Principles of Security — Part 3: Information Security and Cryptography

Peter-M. Seidel

January 18, 2017

Outline

Information, channel security, noninterference

Encryption and decryption

Cryptanalysis and notions of secrecy

Cyphers and modes of operation

Key establishment

What did we learn?

Outline

Information, channel security, noninterference
- Concepts of information and of information security
- Areas of information security
- Covert channels and Trojan horse
- Security models and noninterference

Encryption and decryption

Cryptanalysis and notions of secrecy

Cyphers and modes of operation

Key establishment

What did we learn?

Recall from Lecture 1

Information security

- **secrecy:** "bad *information flows* don't happen"
- **authenticity:** "good *information flows* do happen"

In network computation

- all information flow constraints are security properties

We could also say

Information security

- **confidentiality:** "bad *information flows* don't . . . "
- **integrity:** "good *information flows* do. . . "

Although not synonymous

- secrecy, confidentiality and privacy
- authenticity and integrity

are used interchanteably

Security speak

(overheard at a security conference)

Speaker: Isn't it terrifying that on the Internet we have no privacy?

Charlie: You mean *confidentiality*. Get your terms straight.

Radia: Why do security types insist on inventing their own language?

Mike: It's a denial-of-service attack.

Charlie: You mean *chosen cyphertext attack*...

Variants

(a possible assignment of meanings)

Bad information flows

- **secret information:** disclosure prevented
  - e.g., by cryptography
- **private information:** disclosure when authorized
  - information privately owned
- **confidential information:** disclosure restricted
  - penalized when detected

Variants

(a possible assignment of meanings)

Bad information flows about resources

- **secret funds:** it is secret that they exist
  - secret ceremony, secret lover. . .
- **private funds:** access is restricted
  - private ceremony, private resort. . .
- **confidential report:** some details confidential
  - content can be disclosed, but not the source

Variants

(a possible assignment of meanings)

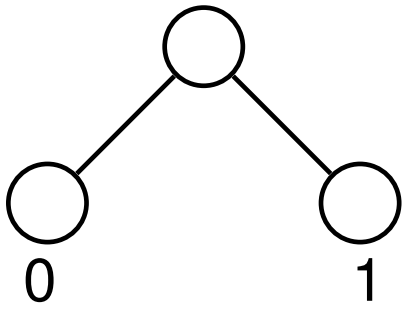
Good information flows

- **authenticity** of a painting, of a letter, of testimony
  - the source of the message is who it says it is
- **integrity** of evidence, of a person
  - the content of the message not been altered, tampered with, compromised

What is information?

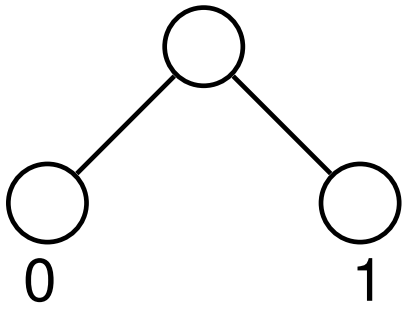
What is information?

Before a coin flip, the outcome is unknown.



What is information?

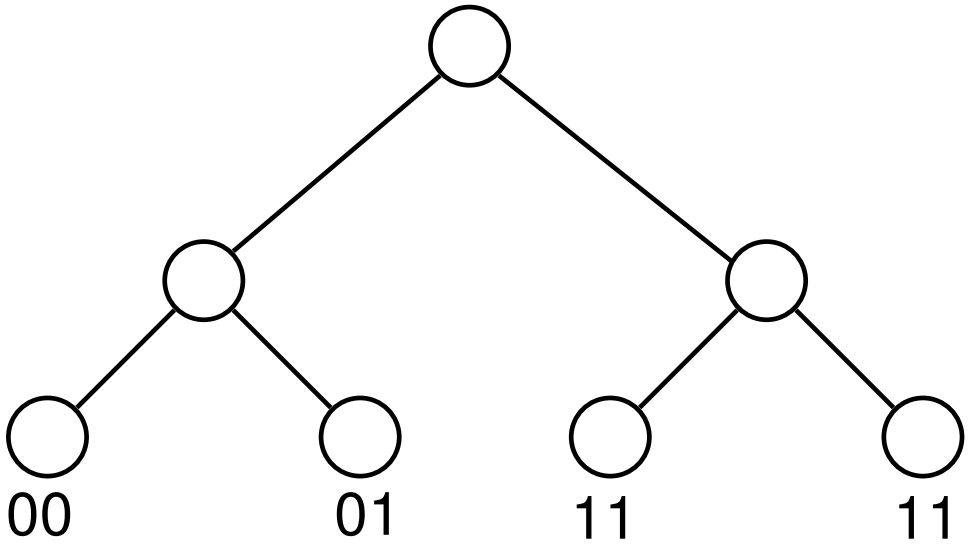
Before a coin flip, the outcome is unknown.



A coin flip yields exactly 1 bit of information.

What is information?

Before two coin flips, the outcome is even more unknown.



Two coin flips give exactly 2 bits of information.

What is information?

Rolling a fair 4-sided die gives the same amount of information like flipping 2 fair coins.

What is information?

Let's get formal (but don't take it too seriously yet).

What is information?

Let's get formal (but don't take it too seriously yet).

Definition

A *source* is a finite or countable set $\mathcal{X}$ given with a probability distribution.

What is information?

Let's get formal (but don't take it too seriously yet).

Definition

A *source* is a finite or countable set $\mathcal{X}$ given with a probability distribution.

A probability distribution over $\mathcal{X}$ is a just function $\mathrm{Prob}_{\mathcal{X}} : \mathcal{X} \longrightarrow [0, 1]$ such that

$$\sum_{x \in \mathcal{X}} \mathrm{Prob}(x) \quad = \quad 1$$

What is information?

Examples

- coin, two coins, dice. . .
  - What will be the outcome?

What is information?

Examples

- coin, two coins, dice. . .
  - What will be the outcome?
- language
  - What will be the next word that I'll say?

What is information?

Examples

- coin, two coins, dice. . .
  - What will be the outcome?
- language
  - What will be the next word that I'll say?
- any observable parameter
  - Who will be the next US president?

What is information?

Definition

Information is the average length of the binary words needed to express the outcome of sampling a source $\mathcal{X}$.

What is information?

Definition

Information is the average length of the binary words needed to express the outcome of sampling a source $\mathcal{X}$. It is denoted $H(\mathcal{X})$.

What is information?

Definition

Information is the average length of the binary words needed to express the outcome of sampling a source $\mathcal{X}$. It is denoted $H(\mathcal{X})$.

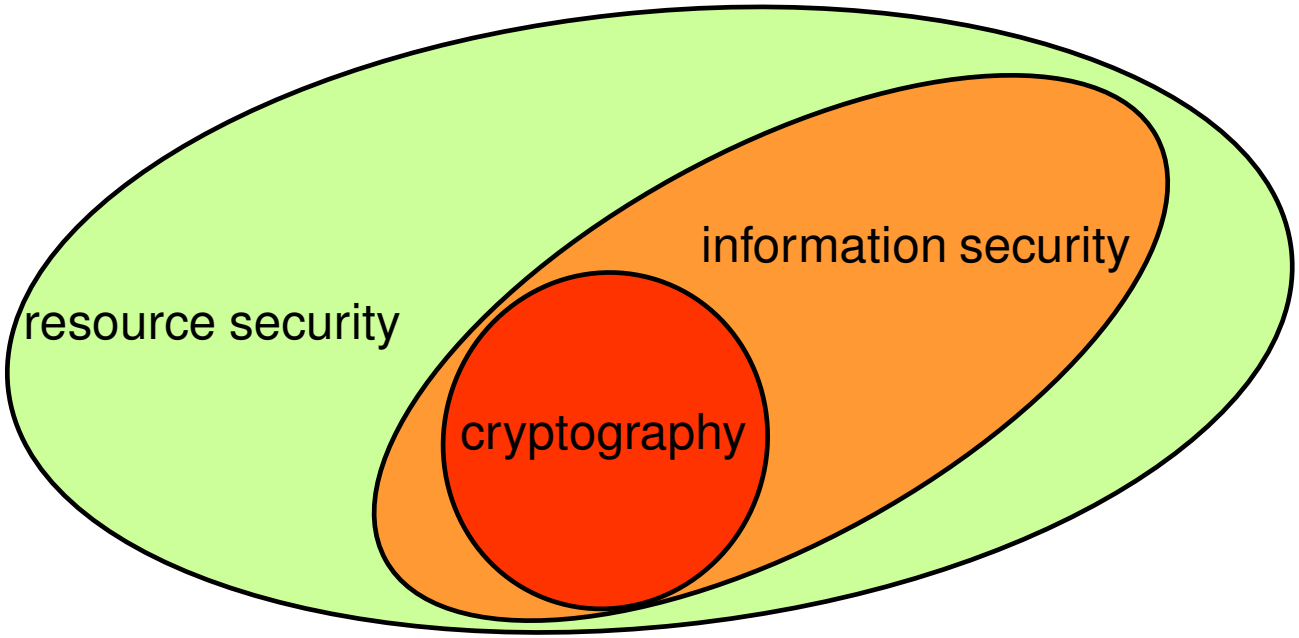
Examples

- $H(\text{coin}) = 1$
- $H(\text{2 coins}) = H(\text{4-sided die}) = 2$
- Biased coins and dice give less information.
- If the outcome of an experiment $\mathcal{X}$ is certain, then $H(\mathcal{X}) = 0$.

Areas of information security

Just like

- information is a special kind of a resource,
- a message is a special kind of information sample



Information gathering

Information can be acquired by

- observing accesses to resources
- receiving messages

Information gathering

Information can be acquired by

- observing accesses to resources
- receiving messages

Accordingly, we subdivide information security into:

- observation security, or *channels security*, and
- message security, or *cryptography*.

Observing confidential information

- Information flows through *channels*.

Observing confidential information

- Information flows through *channels*.
- Confidential information leaks through *covert* channels.

Trojan horse

is a covert channel installed through *social engineering*

Figure: A channel is concealed in a resource

Trojan horse

is a covert channel installed through *social engineering*

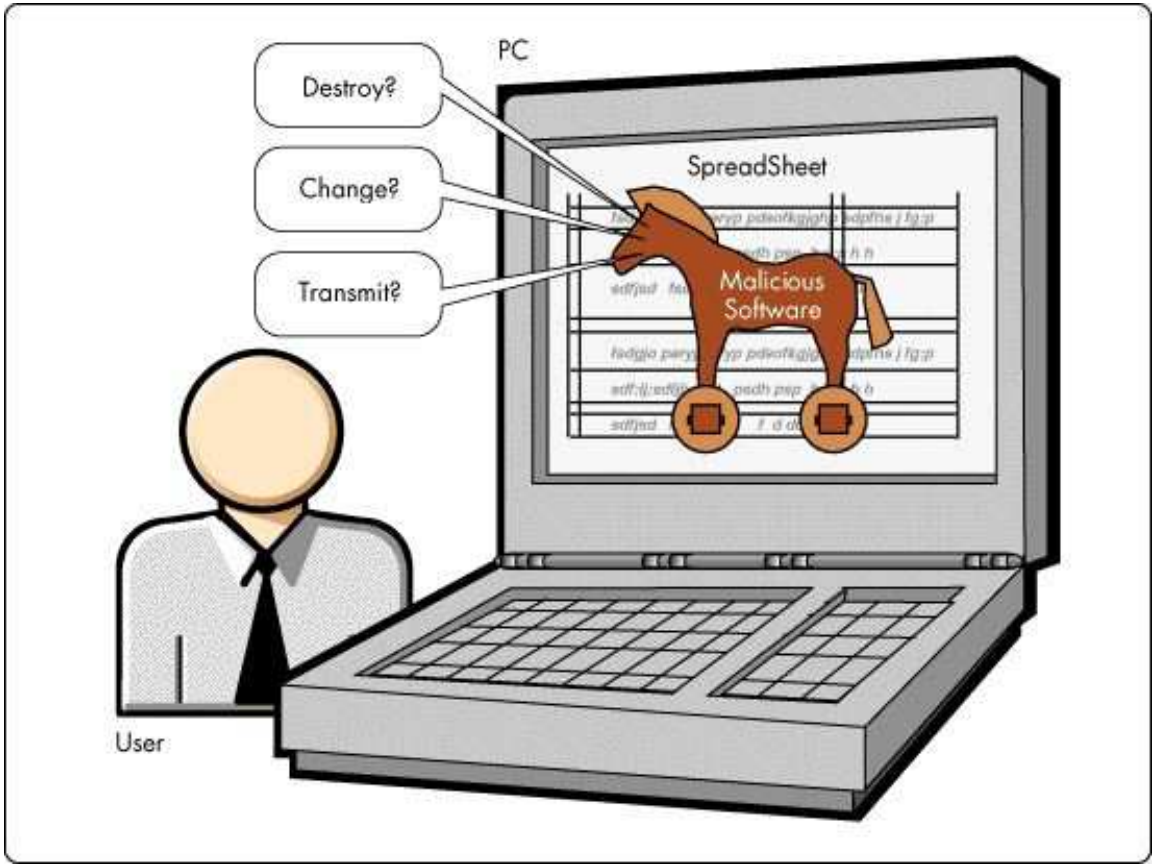


Figure: A channel is concealed in a resource.

State machines

Definition

A *state machine* is a map (pair of maps)

$$Q \times I \xrightarrow{\langle nx, ev \rangle} Q \times O$$

where Q, I, O are finite sets, representing

- Q — states
- I — input alphabet
- O — output alphabet
- $Q \times I \xrightarrow{nx} Q$ — next state
- $Q \times I \xrightarrow{ev} O$ — output eval.

State machines

Definition

A *state machine* is a map (pair of maps)

$$Q \times I \xrightarrow{\langle nx, ev \rangle} Q \times O$$

where Q, I, O are finite sets, representing

- Q — states
- I — input alphabet
- O — output alphabet
- $Q \times I \xrightarrow{nx} Q$ — next state
- $Q \times I \xrightarrow{ev} O$ — output eval.

Notation

A state machine is denoted by the name of its state set Q.

Running state machines

Inputs and outputs

The inputs and the outputs of state machines are lists from I and O.

For any set X, the set of lists

$$X^* = \left\{ \langle x_1, x_2, \ldots, x_n \rangle \in X^n \mid n \in \mathbb{N} \right\}$$

is generated from the empty list by prepending

$$\begin{array}{rcl} 1 & \xrightarrow{\langle\rangle} & X^* \\ X \times X^* & \xrightarrow{@} & X^* \end{array}$$

Running state machines

Inputs and outputs

The inputs and the outputs of state machines are lists from I and O.
For any set X, the set of lists in it

$$X^* = \left\{ \langle x_1, x_2, \ldots, x_n \rangle \in X^n \mid n \in \mathbb{N} \right\}$$

can be generated from the empty list by prepending

$$\begin{array}{rcl} 1 & \xrightarrow{\langle\rangle} & X^* \\ X \times X^* & \xrightarrow{@} & X^* \\ \left\langle x, \langle y_1, y_2 \ldots, y_n \rangle \right\rangle & \mapsto & \langle x, y_1, y_2 \ldots, y_n \rangle \end{array}$$

Running state machines

Input-output maps

At any state q, the state machine Q induces a map

$$I^* \xrightarrow{Ev^q} O^*$$

where

$$\begin{aligned} Ev^q\langle\rangle &= \langle\rangle \\ Ev^q(x@ys) &= ev^q(x) @ Ev^{nx^q(x)}(ys) \end{aligned}$$

for $x \in I$ and $ys \in I^*$

Multi level machines

Definition

A *multi level machine* is a map

$$Q \times I \xrightarrow{\langle nx, ev \rangle} Q \times O$$

where Q, I, O are finite sets, representing

- Q — states
- $I = \sum_{\ell \in \mathbb{L}} I_\ell$ — disjoint union of input alphabets
- O — output alphabet

Hi-Lo machines

Definition

A *Hi-Lo machine* is a map

$$Q \times I \xrightarrow{\langle nx, ev \rangle} Q \times O$$

where Q, I, O are finite sets, representing

- Q — states
- $I = I_H + I_L$ — disjoint union input alphabets
- O — output alphabet

Hi-Lo machines

Remark

A Hi-Lo-machine is just a multi level machine with just two levels $\mathbb{L} = \{L < H\}$.

Hi-Lo machines

Notation

The restriction (or purge) $(-)_L : I^* \longrightarrow I_L^*$ is defined

$$\begin{aligned} \langle\rangle_L &= \langle\rangle \\ (x@ys)_L &= \begin{cases} x@ys_L & \text{if } x \in I_L \\ ys_L & \text{otherwise} \end{cases} \end{aligned}$$

Hi-Lo machines

Notation

The restriction (or purge) $(-)_L : I^* \longrightarrow I_L^*$ is defined

$$\begin{aligned} \langle\rangle_L &= \langle\rangle \\ (x@ys)_L &= \begin{cases} x@ys_L & \text{if } x \in I_L \\ ys_L & \text{otherwise} \end{cases} \end{aligned}$$

The outputs of Lo's actions are:

$$\begin{aligned} Ev_L^q\langle\rangle &= \langle\rangle \\ Ev_L^q(x@ys) &= \begin{cases} ev^q(x) \mathbin{@} Ev_L^{nx^q(x)}(ys) & \text{if } x \in I_L \\ Ev_L^{nx^q(x)}(ys) & \text{otherwise} \end{cases} \end{aligned}$$

Covert channels and Trojans

Definition

We say that the Hi-Lo machine Q has a *covert channel* if it has a state q such that

- $xs_L = ys_L$, but
- $Ev_L^q(xs) \neq Ev_L^q(ys)$

holds for some input lists $xs, ys \in I^*$.

Covert channels and Trojans

Definition

We say that the Hi-Lo machine Q has a *covert channel* if it has a state q such that

- $xs_L = ys_L$, but
- $Ev_L^q(xs) \neq Ev_L^q(ys)$

holds for some input lists $xs, ys \in I^*$.
The subject Hi in a Hi-Lo machine with a covert channel is often called a *Trojan (horse)*.

Covert channels and Trojans

Homework

Specify a simple Hi-Lo machine with a covert channel.

Noninterference

(Goguen-Meseguer)

Definition

We say that the Hi-Lo machine Q satisfies the *noninterference* requirement if it has no covert channels, i.e.

$$xs_L = ys_L \implies Ev_L^q(xs) = Ev_L^q(ys)$$

holds for all states q and all inputs $xs, ys \in I^*$.

Noninterference

(Goguen-Meseguer)

Remark

The no-write-down condition

- prevents Hi from sending to Lo
- any publicly visible signals (messages).

Noninterference

(Goguen-Meseguer)

Remark

The no-write-down condition

- prevents Hi from sending to Lo
- any publicly visible signals (messages).

The noninterference condition

- prevents Hi from sending to Lo
- any *secret signals*.

Generalized noninterference

(McCullough, McLean)

Definition

We say that the Hi-Lo machine Q satisfies the *generalized noninterference* requirement if

$$\forall xs\ zs \in I^* \exists ys \in I^*.\ \ xs_L = ys_L \ \wedge\ ys_H = zs_H \\ \wedge\ Ev_L^q(xs) = Ev_L^q(ys)$$

holds for all states q.

Generalized noninterference

(McCullough, McLean)

Homework

Prove that generalized noninterference and noniterference are equivalent for deterministic machines

Remark

Generalized noninterference is also applicable to nondeterministic machines.

Outline

Simple crypto system

Definition

Given the types

- $\mathcal{M}$ of *plaintexts*
- $\mathcal{C}$ of *cyphertexts*
- $\mathcal{K}$ of *keys*

Simple crypto system

Definition

. . . a simple crypto-system is a triple of algorithms:

- key generation $\langle K_E, K_D \rangle : \mathcal{K} \times \mathcal{K}$,
- encryption $E : \mathcal{K} \times \mathcal{M} \longrightarrow \mathcal{C}$, and
- decryption $D : \mathcal{K} \times \mathcal{C} \longrightarrow \mathcal{M}$,

Simple crypto system

Definition

. . . that together provide

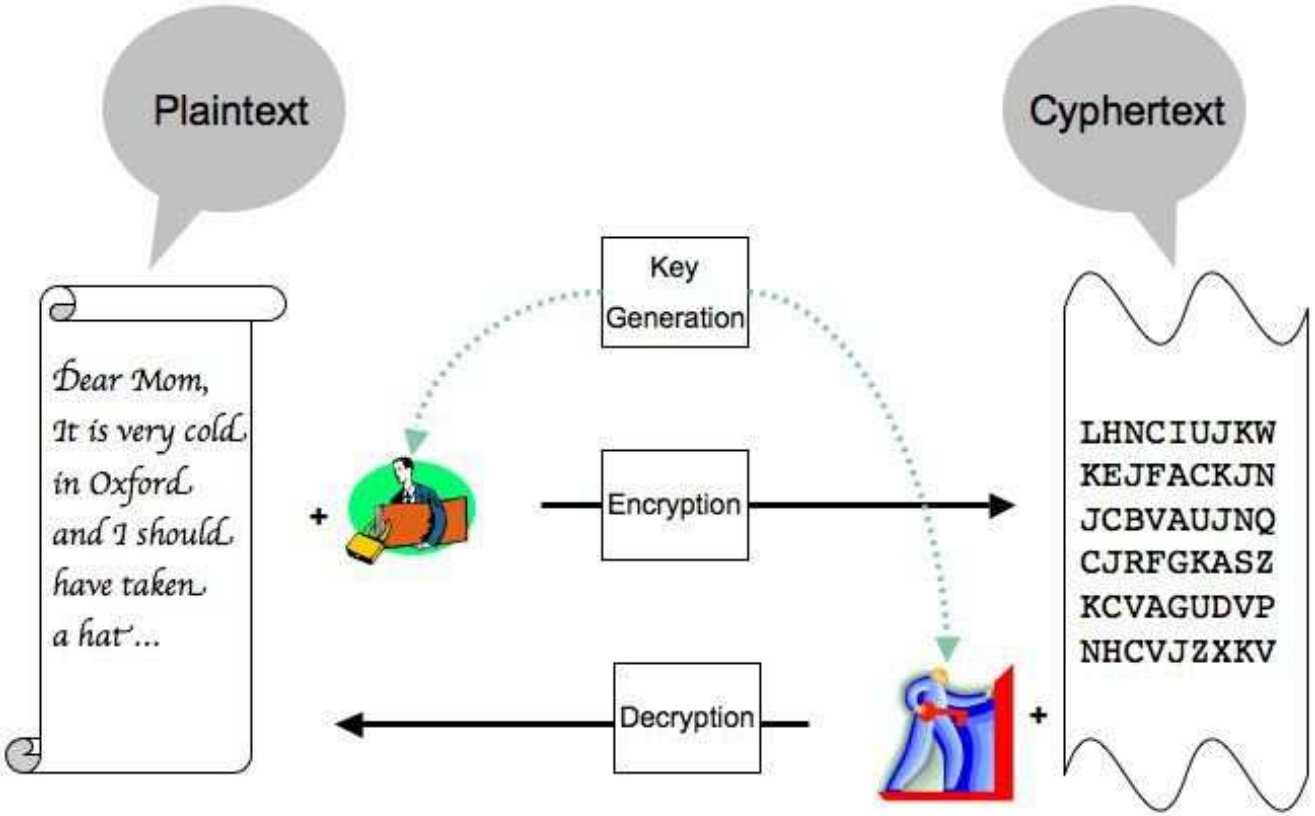
- unique decryption:

$$\mathrm{D}(\mathrm{K_D}, \mathrm{E}(\mathrm{K_E}, m)) \quad = \quad m$$

- trapdoor encryption:

$$\begin{aligned} \forall \mathrm{A} : \mathcal{C} \longrightarrow \mathcal{M}.\ & \left(\forall m.\ \mathrm{A}(\mathrm{E}(\mathrm{K_E}, m)) = m\right) \\ \Longrightarrow\ & \left(\forall c.\ \ \mathrm{A}(c) \qquad\quad = \mathrm{D}(\mathrm{K_D}, c)\right) \end{aligned}$$

Using a cryptosystem



What where do the plaintexts come from?

Remarks

- The space $\mathcal{M}$ may be
  - monoalphabetic: it consists of symbols
    - $\mathcal{M} = \Sigma$
  - polyalphabetic: it consists of blocks of symbols
    - $\mathcal{M} = \Sigma^N$

What where do the plaintexts come from?

Remarks

- The space $\mathcal{M}$ may be
  - monoalphabetic: it consists of symbols
    - $\mathcal{M} = \Sigma$
  - polyalphabetic: it consists of blocks of symbols
    - $\mathcal{M} = \Sigma^N$
- A plaintext is a string from $\mathcal{M}$.

What where do the plaintexts come from?

Remarks

- The space $\mathcal{M}$ may be
  - monoalphabetic: it consists of symbols
    - $\mathcal{M} = \Sigma$
  - polyalphabetic: it consists of blocks of symbols
    - $\mathcal{M} = \Sigma^N$
- A plaintext is a string from $\mathcal{M}$.
- A well-formed message is a *meaningful* plaintext: a word, a sentence, a paragraph.

What where do the plaintexts come from?

Remarks

- The space $\mathcal{M}$ may be
  - monoalphabetic: it consists of symbols
    - $\mathcal{M} = \Sigma$
  - polyalphabetic: it consists of blocks of symbols
    - $\mathcal{M} = \Sigma^N$
- A plaintext is a string from $\mathcal{M}$.
- A well-formed message is a *meaningful* plaintext: a word, a sentence, a paragraph.
- Not every plaintext is a well-formed message.

What shall we study?

- Cryptography: science of crypto systems
  - Cryptology: designing crypto systems
    - to encrypt plaintexts as cyphertexts
    - so that only those with a key can decrypt them
  - Cryptanalysis: breaking crypto systems
    - to extract the plaintexts without a key
    - or even better, to extract the key

Examples

Encode letters as numbers

a	b	c	c	e	f	g	h	i	j	k	l	m
0	1	2	3	4	5	6	7	8	9	10	11	12
n	o	p	q	r	s	t	u	v	w	x	y	z
13	14	15	16	17	18	19	20	21	22	23	24	25

Example 1.1: Shift cypher

(monoalphabetic: Cæsar $k = 3$, ROT13 $k = 13\ldots$)

- $\mathcal{M} = \mathcal{C} = \mathbb{Z}_{26} = \{0, 1, 2, 3, \ldots, 25\}$
- $\mathcal{K} = \mathbb{Z}_{26}$
- $\mathsf{K}_\mathsf{E} = \mathsf{K}_\mathsf{D} = k$
- $\mathsf{E}(k, m) = m + k \mod 26$
- $\mathsf{D}(k, c) = c - k \mod 26$

Example 1.1: Shift cypher

(monoalphabetic: Cæsar $k = 3$, ROT13 $k = 13\ldots$)

E.g., the key $k = 5$ gives

tx:	i	t	i	s	v	e	r	y	c	o	l	d
$\vec{m}$	8	19	8	18	21	4	17	24	2	14	11	3
k	5	5	5	5	5	5	5	5	5	5	5	5
$\vec{c}$	13	24	13	23	0	9	22	3	7	19	16	8
CY:	N	Y	N	X	A	J	W	D	H	T	Q	I

where

a 0	b 1	c 2	d 3	e 4	f 5	g 6	h 7	i 8	j 9	k 10	l 11	m 12
n 13	o 14	p 15	q 16	r 17	s 18	t 19	u 20	v 21	w 22	x 23	y 24	z 25

Example 1.2: Shift cypher

(polyalphabetic)

- $\mathcal{M} = \mathcal{C} = \mathbb{Z}_{26}^N$
- $\mathcal{K} = \mathbb{Z}_{26}^N$
- $\mathsf{K_E} = \mathsf{K_D} = \vec{k} = \langle k_1, k_2, \ldots, k_N \rangle$
- $\mathsf{E}(\vec{k}, \vec{m}) = \vec{m} + \vec{k} \mod 26$
- $\mathsf{D}(\vec{k}, \vec{c}) = \vec{c} - \vec{k} \mod 26$

Example 1.2: Shift cypher

(polyalphabetic)

E.g., the block length $N = 6$ and the keyword kd="monkey" give

tx:	i	t	i	s	v	e	r	y	c	o	l	d
$\vec{m}$	8	19	8	18	21	4	17	24	2	14	11	3
kd:	m	o	n	k	e	y	m	o	n	k	e	y
$\vec{k}$	12	14	13	10	4	24	12	14	13	10	4	24
$\vec{c}$	20	7	21	2	25	2	3	12	15	24	15	1
CY:	U	H	V	C	Z	B	C	M	P	Y	P	B

where

a	b	c	c	e	f	g	h	i	j	k	l	m
0	1	2	3	4	5	6	7	8	9	10	11	12
n	o	p	q	r	s	t	u	v	w	x	y	z
13	14	15	16	17	18	19	20	21	22	23	24	25

Example 1.2: Shift cypher

(polyalphabetic)

Terminology

A polyalphabetic shift cypher where

- each key $K \in \mathbb{Z}_{26}^N$ is used to encrypt
- a single message $\vec{m} \in \mathbb{Z}_{26}^N$

is called a *one-time-pad*. It is

- perfectly secure, but it reduces
- the task to transfer an N-character message to
- the task to transfer an N-character key.

Example 1.2: Shift cypher

(polyalphabetic)

Fact

A polyalphabetic shift cypher where

- a key $K \in \mathbb{Z}_{26}^N$ is used to encrypt
- more than one $\vec{m}_1, \vec{m}_2 \ldots \in \mathbb{Z}_{26}^N$

is insecure.

Example 1.2: Shift cypher

(polyalphabetic)

Fact

A polyalphabetic shift cypher where

- a key $K \in \mathbb{Z}_{26}^N$ is used to encrypt
- more than one $\vec{m}_1, \vec{m}_2 \ldots \in \mathbb{Z}_{26}^N$

is insecure.

We shall prove this.

Example 1.2: Shift cypher

(polyalphabetic)

Terminology vs history

Polyalphabetic shift cyphers are often called *Vigenère's cyphers*.

This is a sad confusion. Vigenère had nothing to do with polyalphabetic shift cyphers.

He designed the first auto-keying cypher.

Example 1.3: Affine cypher

(polyalphabetic)

- $\mathcal{M} = \mathcal{C} = \mathbb{Z}_{26}^N$,
- $\mathcal{K} = \left(\mathbb{Z}_{26}^*\right)^N \times \mathbb{Z}_{26}^N$
- $\mathsf{K}_\mathsf{E} = \mathsf{K}_\mathsf{D} = \left\langle \vec{a}, \vec{k} \right\rangle$
- $\mathsf{E}(\vec{a}, \vec{k}, \vec{m}) = \vec{a} * \vec{m} + \vec{k} \mod 26$
- $\mathsf{D}(\vec{a}, \vec{k}, \vec{c}) = \frac{1}{\vec{a}} * (\vec{c} - \vec{k}) \mod 26$

where

$$\begin{aligned} \vec{a} * \vec{m} &= \langle a_1 m_1, a_2 m_2, \ldots, a_n m_N \rangle \\ \frac{1}{\vec{a}} &= \left\langle \frac{1}{a_1}, \frac{1}{a_2}, \ldots, \frac{1}{a_N} \right\rangle \end{aligned}$$

Example 1.4: Substitition cypher

(monoalphabetic)

- $\mathcal{M} = \mathcal{C} = \Sigma = \{a, b, c, \ldots, z\}$,
- $\mathcal{K} = S(\Sigma) =$ the permutations of Σ
- $\mathsf{K}_\mathsf{E} = \mathsf{K}_\mathsf{D} = \sigma$
- $\mathsf{E}(\sigma, m) = \sigma(m)$
- $\mathsf{D}(\sigma, c) = \sigma^{-1}(c)$

Example 1.5: Substitition cypher

(polyalphabetic)

- $\mathcal{M} = \mathcal{C} = \Sigma^N$,
- $\mathcal{K} = S(\Sigma)$, the permutations of Σ
- $\mathsf{K}_\mathsf{E} = \mathsf{K}_\mathsf{D} = \sigma$
- $\mathsf{E}(\sigma, \vec{m}) = \langle \sigma(m_1), \sigma(m_2), \ldots \sigma(m_n) \rangle$
- $\mathsf{D}(\sigma, \vec{c}) = \left\langle \sigma^{-1}(c_1), \sigma^{-1}(c_2), \ldots \sigma^{-1}(c_n) \right\rangle$

where $N = \{1, 2, \ldots, n\}$

Example 2: Transposition cypher

- $\mathcal{M} = \mathcal{C} = N^N$,
- $\mathcal{K} = S(N) =$ the permutations of the block positions
- $\mathsf{K}_\mathsf{E} = \mathsf{K}_\mathsf{D} = \sigma$
- $\mathsf{E}(\sigma, \vec{m}) = \left\langle m_{\sigma(1)}, m_{\sigma(2)}, \ldots m_{\sigma(n)} \right\rangle$
- $\mathsf{D}(\sigma, \vec{c}) = \left\langle m_{\sigma^{-1}(1)}, m_{\sigma^{-1}(2)}, \ldots m_{\sigma^{-1}(n)} \right\rangle$

Example 3: RSA

- $\mathcal{M} = \mathcal{C} = \mathbb{Z}_n$, where $n = pq$, p, q prime
- $\mathcal{K} = \mathbb{Z}_{\varphi(n)}$, where $\varphi(n) = \#\{k < n \mid \gcd(n, k) = 1\}$
- $\mathrm{K_E} = e$
- $\mathrm{K_D} = e^{-1} \mod \varphi(n)$
- $\mathrm{E}(e, m) = m^e \mod n$
- $\mathrm{D}(d, c) = c^d \mod n$

Example 3: RSA

- $\mathcal{M} = \mathcal{C} = \mathbb{Z}_n$, where $n = pq$, p, q prime
- $\mathcal{K} = \mathbb{Z}_{\varphi(n)}$, where $\varphi(n) = \#\{k < n \mid \gcd(n,k) = 1\}$
- $\mathrm{K_E} = e$ ⇝ **public** key
- $\mathrm{K_D} = e^{-1} \mod \varphi(n)$ ⇝ **private** key
- $\mathrm{E}(e, m) = m^e \mod n$
- $\mathrm{D}(d, c) = c^d \mod n$

Example 3: RSA

Idea of public key cryptography

- K_E is publicly announced
  - eveyone can encrypt
- K_D is kept secret
  - only those who have it can decrypt

Example 3: RSA

Idea of public key cryptography

- K_E is publicly announced
  - eveyone can encrypt
- K_D is kept secret
  - only those who have it can decrypt

It is important that K_D cannot be derived from K_E.

Example 3: RSA

History of public key cryptography

- Whit Diffie and Marty Hellman proposed computational hardness as a new foundation for cryptography in **1976**.
- Ron Rivest, Adi Shamir and Len Adleman (RSA) implemented that idea using exponentiation in **1978**.

Example 3: RSA

History of public key cryptography

- Whit Diffie and Marty Hellman proposed computational hardness as a new foundation for cryptography in **1976**.
- Ron Rivest, Adi Shamir and Len Adleman (RSA) implemented that idea using exponentiation in **1978**.
- The RSA patent became a base of a very profitable company.

Example 3: RSA

History of public key cryptography

- Whit Diffie and Marty Hellman proposed computational hardness as a new foundation for cryptography in **1976**.
- Ron Rivest, Adi Shamir and Len Adleman (RSA) implemented that idea using exponentiation in **1978**.
- The RSA patent became a base of a very profitable company. All involved became rich and famous.

Example 3: RSA

History of public key cryptography

- In December 1997, the British Government Communications Headquarters (GCHQ) released five papers.

Example 3: RSA

History of public key cryptography

- In December 1997, the British Government Communications Headquarters (GCHQ) released five papers.
- James Ellis' paper *"The possibility of non-secret encryption"* proposed computational hardness as a foundation for cryptography.
- Clifford Cocks' paper *"A note on non-secret encryption"* implemented that idea using exponentiation.

Example 3: RSA

History of public key cryptography

- In December 1997, the British Government Communications Headquarters (GCHQ) released five papers.
- James Ellis' paper *"The possibility of non-secret encryption"* proposed computational hardness as a foundation for cryptography. ⇜ **1970**
- Clifford Cocks' paper *"A note on non-secret encryption"* implemented that idea using exponentiation. ⇜ **1973**

Example 3: RSA

History of public key cryptography

- James Ellis retired in 1986 and died in November 1997.
- Clifford Cocks became the Chief Mathematician at GCHQ in 2007.

Example 3: RSA

History of public key cryptography

- James Ellis retired in 1986 and died in November 1997.
- Clifford Cocks became the Chief Mathematician at GCHQ in 2007.
- Public key cryptography was critical in arm treaty control as of 1986, but was not deployed earlier.

Example 3: RSA

- Take $p = 11$ and $q = 17$.

Example 3: RSA

- Take $p = 11$ and $q = 17$. Hence
  - $n = pq = 187$,

Example 3: RSA

- Take $p = 11$ and $q = 17$. Hence
  - $n = pq = 187$, and
  - $\varphi(n) = (11 - 1)(17 - 1) = 160$

Example 3: RSA

- Take $p = 11$ and $q = 17$. Hence
  - $n = pq = 187$, and
  - $\varphi(n) = (11 - 1)(17 - 1) = 160$
- Take $K_E = e = 3$

Example 3: RSA

- Take $p = 11$ and $q = 17$. Hence
  - $n = pq = 187$, and
  - $\varphi(n) = (11 - 1)(17 - 1) = 160$
- Take $K_E = e = 3$
- Then $K_D = d = 3^{-1} = 107 \mod 160$

Example 3: RSA

- Take $p = 11$ and $q = 17$. Hence
  - $n = pq = 187$, and
  - $\varphi(n) = (11 - 1)(17 - 1) = 160$
- Take $K_E = e = 3$
- Then $K_D = d = 3^{-1} = 107 \mod 160$
- $E(3, p) = J$ because
  - $E(3, 15) = 15^3 = 3375 = 9 \mod 187$

Example 3: RSA

- Take $p = 11$ and $q = 17$. Hence
  - $n = pq = 187$, and
  - $\varphi(n) = (11 - 1)(17 - 1) = 160$
- Take $K_E = e = 3$
- Then $K_D = d = 3^{-1} = 107 \mod 160$
- $E(3, p) = J$ because
  - $E(3, 15) = 15^3 = 3375 = 9 \mod 187$
- $D(107, J) = p$ because
  - $D(107, 9) = 9^{107} = 15 \mod 187$

Example 3: RSA

Homework

Prove that *Euler's totient function*

$$\begin{array}{rcl} \varphi : \mathbb{N} & \longrightarrow & \mathbb{N} \\ n & \longmapsto & \#\{k < n \mid \gcd(n,k) = 1\} \end{array}$$

has the following properties:

- $\varphi(p^k) = (p-1)p^{k-1}$ if p is prime
- $\varphi(mn) = \varphi(m) \cdot \varphi(n)$ if $\gcd(m,n) = 1$

Derive a general formula to compute $\varphi(n)$.

Example 3: RSA

. . . is a crypto system because

- **unique decryption** holds by

$$ed = 1 \mod \varphi(n) \implies (m^e)^d = m \mod n$$

Example 3: RSA

. . . is a crypto system because

- **unique decryption** holds by

$$ed = 1 \mod \varphi(n) \implies (m^e)^d = m \mod n$$

- **trapdoor encryption** holds since for every A

$$\forall m.\mathrm{A}(m^e) = m \mod n \implies \forall c.\mathrm{A}(c) = c^d \mod n$$

where $ed = 1 \mod \varphi(n)$

Example 3: RSA

To prove that the RSA satisfies these requirements, we need some basic arithmetic.

Refresher in arithmetic

Definition

Let $(G, \cdot, 1)$ be a finite group and $g \in G$. We define

$$\begin{aligned} \mathrm{ord}(G) &= \#G \text{ (the number of elements)} \\ \mathrm{ord}(g) &= \#\langle g \rangle = \min\{\ell \mid g^\ell = 1\} \end{aligned}$$

Theorem (Lagrange)

For every $g \in G$ holds $\mathrm{ord}(g) \mid \mathrm{ord}(G)$.

Refresher in arithmetic

Definition

The multiplicative group of *invertible* elements of $\mathbb{Z}_n$ is

$$\mathbb{Z}_n^* \quad = \quad \{x \in \mathbb{Z}_n \mid \exists y.\ xy = 1 \mod n\}$$

Lemma

$k \in \mathbb{Z}_n$ is invertible iff it is mutually prime with n, i.e.

$$k \in \mathbb{Z}_n^* \quad \Longleftrightarrow \quad \gcd(n, k) = 1$$

Hence $\mathrm{ord}(\mathbb{Z}_n^*) = \#\{k < n \mid \gcd(n, k) = 1\} = \varphi(n)$.

Refresher in arithmetic

Corollary (Euler)

For every invertible $k \in \mathbb{Z}_n^*$ holds

$$k^{\varphi(n)} \quad = \quad 1 \mod n$$

Refresher in arithmetic

Corollary (Euler)

For every invertible $k \in \mathbb{Z}_n^*$ holds

$$k^{\varphi(n)} \quad = \quad 1 \mod n$$

Proof.

By the Theorem, $\text{ord}(k) \mid \text{ord}(\mathbb{Z}_n^*)$.
By the Lemma, $\text{ord}(\mathbb{Z}_n^*) = \varphi(n)$. □

RSA unique decryption

Conclusion

Hence the **unique decryption** property of RSA

$$ed = 1 \mod \varphi(n) \iff \exists \ell.\ ed = 1 + \ell\varphi(n)$$
$$\implies m^{ed} = m^{1+\ell\varphi(n)} = m \mod n$$

RSA Assumption

RSA Problem

- input:
  - $n = pq \in \mathbb{N}$ where p and q are prime
  - $c \in \mathbb{Z}_n^*$, i.e. $\gcd(c, n) = 1$
  - $e \in \mathbb{Z}_{\varphi(n)}$, i.e. $\gcd(e, p-1) = \gcd(e, q-1) = 1$
- output:
  - $m = \sqrt[e]{c} \mod n$, i.e. $m^e = c \mod n$

RSA Assumption

RSA Problem

- input:
  - $n = pq \in \mathbb{N}$ where p and q are prime
  - $c \in \mathbb{Z}_n^*$, i.e. $\gcd(c, n) = 1$
  - $e \in \mathbb{Z}_{\varphi(n)}$, i.e. $\gcd(e, p-1) = \gcd(e, q-1) = 1$
- output:
  - $m = \sqrt[e]{c} \mod n$, i.e. $m^e = c \mod n$

RSA Assumption

There is no feasible algorithm solving the RSA Problem.

RSA trapdoor encryption

Conclusion

Hence the **trapdoor encryption** property of RSA

$$\forall m.\mathrm{A}(m^e) = m \mod n \implies \forall c.\mathrm{A}(c) = c^d \mod n$$

where $ed = 1 \mod \varphi(n)$

RSA trapdoor encryption

Remark

RSA problem can be solved by finding $d = e^{-1} \mod \varphi(n)$
i.e. by finding d, ℓ such that $de + \ell\varphi(n) = 1$.

RSA trapdoor encryption

Remark

RSA problem can be solved by finding $d = e^{-1} \mod \varphi(n)$
i.e. by finding d, ℓ such that $de + \ell\varphi(n) = 1$.
But computing $\varphi(n)$ requires factoring n.

RSA trapdoor encryption

Remark

RSA problem can be solved by finding $d = e^{-1} \mod \varphi(n)$
i.e. by finding d, ℓ such that $de + \ell\varphi(n) = 1$.
But computing $\varphi(n)$ requires factoring n.
It is believed that factoring is not feasible:
if n has only large factors, they are hard to find.

Coding

Definition

A *coding scheme* is an injective function $f : \mathcal{X} \longrightarrow \mathcal{G}$, where

- $\mathcal{X}$ is a source, and
- $\mathcal{G} \subset \Sigma^*$ is a language (or code).

Examples of coding

- Morse code:
  - *source*: characters
  - *code:* strings of dots and dashes
- telegraphic codes:

source	CODE
answer my question!	LYOUI
are you trying to weasel out?	BYOXO
you are a skunk!	BMULD
not clearly coded, please repeat	AYYLU

- English, Chinese. . . :
  - *source*: meaningful phrases
  - *code*: orthography

Coding vs encryption

Terminology

The elements $\gamma \in \mathcal{G} \subseteq \Sigma^*$ are called *codewords*.
Codewords are used as *well-formed* messages.

Coding vs encryption

Terminology

The elements $\gamma \in \mathcal{G} \subseteq \Sigma^*$ are called *codewords*.
Codewords are used as *well-formed* messages.

Remark

We usually take $\mathcal{M} = \Sigma$.
Any string of plaintexts $\vec{m} \in \Sigma^*$ can be a message. (E.g., meaningful words and meaningless strings.)

Coding vs encryption

Terminology

The elements $\gamma \in \mathcal{G} \subseteq \Sigma^*$ are called *codewords*.
Codewords are used as *well-formed* messages.

Remark

We usually take $\mathcal{M} = \Sigma$.
Any string of plaintexts $\vec{m} \in \Sigma^*$ can be a message. (E.g., meaningful words and meaningless strings.)
Not every message is a codeword.

Coding vs encryption

Terminology

The elements $\gamma \in \mathcal{G} \subseteq \Sigma^*$ are called *codewords*.
Codewords are used as *well-formed* messages.

Remark

We usually take $\mathcal{M} = \Sigma$.
Any string of plaintexts $\vec{m} \in \Sigma^*$ can be a message. (E.g., meaningful words and meaningless strings.)
Not every message is a codeword.
Those that are are said to be *well-formed*.

Coding vs encryption

Upshot

The difference between

- decryption $\mathcal{C} \xrightarrow{\mathsf{D}} \mathcal{M}$
- decoding $\mathcal{M}^* \hookleftarrow \mathcal{G}$

will play an important role in *cryptanalysis*.

Outline

Cryptanalytic attacks

Symmetric key attacks

When $K_E = K_D = K$, the attacks are

- cyphertext only (COA):

$$E(K, m_1), \ldots, E(K, m_\ell) \quad \vdash \quad K$$

- known plaintext (KPA), chosen plaintext (CPA):

$$m_1, \ldots, m_\ell, E(K, m_1), \ldots, E(K, m_\ell) \quad \vdash \quad K$$

- chosen cyphertext (CCA):

$$c_1, \ldots, c_\ell, D(K, c_1), \ldots, D(K, c_\ell) \quad \vdash \quad K$$

Cryptanalytic attacks

Asymmetric key attacks

When K_E is publicly known

- cyphertext only (COA):

$$K_E, E(K_E, m_1), \ldots, E(K_E, m_\ell) \quad \vdash \quad K_D$$

- known plaintext (KPA), chosen plaintext (CPA):

$$K_E, m_1, \ldots, m_\ell, E(K_E, m_1), \ldots, E(K_E, m_\ell) \quad \vdash \quad K_D$$

- chosen cyphertext (CCA):

$$K_E, c_1, \ldots, c_\ell, D(K_D, c_1), \ldots, D(K_D, c_\ell) \quad \vdash \quad K_D$$

- adaptive chosen cyphertext (CCA2): ...(later!)

COA on monoalphabetic shift cypher

- $\mathcal{M} = \mathcal{C} = \mathbb{Z}_{26}$
- $\mathcal{K} = \mathbb{Z}_{26}$
- $\mathsf{K}_\mathsf{E} = \mathsf{K}_\mathsf{D} = k$
- $\mathsf{E}(k, m) = m + k \mod 26$
- $\mathsf{D}(k, c) = c - k \mod 26$

COA on monoalphabetic shift cypher

- $\mathcal{M} = C = \mathbb{Z}_{26}$
- $\mathcal{K} = \mathbb{Z}_{26}$
- $\mathsf{K}_\mathsf{E} = \mathsf{K}_\mathsf{D} = k$
- $\mathsf{E}(k, m) = m + k \mod 26$
- $\mathsf{D}(k, c) = c - k \mod 26$

Idea

Since there are just $\#\mathcal{K} = 26$ possible keys, simply try one after the other.

COA on monoalphabetic shift cypher

CY:	N	Y	N	X	A	J	W	D	H	T	Q	I
$\vec{c}$	13	24	13	23	0	9	22	3	7	19	16	8
k_1	1	1	1	1	1	1	1	1	1	1	1	1
$\vec{m}_1$	12	23	12	22	25	8	21	2	6	18	15	7
tx_1:	m	x	m	w	z	i	v	c	g	s	p	h

COA on monoalphabetic shift cypher

CY:	N	Y	N	X	A	J	W	D	H	T	Q	I
$\vec{c}$	13	24	13	23	0	9	22	3	7	19	16	8
k_2	2	2	2	2	2	2	2	2	2	2	2	2
$\vec{m}_2$	11	22	11	21	24	7	20	1	5	17	14	6
tx_2:	l	w	l	v	y	h	u	b	f	r	o	g

COA on monoalphabetic shift cypher

CY:	N	Y	N	X	A	J	W	D	H	T	Q	I
$\vec{c}$	13	24	13	23	0	9	22	3	7	19	16	8
k_5	5	5	5	5	5	5	5	5	5	5	5	5
$\vec{m}_5$	8	19	8	18	21	4	17	24	2	14	11	3
tx_5:	i	t	i	s	v	e	r	y	c	o	l	d

COA on substitution cypher

- $\mathcal{M} = \mathcal{C} = \Sigma = \{a, b, c, \ldots, z\}$,
- $\mathcal{K} = S(\Sigma) =$ the permutations of Σ
- $\mathsf{K}_\mathsf{E} = \mathsf{K}_\mathsf{D} = \sigma$
- $\mathsf{E}(\sigma, m) = \sigma(m)$
- $\mathsf{D}(\sigma, c) = \sigma^{-1}(c)$

COA on substitution cypher

- $\mathcal{M} = \mathcal{C} = \Sigma = \{a, b, c, \ldots, z\}$,
- $\mathcal{K} = S(\Sigma) =$ the permutations of Σ
- $\mathsf{K}_\mathsf{E} = \mathsf{K}_\mathsf{D} = \sigma$
- $\mathsf{E}(\sigma, m) = \sigma(m)$
- $\mathsf{D}(\sigma, c) = \sigma^{-1}(c)$

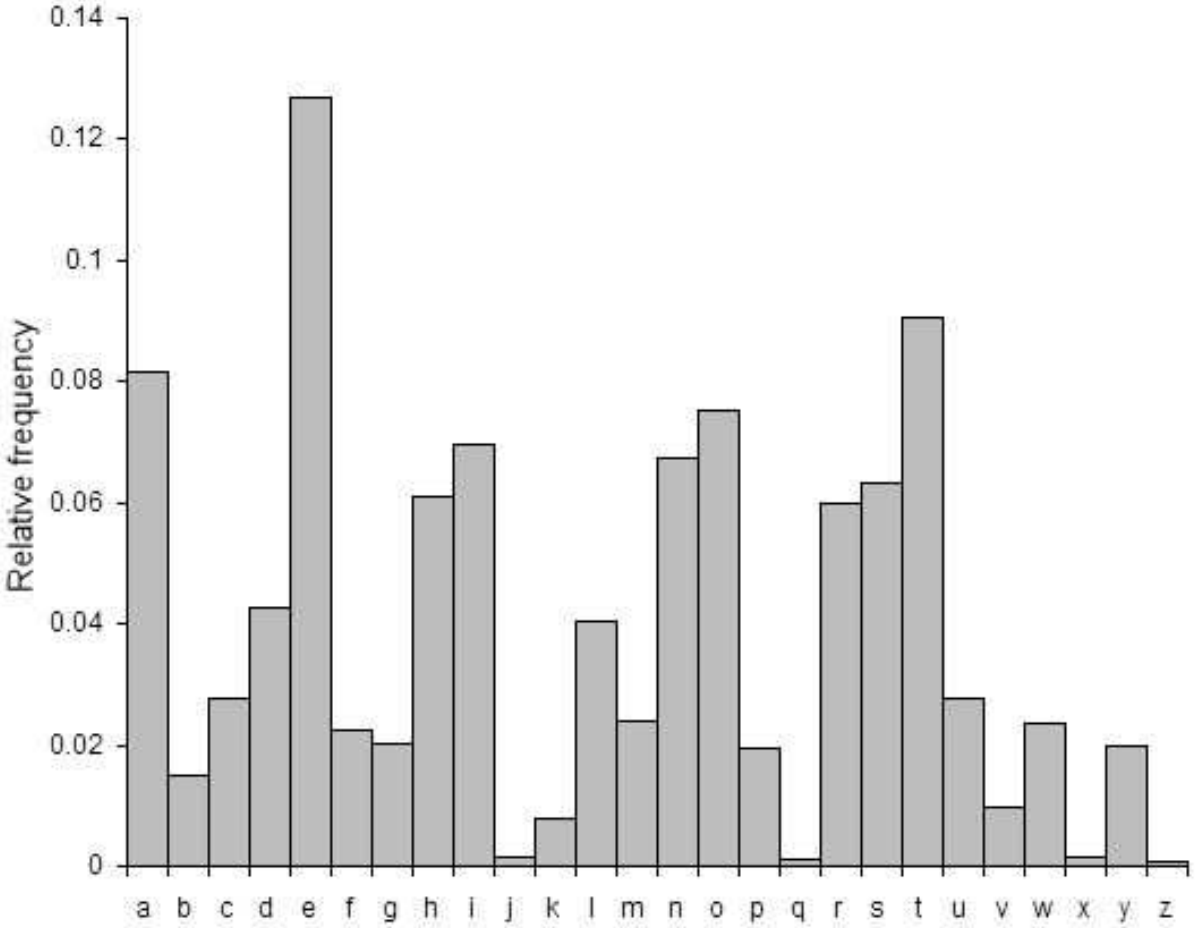
Fact

Since $\#\mathcal{K} = 26! \approx 4 \cdot 10^{26}$, enumerating the keys and *searching for a well-formed plaintext* will not help.

COA on substitution cypher

Idea

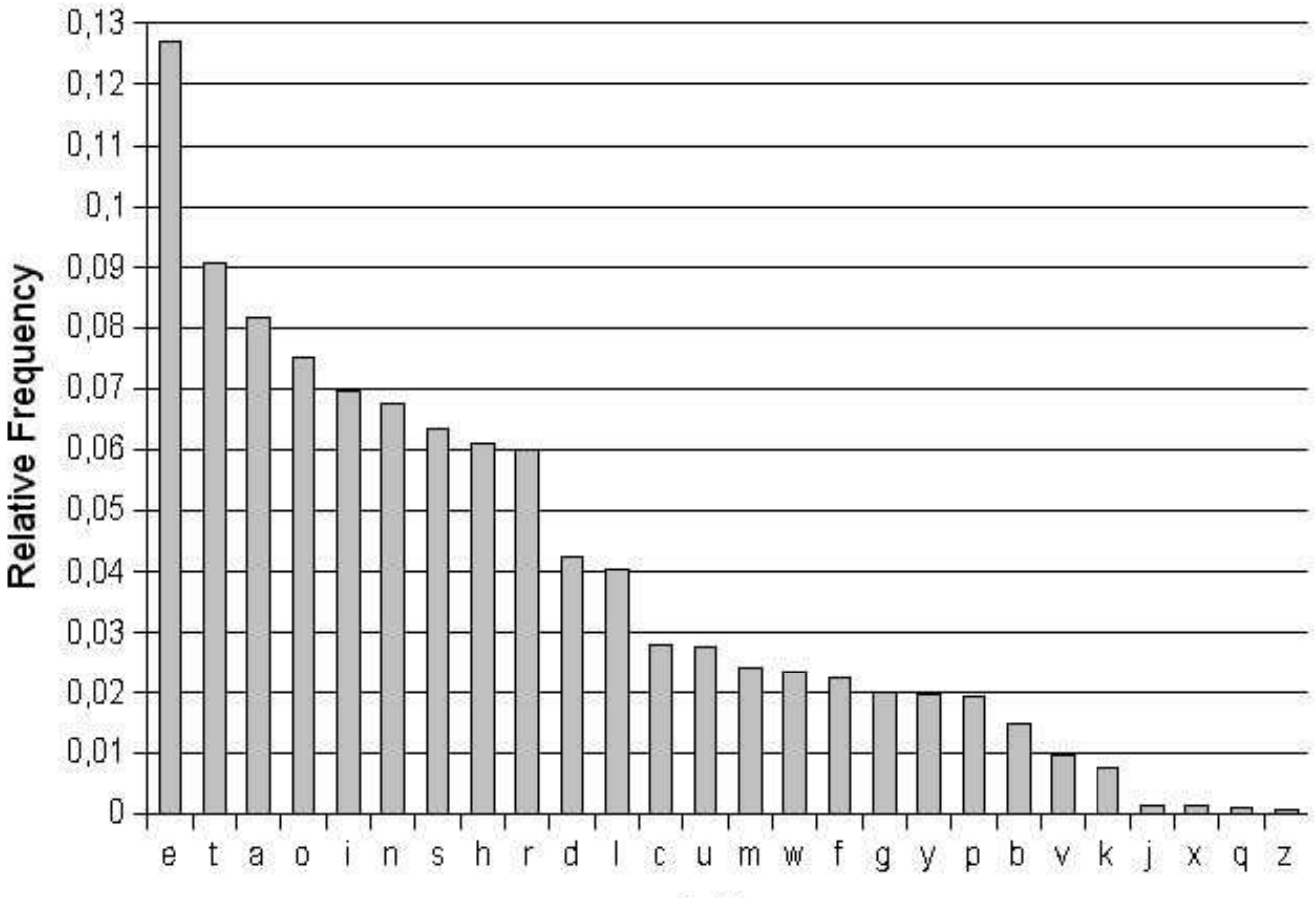
Align the letter frequencies of plaintext (e.g. English)...



COA on substitution cypher

Idea

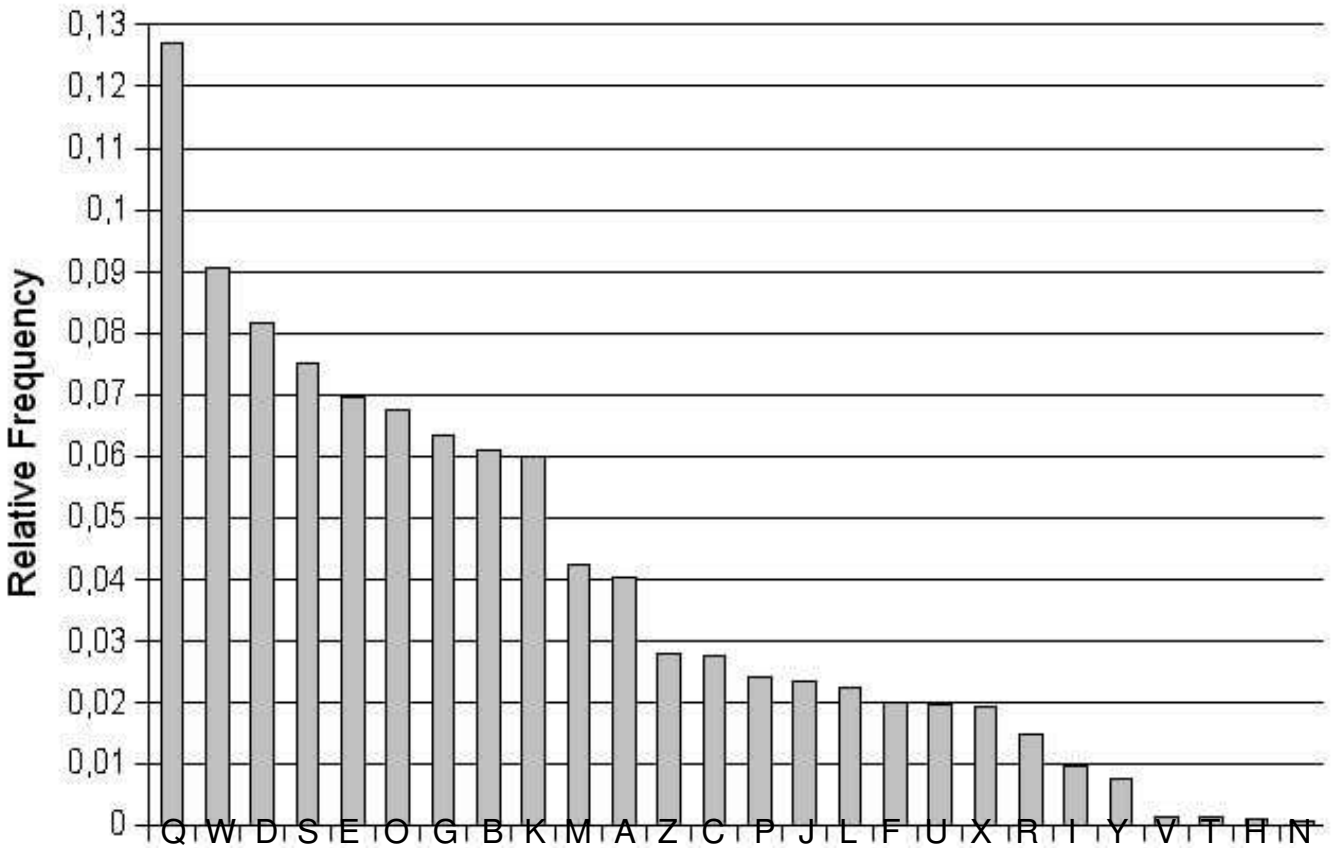
Align the letter frequencies of plaintext (e.g. English)...



COA on substitution cypher

Idea

. . . with the letter frequencies of the cyphertext



COA on substitution cypher

Summary

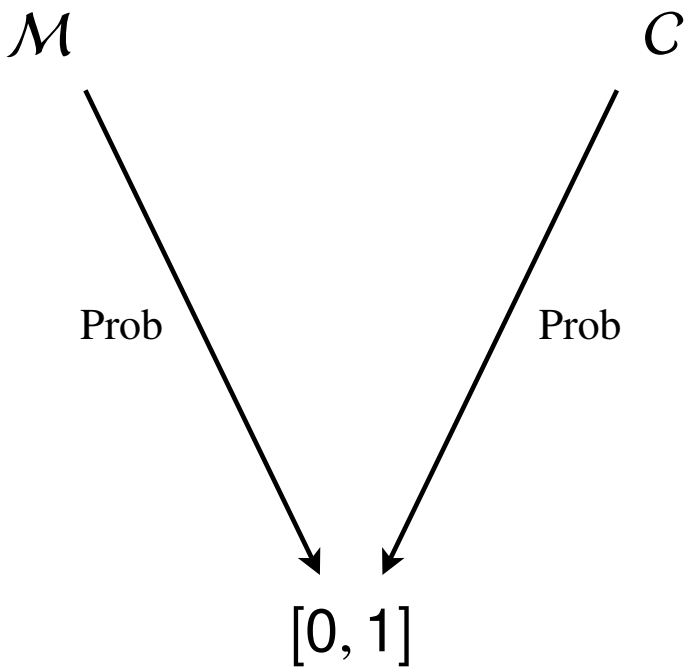
- the messages are drawn from a source $\mathcal{X}$ and coded along $f : \mathcal{X} \longrightarrow \mathcal{G} \subseteq \mathcal{M}^*$
- the frequency distribution $\mathrm{Prob}_{\mathcal{X}} : \mathcal{X} \longrightarrow [0,1]$ induces the frequency distribution $\mathrm{Prob}_{\mathcal{M}} : \mathcal{M} \longrightarrow [0,1]$

$$\mathrm{Prob}_{\mathcal{M}}\left(\vec{m}\right) \quad = \quad \mathrm{Prob}_{\mathcal{X}}\left(f^{-1}(\vec{m})\right.$$

- the frequency distribution $\mathrm{Prob}_{\mathcal{C}} : \mathcal{C} \longrightarrow [0,1]$ can be extracted if there is enough cyphertext

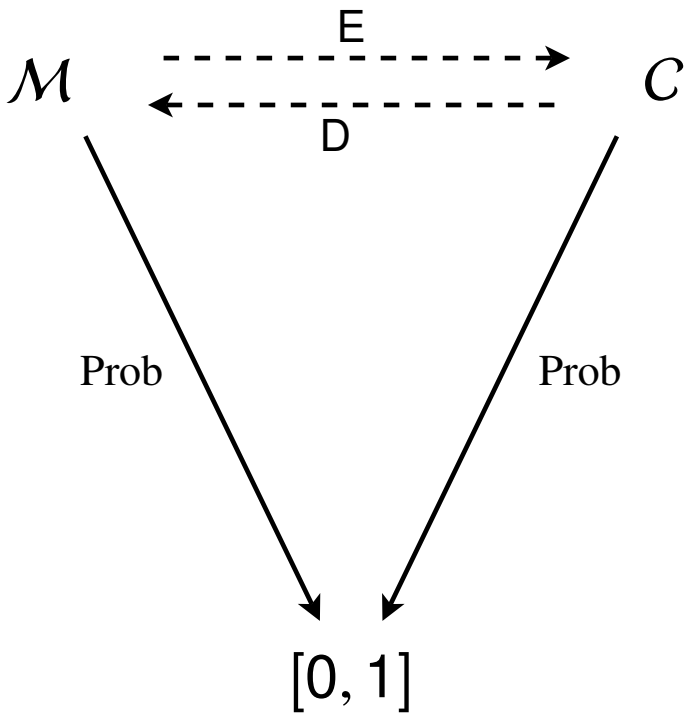
COA on substitution cypher

The patterns



COA on substitution cypher

The patterns are aligned to reconstruct



KPA on the one-time-pad

- $\mathcal{M} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{26}^{N}$
- $\mathsf{E}(\vec{k}, \vec{m}) = \vec{m} + \vec{k}$
- $\mathsf{D}(\vec{k}, \vec{c}) = \vec{c} - \vec{k}$

KPA on the one-time-pad

- $\mathcal{M} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{26}^N$
- $\mathsf{E}(\vec{k}, \vec{m}) = \vec{m} + \vec{k}$
- $\mathsf{D}(\vec{k}, \vec{c}) = \vec{c} - \vec{k}$

Attack

Given $\vec{m}$ and $\mathsf{E}(\vec{k}, \vec{m}) = \vec{m} + \vec{k}$ the cryptanalyst derives

$$\vec{k} \quad = \quad \mathsf{E}(\vec{k}, \vec{m}) - \vec{m}$$

Can we prove that there are no attacks?

Can we prove that there are no attacks?

Proposition

If all keys are equally likely, then the one-time-pad is secure, in the sense that the cyphertext provides no information about the plaintext.

Can we prove that there are no attacks?

We need tools for such proofs!

Guessing

Attack scenario: KPA, CPA

The cryptanalyst knows which crypto system is used. He wants to derive the key from the known or chosen plaintext, and its encryptions

$$m_1, \ldots, m_\ell, \mathrm{E}(\mathrm{K}, m_1), \ldots, \mathrm{E}(\mathrm{K}, m_\ell) \quad \vdash \quad \mathrm{K}$$

Guessing

Attack scenario: KPA, CPA

The cryptanalyst knows which crypto system is used. He wants to derive the key from the known or chosen plaintext, and its encryptions

$$m_1, \ldots, m_\ell, \mathrm{E}(\mathrm{K}, m_1), \ldots, \mathrm{E}(\mathrm{K}, m_\ell) \quad \vdash \quad \mathrm{K}$$

In some cases, he

- may not know the plaintext, but
- can recognize well-formed messages.

Guessing

Terminology

A *random variable* is a function $X : \mathcal{X} \longrightarrow V$ where

- $\mathcal{X}$ is a source and
- V is a set, representing values.

Guessing

Terminology

A *random variable* is a function $X : \mathcal{X} \longrightarrow V$ where

- $\mathcal{X}$ is a source and
- V is a set, representing values.

Notation

We write

$$\begin{aligned} \text{Prob}(X = v) &= \text{Prob}\{x \in \mathcal{X} \mid X(x) = v\} \\ &= \sum_{X(x)=v} \text{Prob}(x) \end{aligned}$$

Guessing

Guessing process

Given a probability distribution over the key space $\mathcal{K}$, a *guessing attack* is a random variable $G : \mathcal{K}^* \longrightarrow \mathbb{N}$, where

$$G(k_1, k_2, \ldots, k_n) \quad = i$$

means that $k_i = \mathrm{K_D}$.

Guessing

Guessing process

Given a probability distribution over the key space $\mathcal{K}$, a *guessing attack* is a random variable $G : \mathcal{K}^* \longrightarrow \mathbb{N}$, where

$$G(k_1, k_2, \ldots, k_n) \quad = i$$

means that $k_i = \mathrm{K_D}$.

Remark

The intuition is that we are given some cyphertext $\vec{c}$, and we test whether $\mathrm{D}(k_i, \vec{c})$ is a well-formed message for one k_i after the other.

Guessing

Exercise

Suppose that there are $\ell = \#\mathcal{K}$ keys, and that they are all equally likely. What is the probability that

- $G = 1$, i.e. the key is guessed at once,
- $G = n$, i.e. the key is guessed after exactly n tries.
- $G \leq n$, i.e. the key is guessed in at most n tries.

Guessing

Solution

- Since there are $\ell = \#\mathcal{K}$ equally likely keys,
  - the probability that the right key is drawn at once is $\text{Prob}(G = 1) = p_1 = \frac{1}{\ell}$;

Guessing

Solution

- Since there are $\ell = \#\mathcal{K}$ equally likely keys,
  - the probability that the right key is drawn at once is $\mathrm{Prob}(G = 1) = p_1 = \frac{1}{\ell}$;
  - the probability that the right key is *not* drawn at once is $q_1 = \mathrm{Prob}(G \neq 1) = 1 - p_1 = \frac{\ell - 1}{\ell}$.

Guessing

Solution

- Since there are $\ell = \#\mathcal{K}$ equally likely keys,
  - the probability that the right key is drawn at once is $\text{Prob}(G = 1) = p_1 = \frac{1}{\ell}$;
  - the probability that the right key is *not* drawn at once is $q_1 = \text{Prob}(G \neq 1) = 1 - p_1 = \frac{\ell - 1}{\ell}$. In this case, we draw again, from $\ell - 1$ untested keys.

Guessing

Solution

- Since there are $\ell = \#\mathcal{K}$ equally likely keys,
  - the probability that the right key is drawn at once is $\text{Prob}(G = 1) = p_1 = \frac{1}{\ell}$;
  - the probability that the right key is *not* drawn at once is $q_1 = \text{Prob}(G \neq 1) = 1 - p_1 = \frac{\ell-1}{\ell}$. In this case, we draw again, from $\ell - 1$ untested keys. This time,
    - the probability that the right key is drawn immediately is now $p_2 = \frac{1}{\ell-1}$, and thus $\text{Prob}(G = 2) = q_1 \cdot p_2 = \frac{\ell-1}{\ell} \cdot \frac{1}{\ell-1} = \frac{1}{\ell}$;

Guessing

Solution

- Since there are $\ell = \#\mathcal{K}$ equally likely keys,
  - the probability that the right key is drawn at once is $\text{Prob}(G = 1) = p_1 = \frac{1}{\ell}$;
  - the probability that the right key is *not* drawn at once is $q_1 = \text{Prob}(G \neq 1) = 1 - p_1 = \frac{\ell-1}{\ell}$. In this case, we draw again, from $\ell - 1$ untested keys. This time,
    - the probability that the right key is drawn immediately is now $p_2 = \frac{1}{\ell-1}$, and thus $\text{Prob}(G = 2) = q_1 \cdot p_2 = \frac{\ell-1}{\ell} \cdot \frac{1}{\ell-1} = \frac{1}{\ell}$;
    - whereas the probability that the right key is still not drawn is $q_2 = \frac{\ell-2}{\ell-1} \ldots$

Guessing

In general, with $p_i = \frac{1}{\ell - i + 1}$ and $q_i = \frac{\ell - i}{\ell - i + 1}$, the probability that a particular key is drawn in the n-th draw is

$$\begin{aligned} \text{Prob}(G = n) &= q_1 \cdot q_2 \cdots q_{n-1} \cdot p_n \\ &= \frac{\ell - 1}{\ell} \cdot \frac{\ell - 2}{\ell - 1} \cdots \frac{\ell - n + 1}{\ell - n + 2} \cdot \frac{1}{\ell - n + 1} \\ &= \frac{1}{\ell} \end{aligned}$$

Guessing

In general, with $p_i = \frac{1}{\ell - i + 1}$ and $q_i = \frac{\ell - i}{\ell - i + 1}$, the probability that a particular key is drawn in the n-th draw is

$$\begin{aligned} \text{Prob}(G = n) &= q_1 \cdot q_2 \cdots q_{n-1} \cdot p_n \\ &= \frac{\ell - 1}{\ell} \cdot \frac{\ell - 2}{\ell - 1} \cdots \frac{\ell - n + 1}{\ell - n + 2} \cdot \frac{1}{\ell - n + 1} \\ &= \frac{1}{\ell} \end{aligned}$$

The probability that a particular key is drawn in at most n tries is

$$\text{Prob}(G \leq n) = \sum_{i=1}^{n} \text{Prob}(G = i) = \frac{n}{\ell}$$

Elements of probability

Notation

Given a source $\mathcal{X}$ and events $\alpha, \beta, \gamma \ldots \subseteq \mathcal{X}$, we write

$$\left[\alpha\right] = \sum_{x \in \alpha} \mathrm{Prob}(x)$$

$$\left[\alpha \vdash \beta\right] = \frac{\left[\alpha \cap \beta\right]}{\left[\alpha\right]}$$

Elements of probability

Remark

Traditionally, our $\left[\alpha \vdash \beta\right]$ is written $\mathrm{Prob}\,(\beta \mid \alpha)$, and called conditional probability.

Elements of probability

Remark

Traditionally, our $\left[\alpha \vdash \beta\right]$ is written $\mathrm{Prob}\left(\beta \mid \alpha\right)$, and called conditional probability.

While the traditional notations need to be respected, cryptography puts conditional probability to heavy use, and abuse.

Elements of probability

Remark

Traditionally, our $\left[\alpha \vdash \beta\right]$ is written $\mathrm{Prob}\left(\beta \mid \alpha\right)$, and called conditional probability.

While the traditional notations need to be respected, cryptography puts conditional probability to heavy use, and abuse.

$\left[\alpha \vdash \beta\right]$ **tells how likely it is to guess** β **from** α**.**

Elements of probability

Homework

$$\left[\alpha \vdash \neg\beta\right] = 1 - \left[\alpha \vdash \beta\right]$$

$$\left[\beta\right] = \left[\alpha\right] \cdot \left[\alpha \vdash \beta\right] + \left[\neg\alpha\right] \cdot \left[\neg\alpha \vdash \beta\right]$$

$$\left[\alpha \vdash \beta \cup \gamma\right] = \left[\alpha \vdash \beta\right] + \left[\alpha \vdash \gamma\right] - \left[\alpha \vdash \beta \cap \gamma\right]$$

Elements of probability

Homework

$$\left[\alpha \vdash \neg\beta\right] = 1 - \left[\alpha \vdash \beta\right]$$

$$\left[\beta\right] = \left[\alpha\right] \cdot \left[\alpha \vdash \beta\right] + \left[\neg\alpha\right] \cdot \left[\neg\alpha \vdash \beta\right]$$

$$\left[\alpha \vdash \beta \cup \gamma\right] = \left[\alpha \vdash \beta\right] + \left[\alpha \vdash \gamma\right] - \left[\alpha \vdash \beta \cap \gamma\right]$$

Moreover

$$\left[\alpha \cap \beta\right] = \left[\alpha\right] \cdot \left[\beta\right] \iff \left[\alpha \vdash \beta\right] = \left[\beta\right]$$
$$\iff \left[\beta \vdash \alpha\right] = \left[\alpha\right]$$

Elements of probability

Bayes theorem

$$\left[\beta \vdash \alpha\right] = \frac{\left[\alpha\right]\left[\alpha \vdash \beta\right]}{\left[\alpha\right]\left[\alpha \vdash \beta\right] + \left[\neg\alpha\right]\left[\neg\alpha \vdash \beta\right]}$$

Elements of probability

Proposition

$$\begin{array}{rcl} \left[\beta \vdash \alpha\right] & = & \left[\gamma \vdash \alpha\right] \\ & \Downarrow & \\ \left[\alpha \vdash \beta\right] \cdot \left[\beta \vdash \gamma\right] & = & \left[\alpha \vdash \gamma\right] \cdot \left[\gamma \vdash \beta\right] \end{array}$$

Elements of probability

Proposition

Since

$$\left[\alpha \vdash \beta \cap \gamma\right] \quad = \quad \left[\alpha \vdash \beta\right] \cdot \left[\alpha \cap \beta \vdash \gamma\right]$$

it follows that

$$\left[\alpha \vdash \beta\right] \cdot \left[\alpha \cap \beta \vdash \gamma\right] \quad \leq \quad \left[\alpha \vdash \gamma\right]$$

with the equality when $\left[\alpha \cap \gamma \vdash \beta\right] = 1$, *so that* $\left[\alpha \vdash \gamma\right] = \left[\alpha \vdash \beta \cap \gamma\right]$.

Problem with simple crypto systems

Leaking partial information

The trapdoor encryption condition

$$\forall m.\mathrm{A}(\mathrm{E}(\mathrm{K_E}, m)) = m \implies \forall c.\mathrm{A}(c) = \mathrm{D}(\mathrm{K_D}, c)$$

only talks about *total* decryptions.

Problem with simple crypto systems

Leaking partial information

The trapdoor encryption condition

$$\forall m.\mathrm{A}(\mathrm{E}(\mathrm{K_E}, m)) = m \implies \forall c.\mathrm{A}(c) = \mathrm{D}(\mathrm{K_D}, c)$$

only talks about *total* decryptions.
A simple crypto system can leak *partial* information.

Problem with simple crypto systems

Two kinds of leaks

The attacker may observe traffic and build

- a *partial* map A : $\mathcal{C} \rightharpoonup \mathcal{M}$
  - e.g., by recognizing
    E(K, "yes"), E(K, "no"), E(K, "buy") . . .
- a map A : $\mathcal{C} \longrightarrow \Delta\mathcal{M}$, extracting *partial information*
  - e.g., by comparing E(K, m_0), E(K, m_1) . . .

Example: Reusing one-time-pad

Proposition

If the same one-time-pad key is used to encrypt more than one block, then a CPA attacker can extract partial information.

Example: Reusing one-time-pad

Proposition

If the same one-time-pad key is used to encrypt more than one block, then a CPA attacker can extract partial information.
E.g., the attacker can form two messages such that, if she is given the encryption of one of them, then she can tell which one. (This is one bit of information extracted.)

Example: Reusing one-time-pad

Proof

The CPA attacker forms two messages in the form:

$$\vec{m}_0 = \vec{m} @ \vec{m} \qquad \vec{m}_1 = \vec{m} @ \vec{\ell}$$

where $\vec{x} @ \vec{y}$ is concatenation and $\vec{\ell} \neq \vec{m}$ are of length N.

Example: Reusing one-time-pad

Proof

The CPA attacker forms two messages in the form:

$$\vec{m}_0 = \vec{m} @ \vec{m} \qquad \vec{m}_1 = \vec{m} @ \vec{\ell}$$

where $\vec{x}@\vec{y}$ is concatenation and $\vec{\ell} \neq \vec{m}$ are of length N. Encrypting with the key $\vec{k}$ of length N gives

$$\mathrm{E}(\vec{k}, \vec{m}_0) = \vec{c} @ \vec{c} \qquad \mathrm{E}(\vec{k}, \vec{m}_1) = \vec{c} @ \vec{d}$$

where $\vec{c} = \vec{m} + \vec{k}$ and $\vec{d} = \vec{m} + \vec{\ell}$.

Probabilistic crypto system

Definition

Given the types

- $\mathcal{M}$ of *plaintexts*
- $\mathcal{C}$ of *cyphertexts*
- $\mathcal{K}$ of *keys*
- $\mathcal{R}$ of *random seeds*

Probabilistic crypto system

Definition

. . . a probabilistic crypto-system is a triple of algorithms:

- key generation $\langle \mathsf{K_E}, \mathsf{K_D} \rangle : \mathcal{R} \longrightarrow \mathcal{K} \times \mathcal{K}$,
- encryption $\mathsf{E} : \mathcal{R} \times \mathcal{K} \times \mathcal{M} \longrightarrow \mathcal{C}$, and
- decryption $\mathsf{D} : \mathcal{K} \times \mathcal{C} \longrightarrow \mathcal{M}$,

When confusion seems unlikely, we abbreviate

- $\mathsf{K}(r)$ to $\mathbb{K}$ and
- $\mathsf{E}(r, k, m)$ to $\mathbb{E}(k, m)$ and even $\mathbb{E}(m)$.

Probabilistic crypto system

Definition

. . . that together provide

- unique decryption:

$$\mathsf{D}(\mathbb{K}_\mathsf{D}, \mathbb{E}(\mathbb{K}_\mathsf{E}, m)) \quad = \quad m$$

- secrecy (Shannon: unconditional, "perfect security"):

$$\left[c \in \mathbb{E}(\mathbb{K}, m) \vdash m \in \mathcal{M}\right] = \left[m \in \mathcal{M}\right] \qquad \text{(IT-SEC)}$$

Probabilistic crypto system

Definition

. . . that together provide

- unique decryption:

$$\mathsf{D}(\mathbb{K}_\mathsf{D}, \mathbb{E}(\mathbb{K}_\mathsf{E}, m)) \quad = \quad m$$

- secrecy:

$$\left[c \in \mathbb{E}(\mathbb{K}, m) \vdash m \in \mathbb{A}(c)\right] = \left[m \in \mathbb{A}(0)\right] \quad \text{(COM-SEC)}$$

for every feasible probabilistic algorithm $\mathbb{A} : \mathcal{C} \longrightarrow \mathcal{M}$, (i.e. $\mathsf{A} : \mathcal{R} \times \mathcal{K} \times \mathcal{C} \longrightarrow \mathcal{M}$)

Probabilistic crypto system

Definition

. . . that together provide

- unique decryption:

$$\mathsf{D}(\mathbb{K}_\mathsf{D}, \mathbb{E}(\mathbb{K}_\mathsf{E}, m)) \quad = \quad m$$

- secrecy:

$$\left[m_0, m_1 \in \mathcal{M}, c \in \mathbb{E}(\mathbb{K}, m_b) \vdash b \in \{0,1\}\right] =$$

$$\left[m_0, m_1 \in \mathcal{M} \vdash b \in \{0,1\}\right] = \frac{1}{2} \qquad \text{(IT-IND)}$$

Probabilistic crypto system

Definition

. . . that together provide

- unique decryption:

$$\mathsf{D}(\mathbb{K}_\mathsf{D}, \mathbb{E}(\mathbb{K}_\mathsf{E}, m)) \quad = \quad m$$

- secrecy:

$$\left[m_0, m_1 \in \mathcal{M}, c \in \mathbb{E}(m_b) \vdash b \in \mathbb{A}(m_0, m_1, c)\right] \leq$$

$$\left[m_0, m_1 \in \mathcal{M} \vdash b \in \mathbb{A}(m_0, m_1, 0)\right] \leq \frac{1}{2} \quad \text{(COM-IND)}$$

for any feasible probabilistic $\mathbb{A} : \mathcal{M} \times \mathcal{M} \times C \longrightarrow \{0, 1\}$ (with K_E and the seed implicit)

Probabilistic crypto system

Definition

. . . that together provide

- unique decryption:

$$\mathsf{D}(\mathbb{K}_\mathsf{D}, \mathbb{E}(\mathbb{K}_\mathsf{E}, m)) \quad = \quad m$$

- secrecy (Goldwasser-Micali: "semantic security")

$$\Big[m_0, m_1 \in \mathbb{A}_0, c \in \mathbb{E}(m_b) \vdash$$

$$b \in \mathbb{A}_1(m_0, m_1, c)\Big] \quad \leq \quad \frac{1}{2} \quad \text{(IND-CPA)}$$

for any probabilistic algorithm $\mathbb{A} = \langle \mathbb{A}_0, \mathbb{A}_1 \rangle \ldots$

Probabilistic crypto system

Definition

. . . that together provide

- unique decryption:

$$\mathsf{D}(\mathbb{K}_\mathsf{D}, \mathbb{E}(\mathbb{K}_\mathsf{E}, m)) \quad = \quad m$$

- secrecy (under chosen cyphertext attack):

$$\left[\begin{array}{l} c_0 \in \mathbb{A}_0,\ m \in \mathsf{D}(c_0), \\ m_0, m_1 \in \mathbb{A}_1(c_0, m), c \in \mathbb{E}(m_b) \end{array} \vdash \right.$$

$$\left. b \in \mathbb{A}_2(c_0, m, m_0, m_1, c) \right] \quad \leq \quad \frac{1}{2} \quad \text{(IND-CCA)}$$

for any probabilistic algorithm $\mathbb{A} = \langle \mathbb{A}_0, \mathbb{A}_1, \mathbb{A}_2 \rangle$. . .

Probabilistic crypto system

Definition

. . . that together provide

- unique decryption:

$$\mathsf{D}(\mathbb{K}_\mathsf{D}, \mathbb{E}(\mathbb{K}_\mathsf{E}, m)) \quad = \quad m$$

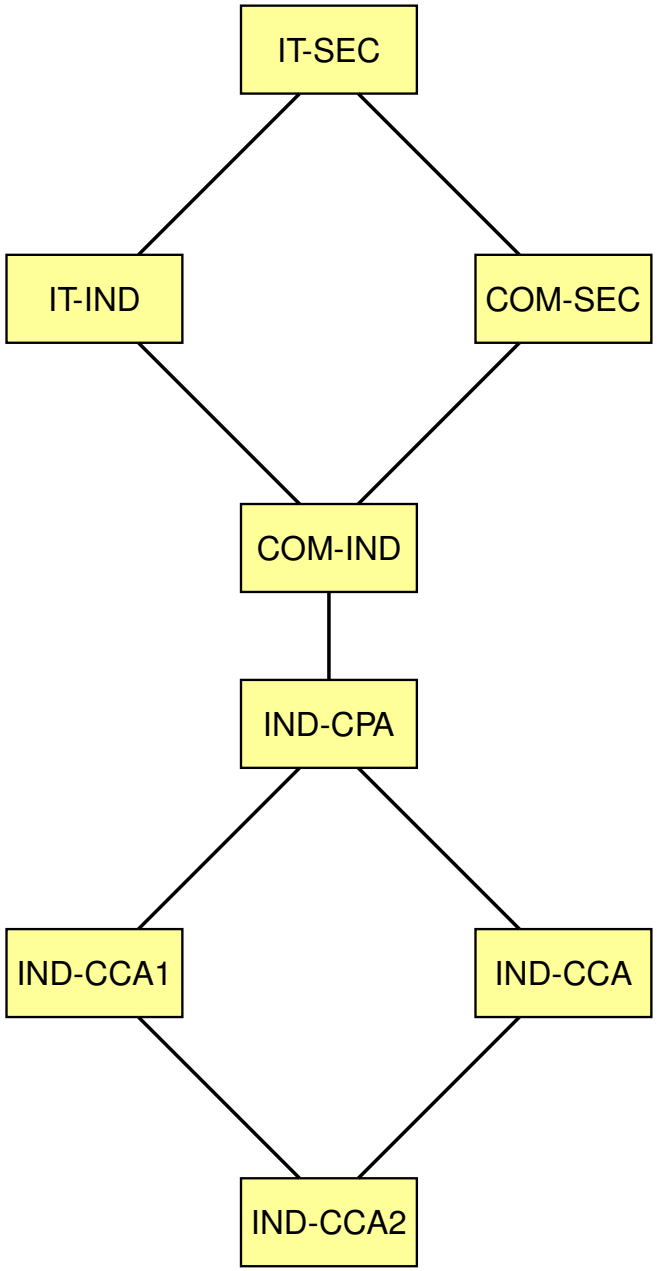
- secrecy (under *adaptive* chosen cyphertext attack):

$$\left[\begin{array}{l} c_0 \in \mathbb{A}_0,\ m \in \mathsf{D}(c_0), \\ m_0, m_1 \in \mathbb{A}_1(c_0, m), c \in \mathbb{E}(m_b) \\ c_1 \in \mathbb{A}_2(c_0, m, m_0, m_1), \widetilde{m} \in \mathsf{D}(c_1 \neq c) \end{array} \vdash \right.$$

$$\left. b \in \mathbb{A}_3(c_0, m, m_0, m_1, c, c_1, \widetilde{m}) \right] \quad \leq \quad \frac{1}{2} \quad \text{(IND-CCA2)}$$

for any probabilistic algorithm $\mathbb{A} = \langle \mathbb{A}_0, \mathbb{A}_1, \mathbb{A}_2, \mathbb{A}_3 \rangle \ldots$

Taxonomy of secrecy properties



Example: El Gamal

Fix a finite field $\mathbb{F}$ and $g \in \mathbb{F}^*$.

$$\mathcal{M} = \mathcal{R} = \mathbb{F} \qquad \mathsf{K}_\mathsf{E}(a) = g^a$$

$$\mathcal{C} = \mathbb{F}^* \times \mathbb{F} \qquad \mathsf{K}_\mathsf{D}(a) = a$$

$$\mathcal{K} = \mathbb{F}^* \times \mathbb{F}^* \qquad \mathsf{E}(r, k, m) = \left\langle g^r, k^r \cdot m \right\rangle$$

$$\mathsf{D}\left(\overline{k}, \langle c_1, c_2 \rangle\right) = \frac{c_2}{c_1^{\overline{k}}}$$

Example: El Gamal

Fix a finite field $\mathbb{F}$ and $g \in \mathbb{F}^*$.

$$\mathcal{M} = \mathcal{R} = \mathbb{F} \qquad \mathsf{K}_\mathsf{E}(a) = g^a$$
$$\mathcal{C} = \mathbb{F}^* \times \mathbb{F} \qquad \mathsf{K}_\mathsf{D}(a) = a$$
$$\mathcal{K} = \mathbb{F}^* \times \mathbb{F}^* \qquad \mathsf{E}(r, k, m) = \left\langle g^r, k^r \cdot m \right\rangle$$
$$\mathsf{D}\left(\overline{k}, \langle c_1, c_2 \rangle\right) = \frac{c_2}{c_1^{\overline{k}}}$$

Unique decryption

$$\begin{aligned} \mathsf{D}\left(\mathsf{K}_\mathsf{D}(a), \mathsf{E}(r, \mathsf{K}_\mathsf{E}(a), m)\right) &= \mathsf{D}\left(a, \mathsf{E}(r, g^a, m)\right) \\ &= \mathsf{D}\left(a, \left\langle g^r, (g^a)^r \cdot m \right\rangle\right) \\ &= \frac{g^{ar} \cdot m}{(g^r)^a} = m \end{aligned}$$

Unconditional security of one-time-pad

Proposition

If all keys are equally likely, then the one-time-pad is unconditionally secure, i.e. it satisfies (IT-SEC).

Unconditional security of one-time-pad

Proposition

If all keys are equally likely, then the one-time-pad is unconditionally secure, i.e. it satisfies (IT-SEC).

Proof

$\left[c \in \mathcal{C} \vdash m \in \mathcal{M}\right] = \left[m \in \mathcal{M}\right]$ follows from
$\left[m \in \mathcal{M} \vdash c \in \mathcal{C}\right] = \left[c \in \mathcal{C}\right]$ because

$$\left[c \in \mathcal{C} \vdash m \in \mathcal{M}\right] = \frac{\left[m \in \mathcal{M}\right] \cdot \left[m \in \mathcal{M} \vdash c \in \mathcal{C}\right]}{\left[c \in \mathcal{C}\right]}$$

...

Unconditional security of one-time-pad

Proof (continued)

On one hand, for all messages m and cyphertexts c holds

$$\left[m \in \mathcal{M} \vdash c \in \mathcal{C}\right] = \left[k = c - m \in \mathcal{K}\right] = \frac{1}{26^N}$$

Unconditional security of one-time-pad

Proof (continued)

On the other hand, we have

$$\begin{aligned}
\left[c \in C\right] &= \sum_{m+k=c} \left[m \in \mathcal{M}\right] \cdot \left[k \in \mathcal{K}\right] \\
&= \sum_{m \in \mathcal{M}} \left[m \in \mathcal{M}\right] \cdot \left[c - m \in \mathcal{K}\right] \\
&= \frac{1}{26^N} \sum_{m \in \mathcal{M}} \left[m \in \mathcal{M}\right] \\
&= \frac{1}{26^N}
\end{aligned}$$

Security of El Gamal

Computational Diffie-Hellman Assumption (CDH)

There is no feasible probabilistic algorithm CDH : $\mathbb{F}^2 \longrightarrow \mathbb{F}$ such that for all $a, b \in \mathbb{F}$ holds with a high probability

$$\mathrm{CDH}(g^a, g^b) \quad = \quad g^{ab}$$

Security of El Gamal

Computational Diffie-Hellman Assumption (CDH)

There is no feasible probabilistic algorithm CDH : $\mathbb{F}^2 \longrightarrow \mathbb{F}$ such that for all $a, b \in \mathbb{F}$ holds with a high probability

$$\mathrm{CDH}(g^a, g^b) \quad = \quad g^{ab}$$

Decision Diffie-Hellman Assumption (DDH)

There is no feasible prob. algorithm DDH : $\mathbb{F}^3 \longrightarrow \{0, 1\}$ such that for all $a, b \in \mathbb{F}$ holds with a probability $> \frac{1}{2}$

$$\mathrm{DDH}(x, y, z) \quad = \quad \begin{cases} 1 & \text{if } \exists uv.\ x = g^u \wedge y = g^v \wedge z = g^{uv} \\ 0 & \text{otherwise} \end{cases}$$

Security of El Gamal

Proposition

El Gamal satisfies (IND-CPA) if and only if (DDH) holds.
El Gamal does not safisty (IND-CCA).

Security of El Gamal

Recall the definitions:

...

- unique decryption:

$$\mathsf{D}(\mathbb{K}_\mathsf{D}, \mathbb{E}(\mathbb{K}_\mathsf{E}, m)) \quad = \quad m$$

- secrecy (Goldwasser-Micali: "semantic security")

$$\Big[m_0, m_1 \in \mathbb{A}_0, c \in \mathbb{E}(m_b) \vdash b \in \mathbb{A}_1(m_0, m_1, c)\Big] \quad \leq \quad \frac{1}{2} \quad \text{(IND-CPA)}$$

for any probabilistic algorithm $\mathbb{A} = \langle \mathbb{A}_0, \mathbb{A}_1 \rangle$...

Security of El Gamal

Recall the definitions:

...

- unique decryption:

$$\mathsf{D}(\mathbb{K}_\mathsf{D}, \mathbb{E}(\mathbb{K}_\mathsf{E}, m)) \quad = \quad m$$

- secrecy (under chosen cyphertext attack):

$$\left[\begin{array}{l} c_0 \in \mathbb{A}_0,\ m \in \mathsf{D}(c_0), \\ m_0, m_1 \in \mathbb{A}_1(c_0, m), c \in \mathbb{E}(m_b) \end{array} \vdash \right.$$

$$\left. b \in \mathbb{A}_2(c_0, m,\ m_0, m_1, c) \right] \quad \leq \quad \frac{1}{2} \quad \text{(IND-CCA)}$$

for any probabilistic algorithm $\mathbb{A} = \langle \mathbb{A}_0, \mathbb{A}_1, \mathbb{A}_2 \rangle$...

Security of El Gamal

Proof of (DDH)⇒(IND-CPA)

Suppose ¬(IND-CPA).

Security of El Gamal

Proof of (DDH)⇒(IND-CPA)

Suppose ¬(IND-CPA).

This means that there is a feasible probabilistic algorithm $\mathbb{A} = \langle \mathbb{A}_0, \mathbb{A}_1 \rangle$ which

- generates $m_0, m_1 \in \mathbb{A}_0(k)$, and then
- guesses $b \in \mathbb{A}_1(k, m_0, m_1, c_b)$ with a probability $> \frac{1}{2}$
  - where $c_b = \mathrm{E}(s, k, m_b)$ for $b \in \{0, 1\}$.

Security of El Gamal

Proof of (DDH)⇒(IND-CPA)

Suppose ¬(IND-CPA).

This means that there is a feasible probabilistic algorithm $\mathbb{A} = \langle \mathbb{A}_0, \mathbb{A}_1 \rangle$ which

- generates $m_0, m_1 \in \mathbb{A}_0(k)$, and then
- guesses $b \in \mathbb{A}_1(k, m_0, m_1, c_b)$ with a probability $> \frac{1}{2}$
  - where $c_b = \mathrm{E}(s, k, m_b)$ for $b \in \{0, 1\}$.

We construct the algorithm DDH : $\mathbb{F}^3 \longrightarrow \{0, 1\}$ to decide whether a triple $\langle x, y, z \rangle$ is in the form $\langle g^u, g^v, g^{uv} \rangle$ for some $u, v \in \mathbb{F}$.

Security of El Gamal

Proof (continued)

If the private key $K_D = u$, then El Gamal encrypts

$$E(v, g^u, m) = \langle g^v, g^{uv} \cdot m \rangle$$

Security of El Gamal

Proof (continued)

If the private key $K_D = u$, then El Gamal encrypts

$$E(v, g^u, m) \quad = \quad \langle g^v, g^{uv} \cdot m \rangle$$

This means that

$$DDH(x, y, z) = 1 \quad \Longleftrightarrow \quad \forall m. \mathbb{E}(x, m) = \langle y, z \cdot m \rangle$$

Security of El Gamal

Proof (continued)

If the private key $K_D = u$, then El Gamal encrypts

$$E(v, g^u, m) = \langle g^v, g^{uv} \cdot m \rangle$$

This means that

$$DDH(x, y, z) = 1 \iff \forall m. \mathbb{E}(x, m) = \langle y, z \cdot m \rangle$$

But ¬(IND-CPA) says that $\mathbb{A} = \langle A_0, A_1 \rangle$ can decide the right-hand side, so that $m_0, m_1 \in A_0(x)$ gives

$$DDH(x, y, z) = \begin{cases} 1 & \text{if } A_1(x, m_0, m_1, \langle y, z \cdot m_0 \rangle) = 0 \\ & \text{and } A_1(x, m_0, m_1, \langle y, z \cdot m_1 \rangle) = 1 \\ 0 & \text{otherwise} \end{cases}$$

Security of El Gamal

Homework

Complete the proof of the Proposition, showing that

- (IND-CPA)⇒(DDH)
- (IND-CCA) does not hold.

Outline

Modes of operation

ECB
CCB
(Ramzan)

Composite cryptosystems

Shannon's group algebra.
We mix and compose

- substitution cyphers and
- transposition cyphers

In diagrams, substitutions are boxes; but *transpositions* are knots of threads.
Feistel cyphers are a standardized form to perform a simple transposition: they split the output in two sets of strings, and send them to different places.

Algebra of dataflow

There is a whole algebra of transpositions. Transpositions are the terms of an algebra where each variable must be used exactly once. (Pitts-Gabbay: names, variables, nonces.)
The Feistel cypher and the modes of operation are very special terms in this algebra.
DES and AES.

Outline

Information, channel security, noninterference

Encryption and decryption

Cryptanalysis and notions of secrecy

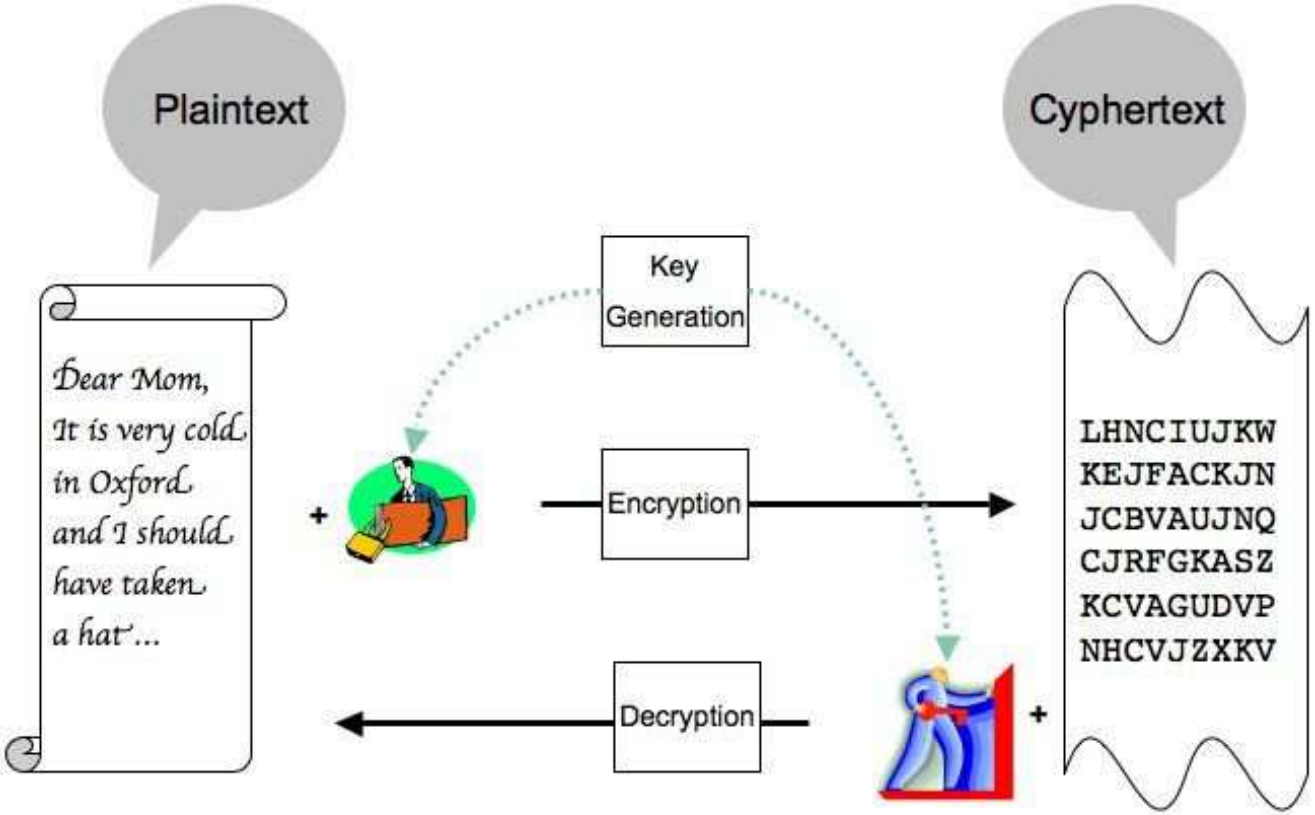
Cyphers and modes of operation

Key establishment

- "Programming Satan's computer"
- Diffie-Hellman Key Agreement
- Needham-Schroeder Public Key Protocol

What did we learn?

Key establishment



Where do the keys come from?

Key establishment

- Traditionally, keys sent through a secure channel
  - messenger, direct handover, physical protection

Key establishment

- Traditionally, keys sent through a secure channel
  - messenger, direct handover, physical protection
- In cyberspace, there are no secure channels
  - only you and me and cryptography

Key establishment in cyberspace

What is cyberspace?

Key establishment in cyberspace

What is cyberspace?

- space of costless communication
  - instantaneous message delivery
  - any two nodes are neighbors: no notion of distance

Key establishment in cyberspace

What is cyberspace?

- space of costless communication
  - instantaneous message delivery
  - any two nodes are neighbors: no notion of distance
- end-to-end architecture (TCP, UDP)
  - simple network links
  - smart network nodes ("ends")

Key establishment in cyberspace

What is cyberspace?

- space of costless communication
  - instantaneous message delivery
  - any two nodes are neighbors: no notion of distance
- end-to-end architecture (TCP, UDP)
  - simple network links
  - smart network nodes ("ends")
- "Satan's computer" (Ross Anderson)
  - network controlled by the adversaries: Eve, Satan
  - security only through crypto at the "ends"

Key establishment in cyberspace

Generate your own public key

- **El Gamal:** Alice generates $K = \langle g^a, a \rangle$
  - she picks $K_D = a$
  - computes $K_E = g^a$ and
  - sends K_E to Bob

Key establishment in cyberspace

Generate your own public key

- **El Gamal:** Alice generates $\mathrm{K} = \langle g^a, a \rangle$
  - she picks $K_\mathrm{D} = a$
  - computes $K_\mathrm{E} = g^a$ and
  - sends K_E to Bob
- **RSA**: Alice generates $\mathrm{K} = \left\langle \langle n, e \rangle, d \rangle \right\rangle$
  - she picks large primes p and q and sets $n = pq$
  - picks $e \in \mathbb{Z}^*_{(p-1)(q-1)}$
  - computes $\mathrm{K_D} = d = e^{-1} \mod (p-1)(q-1)$
  - sends $\mathrm{K_E} = \langle n, e \rangle$ to Bob

Key establishment in cyberspace

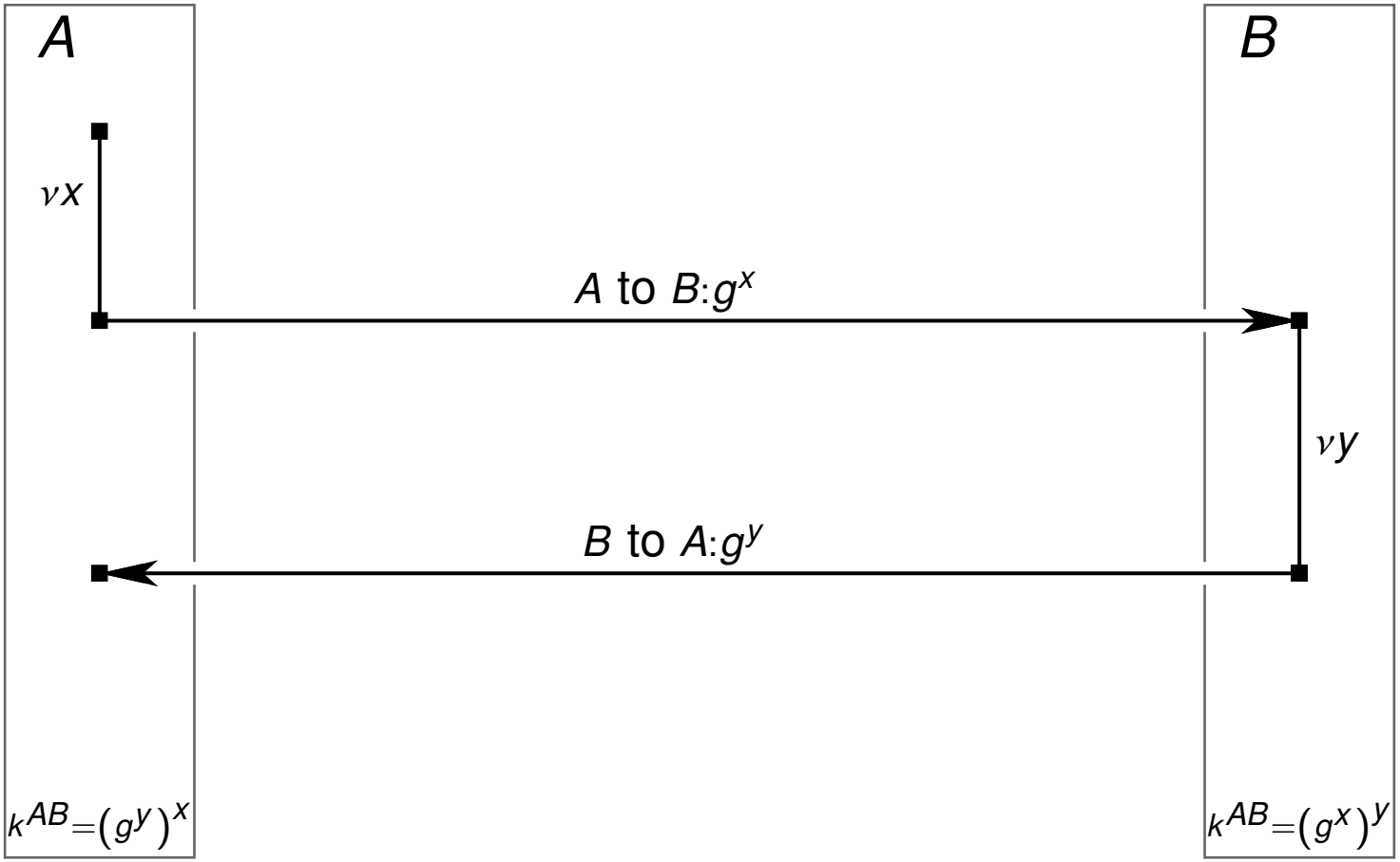
Problem

Eve can impersonate Alice

- Eve can generate K_E and K_D,
- send K_D to Bob
- and say *"Hi, Alice here, this is my key"*.
  - Bob encrypts his messages to Alice by K_E
  - Eve decrypts them by K_D.

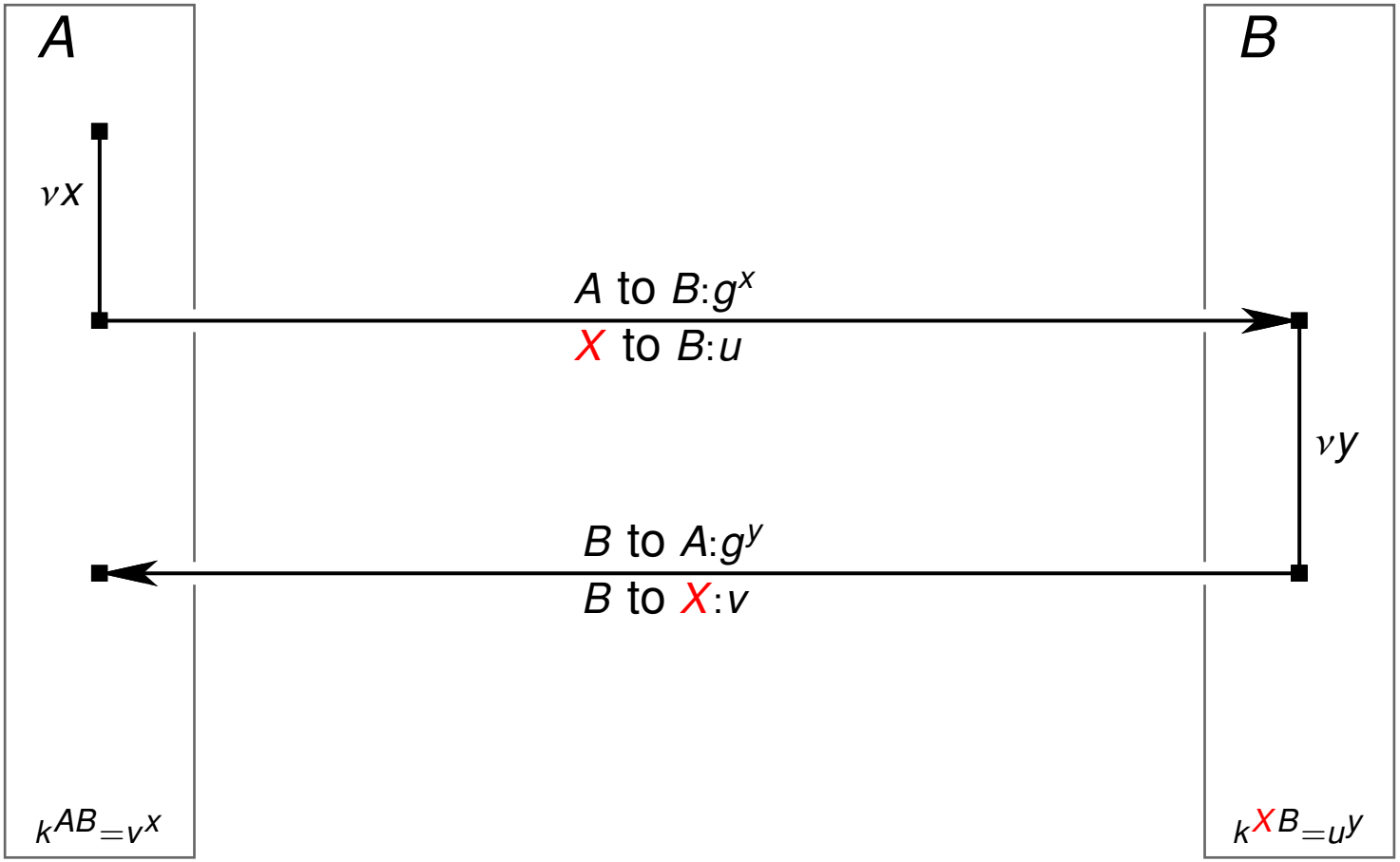
Two party key agreement

Diffie-Hellman Key Agreement Protocol (DHKA)



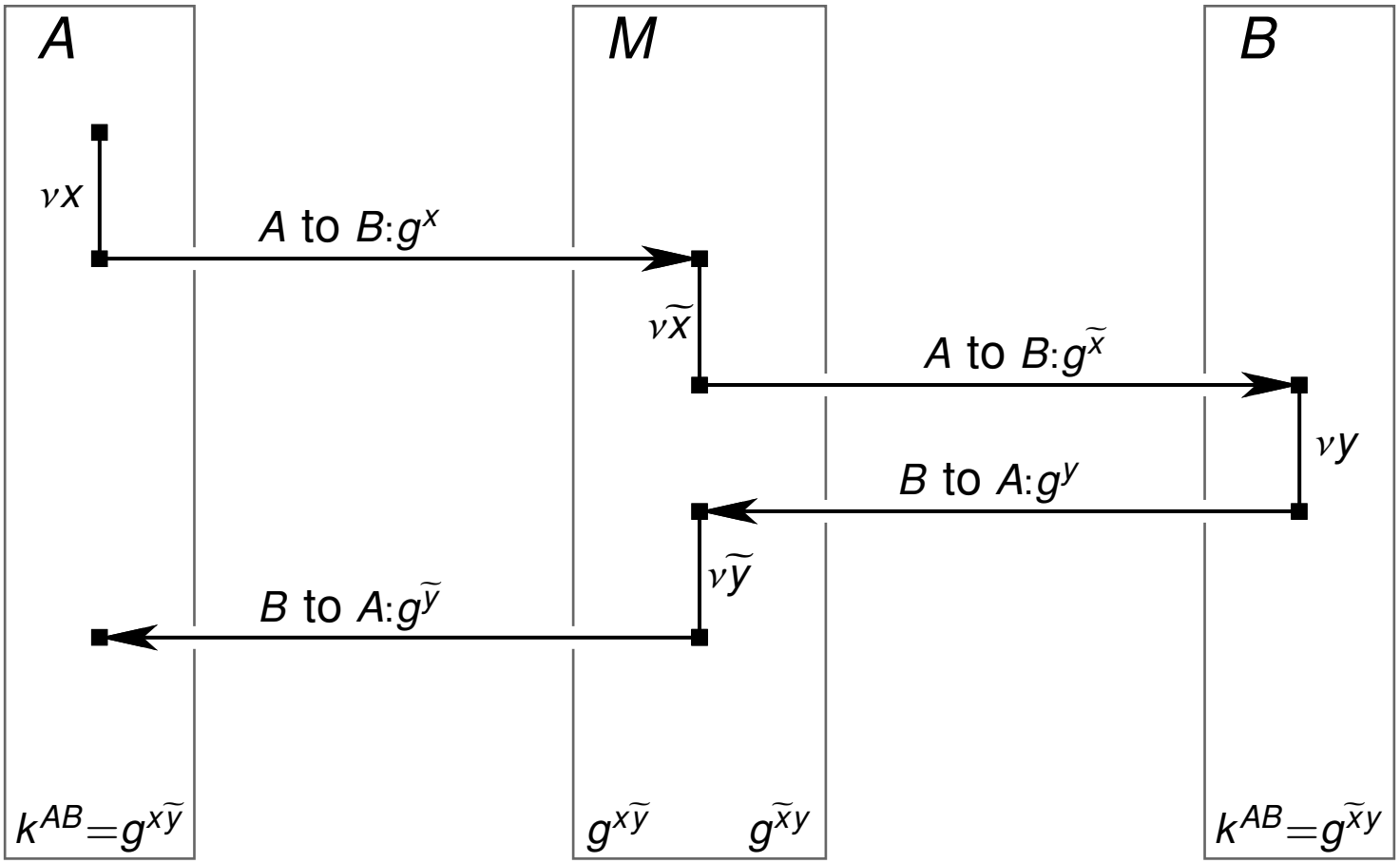
Two party key agreement

Diffie-Hellman Key Agreement Protocol (DHKA)



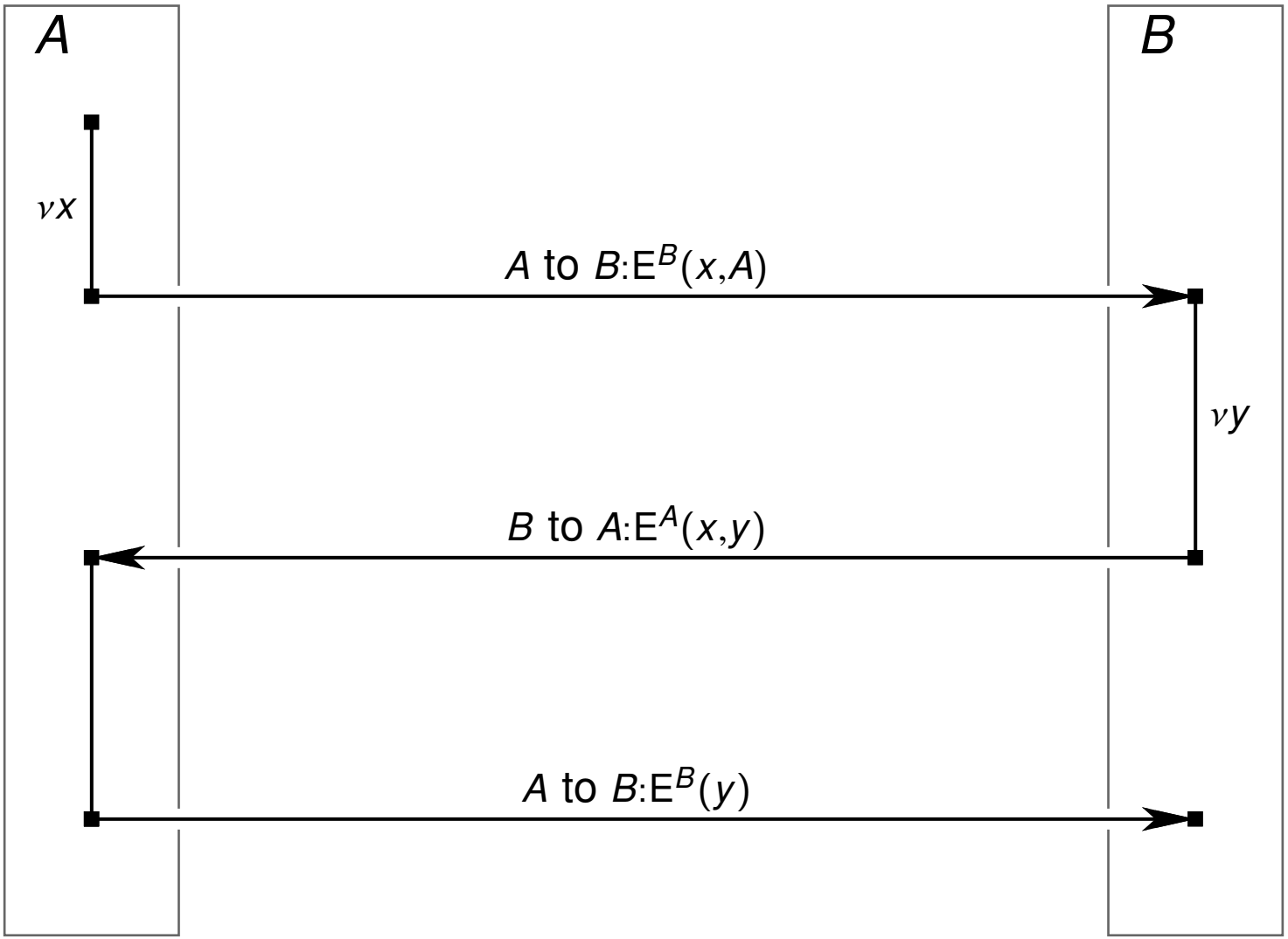
Two party key agreement

Attack on DHKA



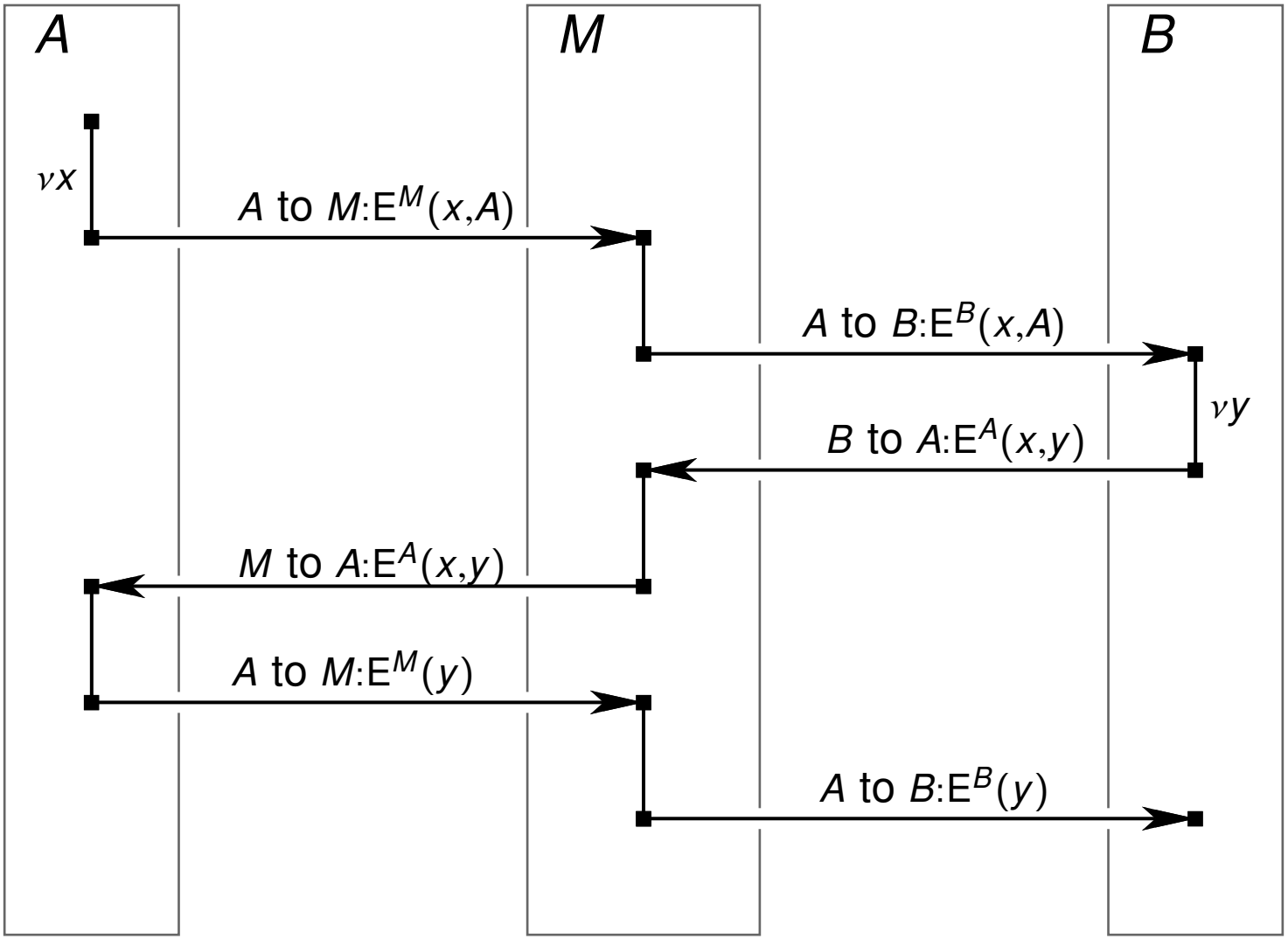
Bootstrapping key agreement

Needham-Schroeder Public Key Protocol (NSPK)



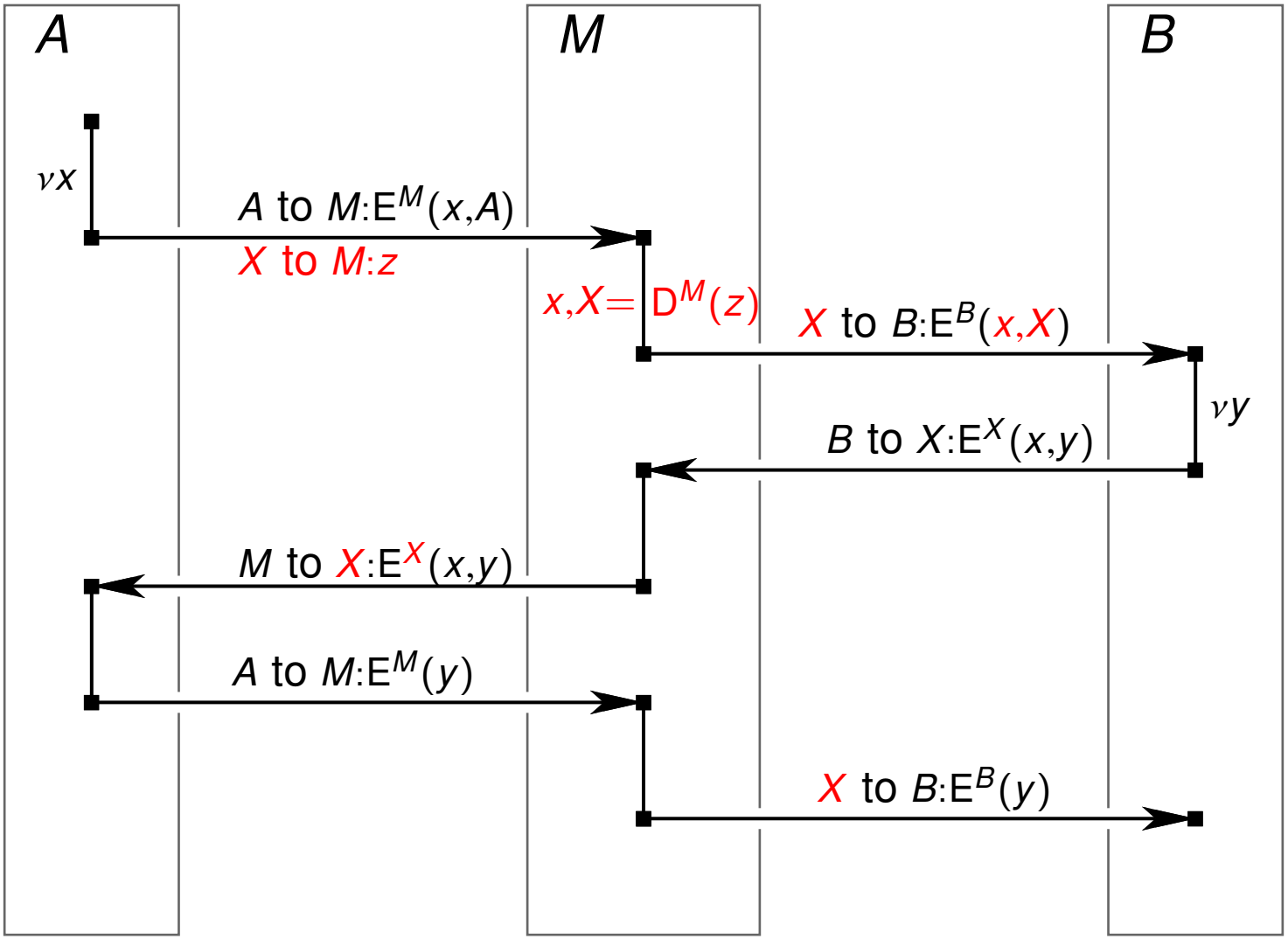
Bootstrapping key agreement

Attack on NSPK



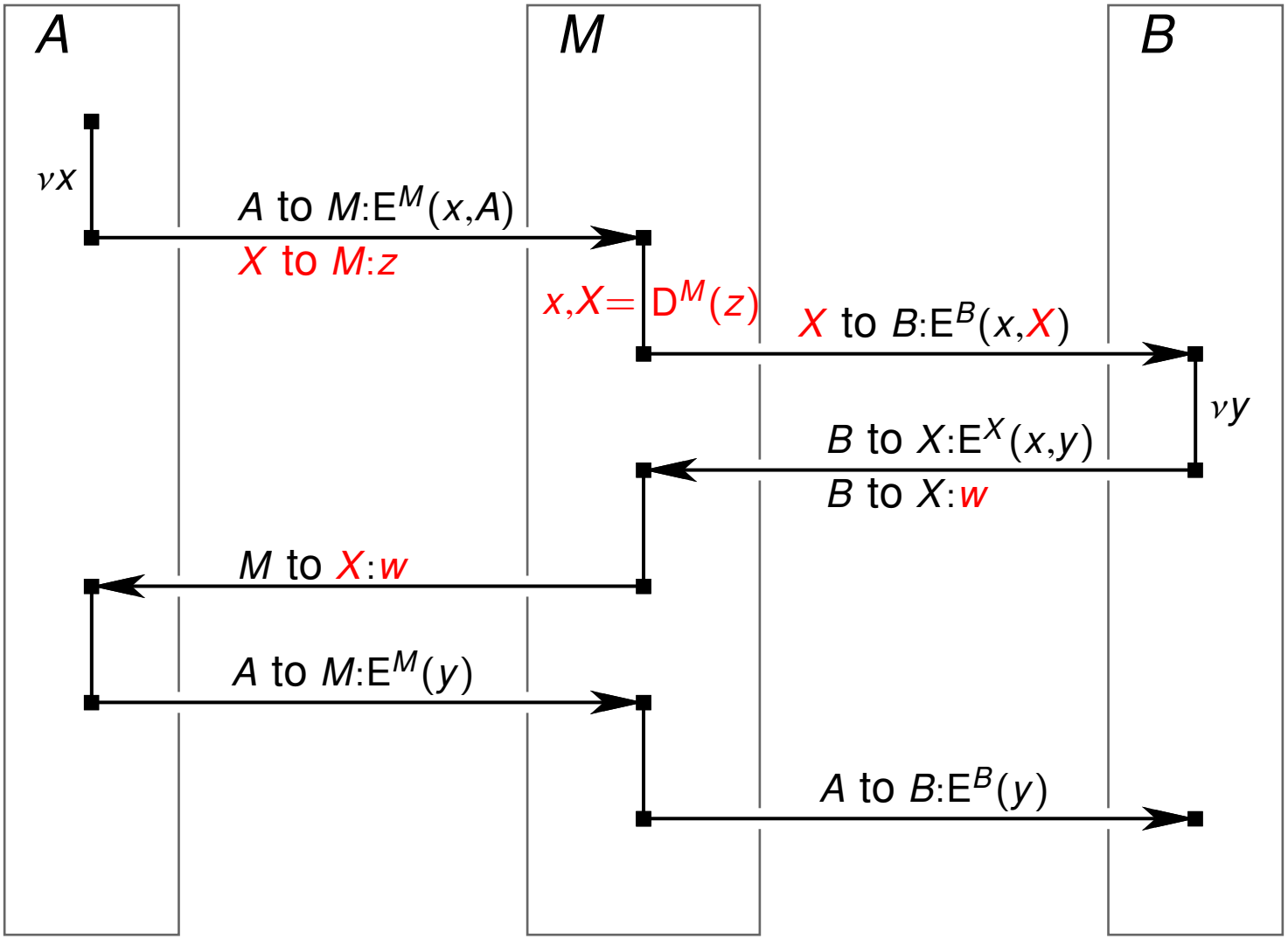
Bootstrapping key agreement

Attack on NSPK



Bootstrapping key agreement

Attack on NSPK



Bootstrapping key agreement

History of NSPK

- NSPK was proposed by in a seminal paper in 1978.

Bootstrapping key agreement

History of NSPK

- NSPK was proposed by in a seminal paper in 1978.
- It was often used and studied.

Bootstrapping key agreement

History of NSPK

- NSPK was proposed by in a seminal paper in 1978.
- It was often used and studied.
- In 1996, Gavin Lowe found the attack
  - using the FDR (Failure Divergence Refinement) checker
  - as a part of his project work at Comlab

Bootstrapping key agreement

History of NSPK

- NSPK was proposed by in a seminal paper in 1978.
- It was often used and studied.
- In 1996, Gavin Lowe found the attack
  - using the FDR (Failure Divergence Refinement) checker
  - as a part of his project work at Comlab
- Later he built Casper.
- More at practicals!

Outline

Lessons about the bad information flows

- information leaks through interference of resources
  - covert channels are hard to eliminate
  - formal models help prevent Trojan intrusions

Lessons about the bad information flows

- information leaks through interference of resources
  - covert channels are hard to eliminate
  - formal models help prevent Trojan intrusions
- secrecy is achieved in complicated ways
  - some of the "purest" maths became the most applied
  - public key crypto needed a public science of crypto

Lessons about the bad information flows

- information leaks through interference of resources
  - covert channels are hard to eliminate
  - formal models help prevent Trojan intrusions
- secrecy is achieved in complicated ways
  - some of the "purest" maths became the most applied
  - public key crypto needed a public science of crypto
- but cryptanalysis is also hard
  - encryptions are not broken every day
  - most security failures arise from **protocol failures**

Lessons about computation

Lessons about computation

- The simple insights that
  - some computations are hard to invert
    - e.g., getting p or q from pq, or a from g^a and g
  - some informations are hard to guess
    - if the source is large and unbiased

Lessons about computation

- The simple insights that
  - some computations are hard to invert
    - e.g., getting p or q from pq, or a from g^a and g
  - some informations are hard to guess
    - if the source is large and unbiased
- point to the important lesson that
  - **complexity** and
  - **randomness**

  are **powerful computational resources**.

Lessons about computation

- The simple insights that
  - some computations are hard to invert
    - e.g., getting p or q from pq, or a from g^a and g
  - some informations are hard to guess
    - if the source is large and unbiased
- point to the important lesson that
  - **complexity** and
  - **randomness**

  are **powerful computational resources**.
- The negative can be used as the positive.

. . . are used to push good information flows

- The absence of bad information flows
- is a fulcrum to move the good information flows.

. . . are used to push good information flows

- The absence of bad information flows
  - "If noone can forge Alice's signature. . .
- is a fulcrum to move the good information flows.
  - . . . then this message must be from Alice :)))"

Guiding principles for the next part

Guiding principles for the next part

- **Every secret must be authenticated**
  - to prevent impersonation.
  - Most protocol failures are authentication failures .

Guiding principles for the next part

- **Every secret must be authenticated**
  - to prevent impersonation.
  - Most protocol failures are authentication failures .
- **Every authentication must be based on a secret**
  - (in cyberspace).
  - The chicken and the egg.

Guiding principles for the next part

- **Every secret must be authenticated**
  - to prevent impersonation.
  - Most protocol failures are authentication failures .
- **Every authentication must be based on a secret**
  - (in cyberspace).
  - The chicken and the egg.
- **Security is always bootstrapped**
  - secrecy and authenticity are based on each other
  - new secrets are derived from old secrets