{Enc}_{K_2}(\text{"}Not\ too\ much, you?\text{"})$

$\mathrm{Enc}_{K_3}(\text{"}Just\ chilling\ out\text{"})$

Can we save their relationship?



K$_1$,K$_2$,K$_3$

K$_1$,K$_2$,K$_3$

$\mathrm{Enc}_{K_1}(\text{"}Whats\ up, Alice?\text{"})$

$\mathrm{Enc}_{K_2}(\text{"}Not\ too\ much, you?\text{"})$

$\mathrm{Enc}_{K_3}(\text{"}Just\ chilling\ out\text{"})$

Perfect Secrecy vs Computational Security

- Perfect Secrecy is Information Theoretic
  - Guarantee is independent of attacker resources

- Computational Security
  - Security against computationally bounded attacker
    - An attacker with infinite resources might break security
  - Attacker might succeed with very small probability
    - Example: Lucky guess reveals secret key
    - Very Small Probability: $2^{-100}, 2^{-1000}, \ldots$

Current Goal

- Define computational security in presence of eavesdropper who intercepts a single (long) message

If you don’t understand what you want to achieve, how can you possibly know when (or if) you have achieved it?

- ~~Show how to build a symmetric encryption scheme with computational security in the presence of an eavesdropper.~~
- ~~Define computational security against an active attacker who might modify the message~~
- ~~Define computational security for multiple messages in presence of an eavesdropper~~

Concrete Security

> “A scheme is (t,ε)-secure if **every** adversary running for time at most t succeeds in breaking the scheme with probability at most ε”

- Example: $t = 2^{60}$ CPU cycles
  - 9 years on a 4GHz processor
  - < 1 minute on fastest supercomputer (in parallel)
- Full formal definition needs to specify “break”
- Important Metric in Practice
  - **Caveat 1**: difficult to provide/prove such precise statements
  - **Caveat 2**: hardware improves over time

Asymptotic Approach to Security

A scheme is secure if every *probabilistic polynomial time* (ppt) adversary “succeeds” with *negligible* probability.

- Two Key Concepts
  - Polynomial time algorithm
  - Negligible Function

Definition: A function $f\colon \mathbb{N} \longrightarrow \mathbb{R}_{\geq 0}$ is negligible if for every positive polynomial p there is an integer N>0 such that for all n > N we have

$$f(n) < \frac{1}{p(n)}$$

Asymptotic Approach to Security

Definition: A function $f\colon \mathbb{N} \longrightarrow \mathbb{R}_{\geq 0}$ is negligible if for every positive polynomial $p(.) > 0$ there is an integer N>0 such that for all n > N we have

$$f(n) < \frac{1}{p(n)}$$

Intuition: If we choose the security parameter n to be sufficiently large then we can make the adversaries success probability very small (negligibly small).

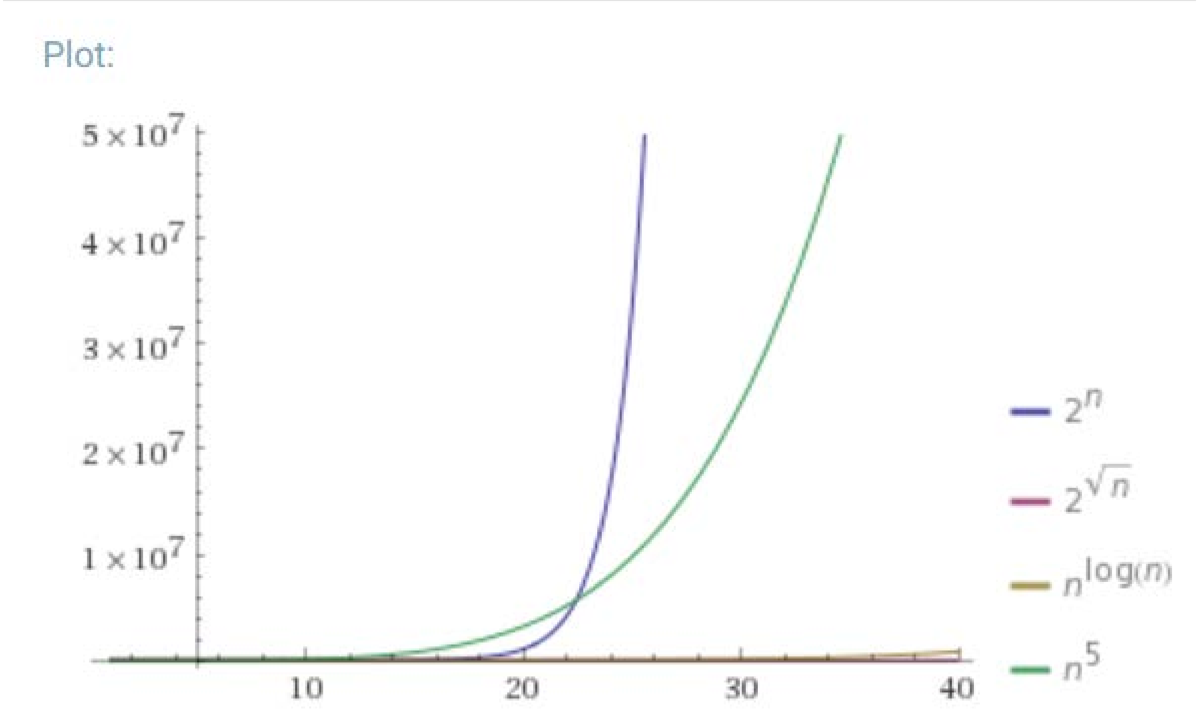
Asymptotic Approach to Security

Definition: A function $f\colon \mathbb{N} \longrightarrow \mathbb{R}_{\geq 0}$ is negligible if for every positive polynomial p there is an integer N>0 such that for all n > N we have

$$f(n) < \frac{1}{p(n)}$$

Which functions below are negligible?

- $f(n) = 2^{-n}$
- $f(n) = n^{-5}$
- $f(n) = 2^{-1000}1000n^{1000}$
- $f(n) = 2^{100}2^{-\sqrt{n}}$
- $f(n) = 2^{-\log n}$
- $f(n) = n^{-\log n}$



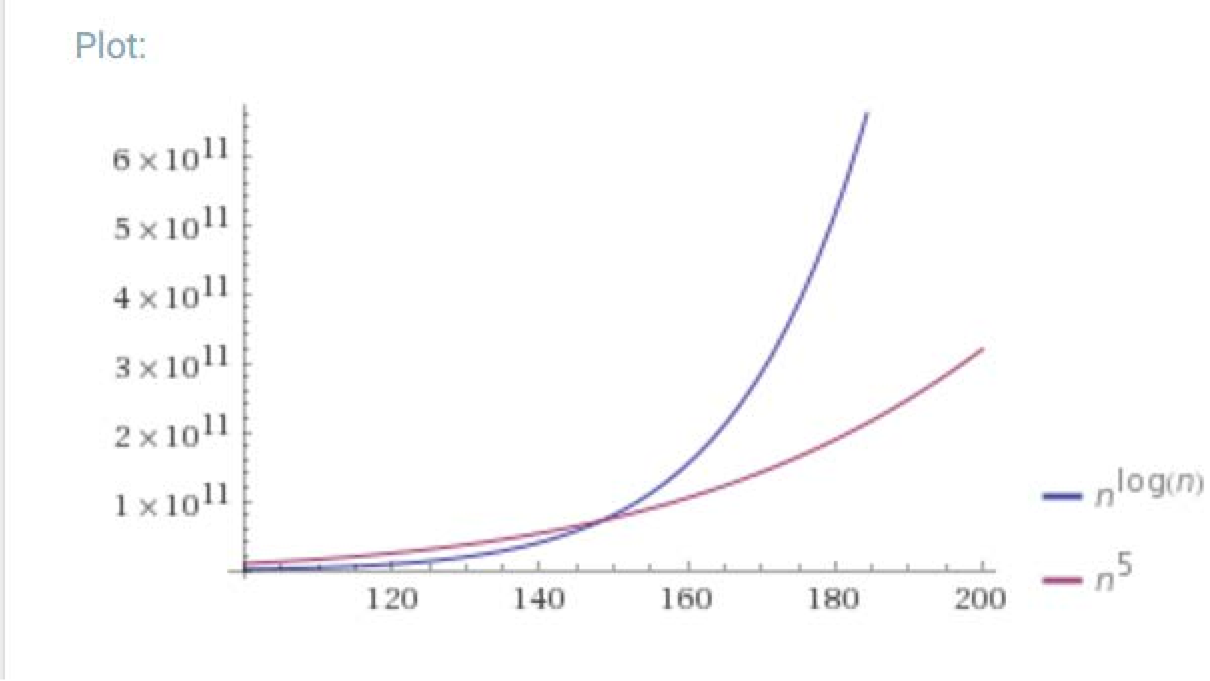
Asymptotic Approach to Security

Definition: A function $f\colon \mathbb{N} \longrightarrow \mathbb{R}_{\geq 0}$ is negligible if for every positive polynomial p there is an integer N>0 such that for all n > N we have

$$f(n) < \frac{1}{p(n)}$$

Which functions below are negligible?

- $f(n) = 2^{-n}$
- $f(n) = n^{-5}$
- $f(n) = 2^{-1000}1000n^{1000}$
- $f(n) = 2^{100}2^{-\sqrt{n}}$
- $f(n) = 2^{-\log n}$
- $f(n) = n^{-\log n}$



Asymptotic Approach to Security

Definition: An (randomized) algorithm A runs in polynomial time if there exists a polynomial p(.) such that for every n-bit input x, A(x) terminates in at most p(n) steps in expectation.

Intuition: If an algorithm A does not run in polynomial time then, for sufficiently large n, it will quickly become impractical for any attacker to run the algorithm A.

Asymptotic Approach to Security

A scheme is secure if every *probabilistic polynomial time* (ppt) adversary “succeeds” with *negligible* probability.

- **General Attack 1:** Test all possible secret keys $\mathrm{k}' \in \mathcal{K}$
  - Doesn’t run in polynomial time, since $|\mathcal{K}| = 2^n$
- **General Attack 2:** Select random key $\mathrm{k}' \in \mathcal{K}$, check if it is correct (otherwise output $\perp$ for “fail”).
  - Only successful with negligible probability 2^{-n}

Advantages of Asymptotic Approach

- **Closure**
  - If subroutine B runs in polynomial time and algorithm A makes poly(n) queries to the subroutine B then A also runs in polynomial time.
  - If f and g are negligible functions then h(n) = f(n)+g(n) is a negligible function
  - If p(.) is a positive polynomial, and f(.) is a negligible function then the function g(n)=f(n)p(n) is also negligible.
- **Church-Turing Thesis**: “reasonable” model of computations are all polynomially equivalent.
- **Implication**: No need to worry about different models of computation (circuits, random access machines, etc…)
- **Disadvantage:** Limited guidance on how big to make security parameter n in practice.

Note: Asymptotic vs Concrete Security

- **Theory of Cryptography:** Often follows Asymptotic Approach
- Course Textbook (Katz-Lindell) follows the asymptotic approach
- **Applied Cryptography:** Concrete Security Analysis is more useful
- **This Course:** We will consider both approaches

Private Key Encryption Syntax (Revisited)

- Message Space: $\mathcal{M}$
- Key Space: $\mathcal{K}$
- Three Algorithms
  - $\mathrm{Gen}(\mathbf{1}^{\boldsymbol{n}}; R)$ (Key-generation algorithm)
    - **Input:** **1^n (security parameter in unary)** + Random Bits
    - **Output:** Secret key $\mathrm{k} \in \mathcal{K}$
  - $\mathrm{Enc}_{\mathrm{k}}(m; \boldsymbol{R})$ (Encryption algorithm)
    - **Input:** Secret key $\mathrm{k} \in \mathcal{K}$ and message $\mathrm{m} \in \mathcal{M}$ + Ra
    - **Output:** ciphertext *c*
  - $\mathrm{Dec}_{\mathrm{k}}(c)$ (Decryption algorithm)
    - **Input:** Secret key $\mathrm{k} \in \mathcal{K}$ and a ciphertex c
    - **Output:** a plaintext message $\mathrm{m} \in \mathcal{M}$ *or* $\perp$ $(\boldsymbol{i.e}$*"Fail")*

- Invariant: $\mathrm{Dec}_k(\mathrm{Enc}_k(\mathrm{m}))=\mathrm{m}$

Requirement: all three algorithms run in probabilistic polynomial time

Quick Comment on Notation:
$\mathrm{K} = \mathrm{Gen}(\mathbf{1}^{\boldsymbol{n}}; R)$ vs.
$\mathrm{K} \leftarrow \mathrm{Gen}(\mathbf{1}^{\boldsymbol{n}})$

Private Key Encryption Syntax (Revisited)

- Message Space: $\mathcal{M}$
- Key Space: $\mathcal{K}$
- Three Algorithms
  - $\mathrm{Gen}(\mathbf{1}^{\boldsymbol{n}}; R)$ (Key-generation algorithm)
    - **Input: 1ⁿ (security parameter in unary)** + Random Bits
    - **Output:** Secret key $\mathrm{k} \in \mathcal{K}$
  - $\mathrm{Enc}_{\mathrm{k}}(m; \boldsymbol{R})$ (Encryption algorithm)
    - **Input:** Secret key $\mathrm{k} \in \mathcal{K}$ and message $\mathrm{m} \in \mathcal{M}$ + Ra
    - **Output:** ciphertext *c*
  - $\mathrm{Dec}_{\mathrm{k}}(c)$ (Decryption algorithm)
    - **Input:** Secret key $\mathrm{k} \in \mathcal{K}$ and a ciphertex c
    - **Output:** a plaintext message $\mathrm{m} \in \mathcal{M}$ *or* $\perp$ ***(i.e"Fail")***
- Invariant: $\mathrm{Dec}_{\mathrm{k}}(\mathrm{Enc}_{\mathrm{k}}(\mathrm{m}))=\mathrm{m}$

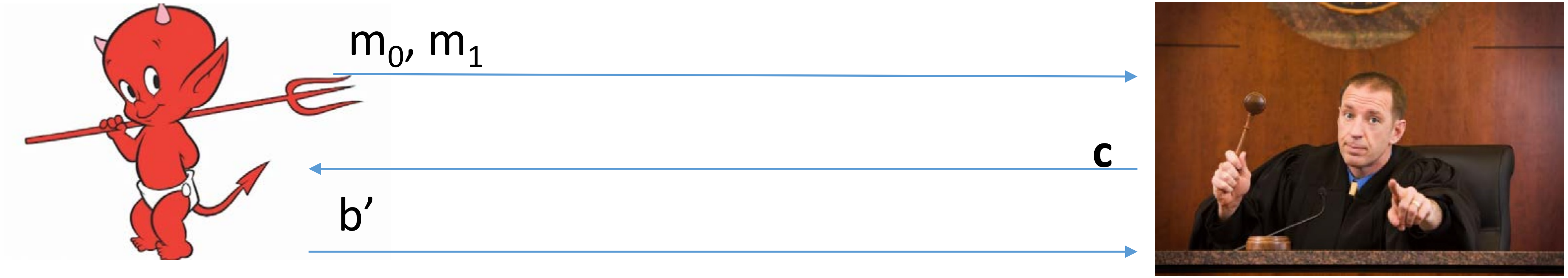
Requirement: all three algorithms run in probabilistic polynomial time

Quick Comment on Notation:

$\mathrm{K} = \mathrm{Gen}(\mathbf{1}^{\boldsymbol{n}}; R)$ vs.

$\mathrm{K} \leftarrow \mathrm{Gen}(\mathbf{1}^{\boldsymbol{n}})$

Adversarial Indistinguishability Experiment



Random bit b
K ← Gen(1^n)
c ← Enc$_K$(m_b)

ppt attacker

negligible function

$\forall$ $\exists\,\mu$ $\Pr\Big[$ $Guesses\ b' = b\Big] \le \frac{1}{2} + \mu$ (n)

Adversarial Indistinguishability Experiment

bit b

(1^n)

(m_b)

$$Formally, let\ \Pi = (Gen, Enc, Dec)\ denote\ the\ encryption\ scheme,$$
$$call\ the\ game\ the\ adversarial\ indistinguishability\ experiment\ and$$
$$define\ a\ random\ variable\ PrivK_{A,\Pi}^{eav}(1^n)\ as\ follows$$

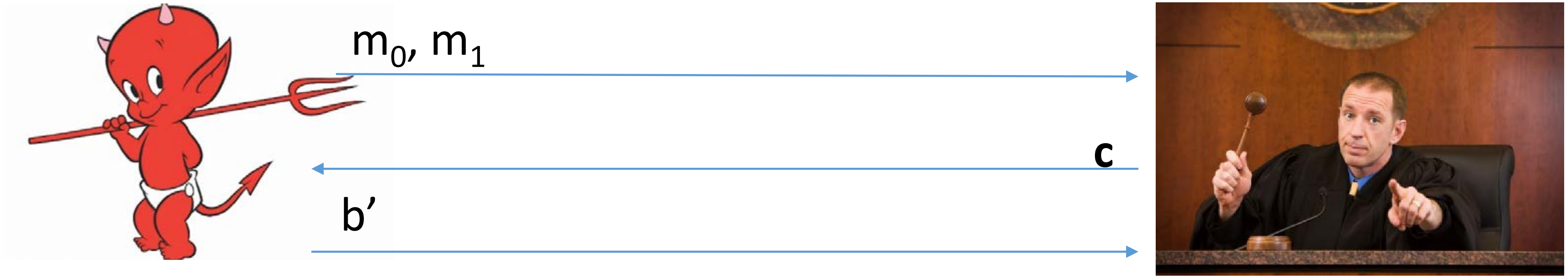
$$PrivK_{A,\Pi}^{eav}(1^n) = \begin{cases} 1 & \text{if } b = b' \\ 0 & \text{otherwise} \end{cases}$$

$$\Pi\ has\ indistinguishable\ encryptions\ in\ the\ presence\ of$$
$$an\ eavesdropper\ if\ for\ all\ PPT\ adversary\ A, there\ exists\ \ a$$

negligible function $\mu(.)$ such that

$$\Pr[PrivK_{A,\Pi}^{eav} = 1] \le \frac{1}{2} + \mu(n)$$

EAV-Secure



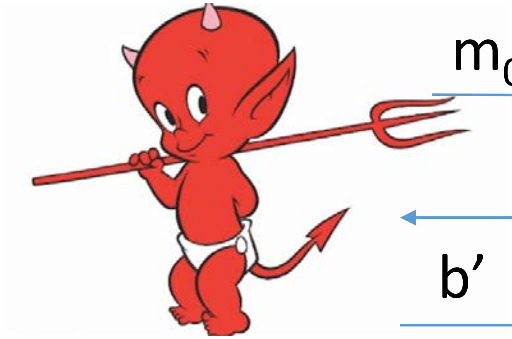
Random bit b
K ← Gen($\color{red}{1^n}$)
c ← Enc$_K$(m$_b$)

ppt attacker

negligible function

$\forall$ $\exists \mu$ $\Pr\Big[$ $Guesses\ b' = b\Big] \le \frac{1}{2} + \mu$ (n)

$\big(t(n), \varepsilon(n)\big)$-EAV-Secure (Concrete Version)



m_0, m_1 →

← c

b' →

Random bit b

K ← Gen(1^n)

c = Enc$_K$(m_b)

running in time at most $t(n)$

specific function

(same for all attackers)

$\forall$ $\Pr\Big[$ $Guesses\ b' = b\Big] \le \dfrac{1}{2} + \varepsilon(n)$

Aside: Message and Ciphertext Length

- In the previous game we typically require that $|m_0|=|m_1|$. Why?

- It is impossible to support arbitrary length messages while hiding all information about plaintext length

- **Limitation:** When could message length be sensitive?
  - Numeric data (5 figure vs 6 figure salary)
  - Database Searches: number of records returned can reveal information about the query
  - Compressed Data: Short compressed string indicates that original plaintext has a lot of redundancy (e.g., CRIME attack on session cookies in HTTPS)

Implications of Indistinguishability

i^th bit of message

Theorem 3.10: Let (Gen, Enc, Dec) be a fixed-length private-key encryption scheme for message of length ℓ that satisfies indistinguishability (prior definition) then for all PPT attackers A and any $\mathrm{i} \leq \ell$ we have

$$\Pr\left[A\left(1^n, \mathrm{Enc}_K(m)\right) = m^i\right] \leq \frac{1}{2} + \mathrm{negl}(n)$$

Where the randomness is taken over $\mathrm{K} \leftarrow \mathrm{Gen}(1^n)$, <u>uniform</u> $\mathrm{m} \in \{0,1\}^{\ell}$ and the randomness of Enc and A.

Remark: A bit weaker than saying eavesdropping attacker obtains ``no additional” information about message m.

Semantic Security

Definition 3.12: Let $\Pi = (\text{Gen, Enc, Dec})$be a fixed-length private key encryption scheme for message of length ℓ. We say that the scheme is semantically secure if for all PPT attackers A there exists a PPT algorithm A' such that for any PPT algorithm Sample all any polynomial time computable functions f and h we have

$$|\Pr[A(1^n, \text{Enc}_K(m), h(m)) = f(m)]$$

S

h(m) background knowledge the attacker might have about m.

A’ doesn’t even get to see an …yption of m! Just the length of m!

Definition 3.12: Let Π = (Ge… be a … ength private key encryption scheme for message of length … e scheme is semantically secure if for all PPT attackers A there e… a … gorithm A’ such that for any PPT algorithm Sample all any polyno… mputable functions f and h we have

$$\left|\Pr[A(1^n, \mathrm{Enc}_K(m), h(m)) = f(m)]\right.$$

Semantic Security

Definition 3.12: Let $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$ be a fixed-length private key encryption scheme for message of length ℓ. We say that the scheme is semantically secure if for all PPT attackers A there exists a PPT algorithm A' such that for any PPT algorithm Sample all any polynomial time computable functions f and h we have

$$|\Pr[A(1^n, \text{Enc}_K(m), h(m)) = f(m)]$$

Another Interpretation of Semantic Security

- World 2: Perfect Secrecy (Attacker doesn't even see ciphertext).
- For all attackers A' (even unbounded) with background knowledge h(m) we have
$$\Pr[A'(1^n,|m|,h(m)) = f(m)] = \Pr[f(m)| \quad h(m),|m|]$$

- World 1: Attacker is PPT and sees ciphertext
  - Best World 1 attacker does no better than World 2 attacker
- $|\Pr[A(1^n,\mathrm{Enc}_K(m),h(m)) = f(m)] - \Pr[A'(1^n,|m|,h(m)) = f(m)]| \leq \mathrm{negl}(n)$

- What is probability over?