Winter'18 CS 485/585

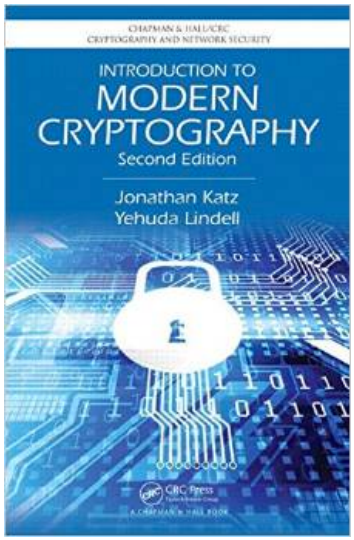
Intro to Cryptography

CS 485/585: Cryptography

- **Meetings:** T/Th 4:40 – 6:30 pm @ FAB 10
- **Instructor:** Fang Song (fang.song@pdx.edu)
- **Office Hours:** T/Th 10:30 – 11:30 am or by appointment @ FAB 120-07
- **Course webpage**: http://www.fangsong.info/teaching/w17_4585_icrypto/
  - Check regularly for updates and announcements.
  - **Resource** page contains useful materials
- **TA:** Nate Launchbury (njl2@pdx.edu). **Office hours**: M 9:30 – 11 am

Text

- **Required**: Katz-Lindell [KL] 2nd edition

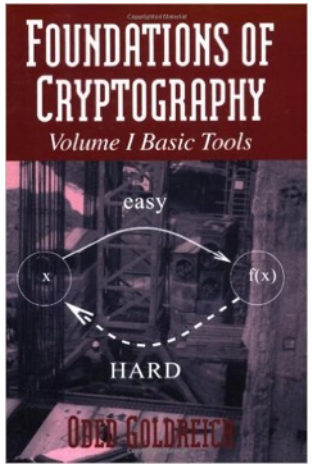


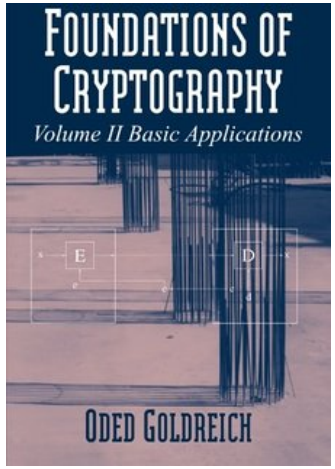
- Goldreich: Foundations of Cryptography Vol I, II

- Boneh-Shoup [BS]: grad course in applied crypto (https://crypto.stanford.edu/~dabo/cryptobook/)

A Graduate Course in Applied Cryptography

Dan Boneh and Victor Shoup





What is this course about?

A **CONCEPTUAL** & **THEORETICAL** tour to modern cryptography

YES

- Ideas
- Formal methods to security
  - How to define, construct, reason/prove?

NO

Important, but not our focus

- Implementations
- How to secure your systems & network?

Goal: cryptographic way of thinking

- Solid foundation for real-world security
- Appreciate the intellectual beauty
- Beneficial far beyond cryptography

Prerequisite

Comfortable with **READING** & **WRITING** *mathematical* **proofs**

Did you *enjoy* CS 311, CS 350 or equiv.?

- Reasonable math background helpful
  - Basic probability theory, linear algebra, number theory, algorithm analysis …
  - Review materials on course webpage

"Big-Oh notation, random variable, independence, matrices, eigenvalue, congruence…"

??? No idea ➔ think again

- Programming not required

Main topics

1. **Overview**. (1 week)
   - History, principles of modern crypto, perfect secrecy

2. **Private-key** *(symmetric)* crypto (4 weeks)
   - Encryption, message authentication, hash functions

3. **Public-key** *(asymmetric)* crypto (3 weeks)
   - Key-exchange, public-key encryption, digital signatures

4. **Selected topics** (2 weeks)
   - Block-chain, zero-knowledge, fully-homomorphic encryption, quantum computing & quantum-safe crypto

Policy: grade

- 40%: 4 homework assignments
- 20%: 4 closed-book in-class quizzes
- 30%: final exam
  - You can prepare 2 double-sided letter-size notes
- 10%: participation

Policy: homework

- Turn in hard copies before lecture on due date
  - **Late** homework: penalty of 30% (<1 day), 50% (1-2 days), 80% (2 - 3days), 100% (> 3 days)
- Your solutions must be intelligible
  - Be ready to explain your soln's verbally, and convince others & **yourself**
- **Collaboration** is encouraged!
  - Form study groups of $\leq$ 3 people, brainstorm etc. (DO IT! VERY HELPFUL!)
  - Write up your solutions independently
  - Mark the names of collaborators on each problem
- Tips: start early!

Policy

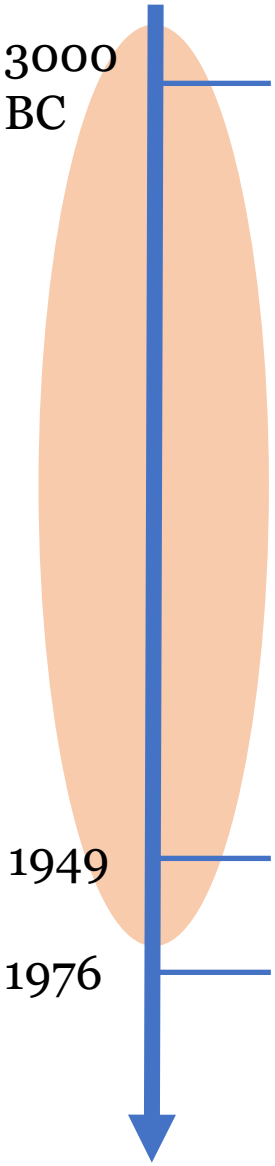
Academic Integrity

- Follow the PSU Student Conduct Code
  - http://www.pdx.edu/dos/codeofconduct
- Any academic dishonesty will be taken seriously!

- Academic accommodations
  - Register with the Disability Resource Center (https://www.pdx.edu/drc/) and contact me

Today

1. History
2. Principles of modern Cryptography
3. Symmetric encryption: Perfect secrecy
4. Review: mathematical background

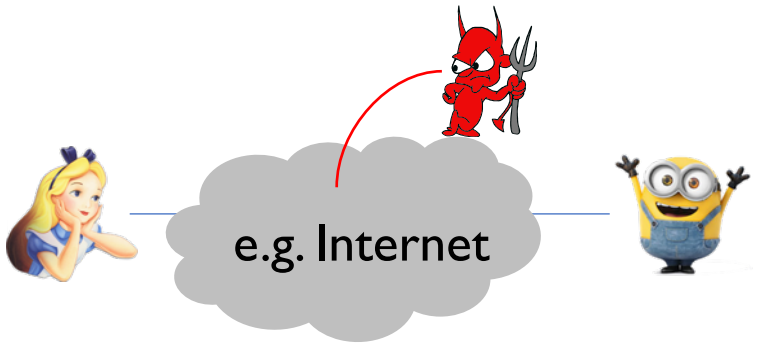


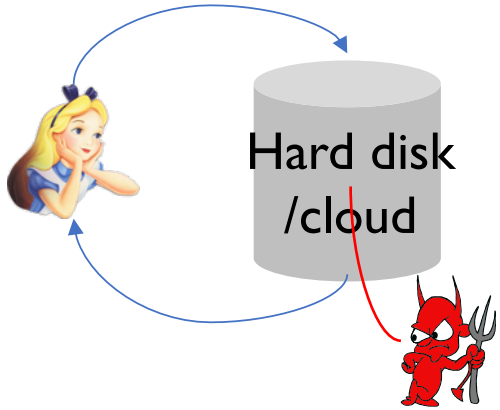
Cryptography is the **art** of writing or solving *codes (ciphers).*

-- *Concise Oxford English Dictionary*

… for *military* activity and *gossip*

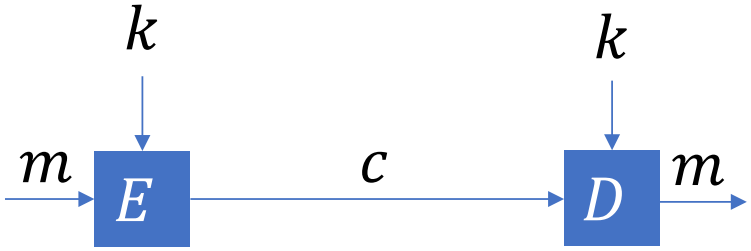
- Two typical scenarios of "secret writing"





The setting of private-key encryption

- Call a cipher an *encryption* scheme



- Syntax of private-key (symmetric) encryption
  - k: private-key (shared-/secret-key)
  - m: plaintext
  - c: ciphertext
  - E: encryption algorithm
  - D: decryption algorithm

Ceasar's cipher

- Example

cryptoisfun

fubswlvixq

- Rule

a b c d … y z

d e f g … b c

$\{a,\ldots,z\} = \{0,\ldots 25\}$

- $k = 3$ fixed
- $E(m_i) = (m_i + 3)\ mod\ 26$

- What if you know it's encoded by C's cipher?

Kerckhoffs' principle

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

i.e. security rely solely on *secrecy* of the *key*

1. Easier to **keep secrecy of & change** a *short* key than *complex* enc/dec algorithms
2. More trust via public scrutiny of the crypto-scheme
3. Easier to maintain at large-scale

Only use *standardized* cryptosystems whenever possible!

Extension: shift & sub cipher

- **Shift** cipher

$$\{a,\ldots,z\} = \{0,\ldots 25\}$$

  - Pick $k \in \{0,..25\}$ and keep it secret
  - $E(m_i) = (m_i + k)\ mod\ 26$
  - But, only 26 possibilities, *brute-force* search!
  - Ex. decipher "dszqupjtgvo"

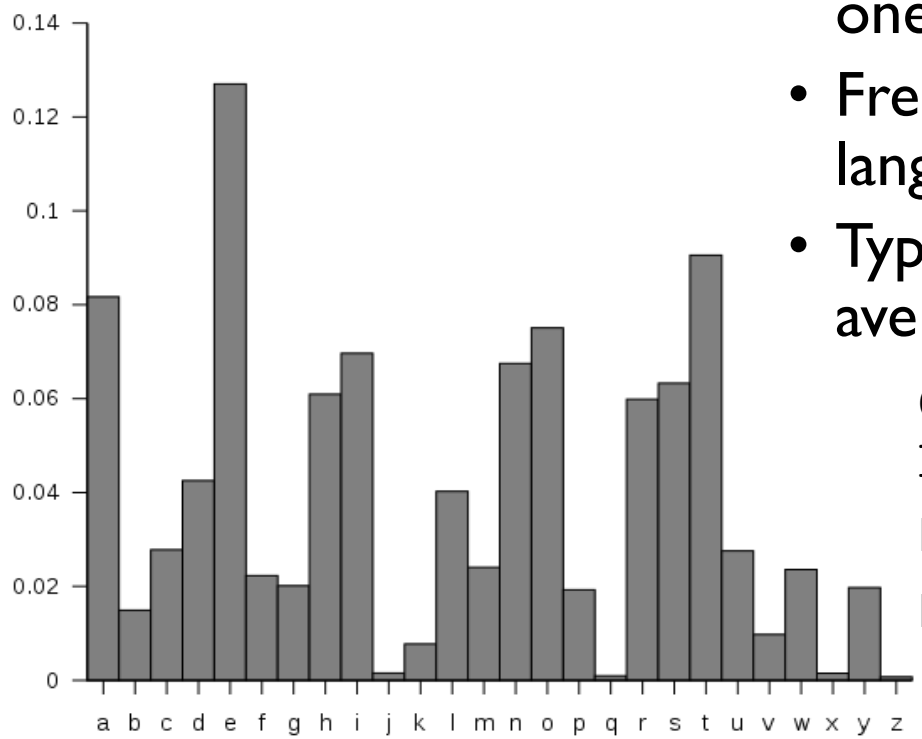
- **Substitution** cipher
  - key k defines a *permutation* on the alphabet
  - Ex. encrypt "cryptoisfun" under the following key

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

  - How many Possible keys? $26! \approx 2^{88}$

Attacking substitution cipher

Frequency analysis



- Cipher preserves frequency: one-to-one correspondence
- Frequency distribution in English language is publicly **known**
- Typical sentences close to average frequency distribution

GSD UVPSDH CDGSFA CLWG QFG ED HDYLVHDA GF ED WDUHDG

D - 18, G - 14, Q - 9,

Ex. Decipher it by hand or online solver

NB: Frequency analysis can automate and improve the attack on shift cipher too

Poly-alphabetic shift cipher

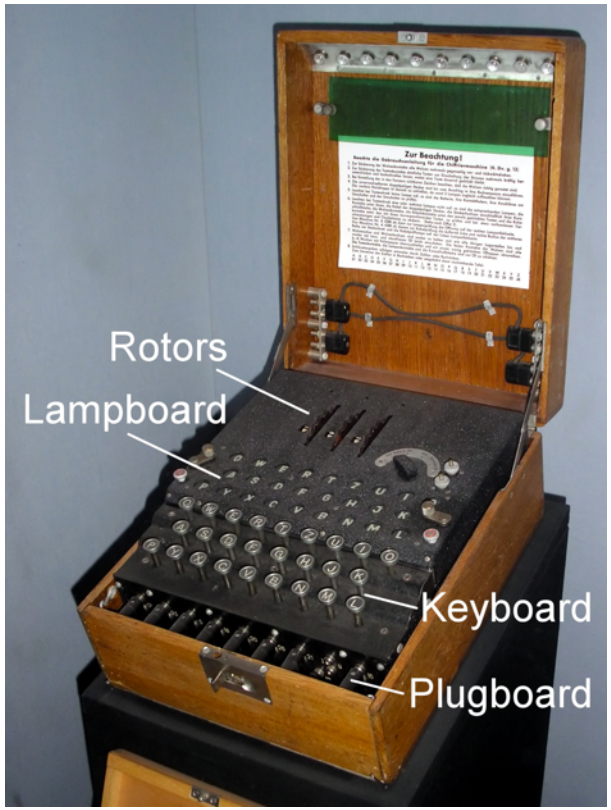
- a.k.a The *Vigenère* cipher
  - Key: a *string* of letters
  - Encrypt:

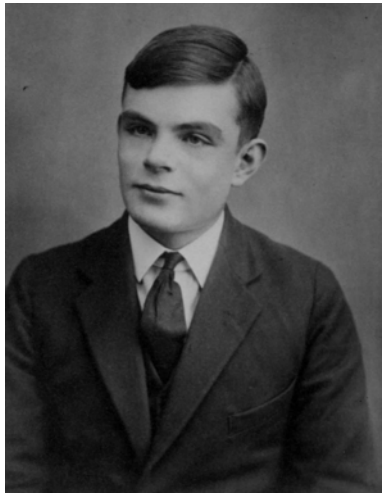
Key:	psu psu psu psu psu psu
Plaintext:	cry pto isf una ndc ool
Ciphertext:	rjs eli xkz jfu cvw dgf

  - Considered "unbreakable" for >300 years
- Attack
  - Key length known: frequency analysis on each sub-string
  - How to determine the key length? Read [KL]

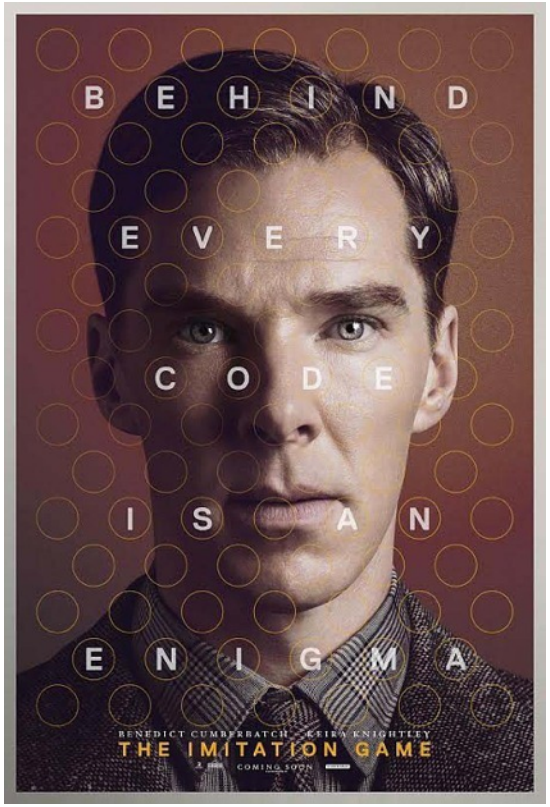
Poly-alphabetic substitution cipher

- Example: Enigma machine in WWII





Alan Turing



Source: imdb

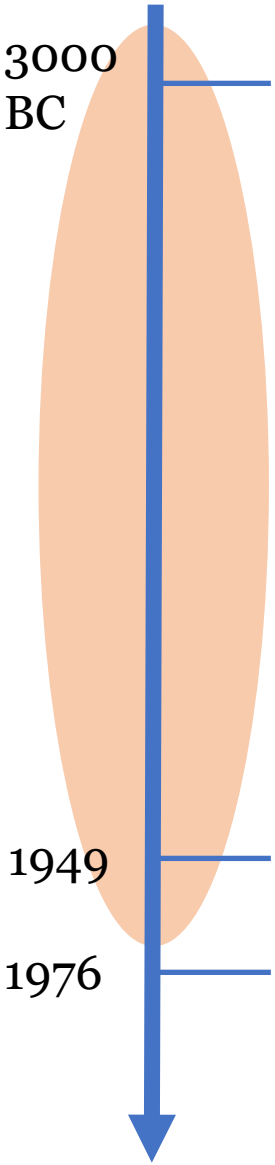
- Attack: same principle as before

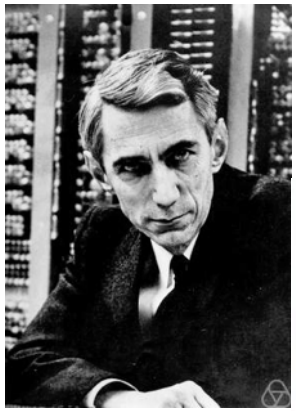
Lessons from historical ciphers

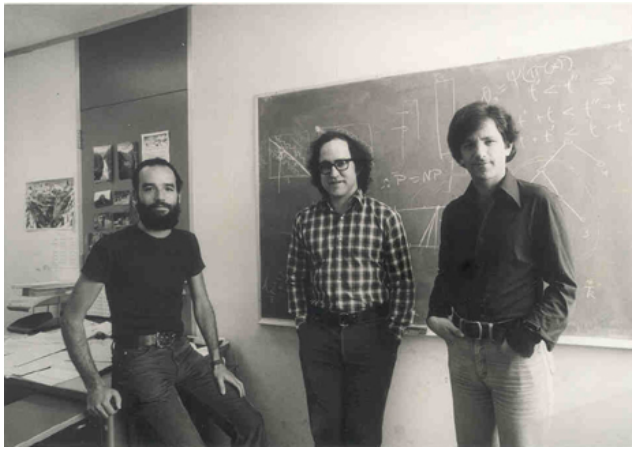
- Designing secure ciphers is hard
- *Looks* unbreakable ≠ *is* unbreakable
- Intelligent but mostly an *art*

Not clear about

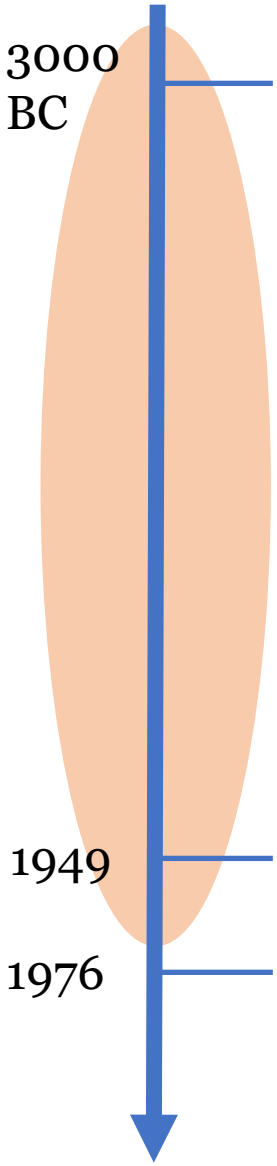
- is a cipher secure?
- under what circumstances?
- what'es "secure" even mean?







Revolution of Modern Cryptography



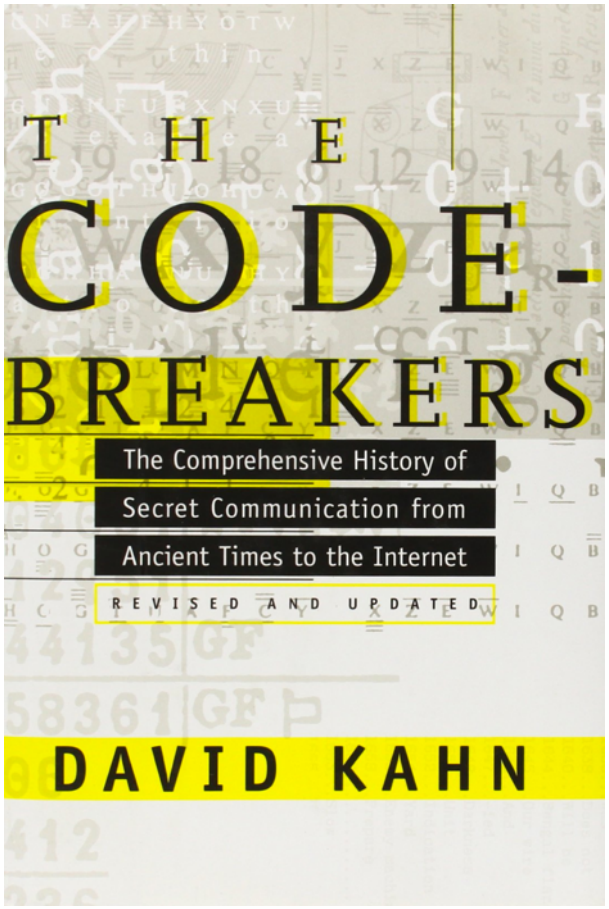
Cryptography is the *art* of writing or solving *codes (ciphers).*

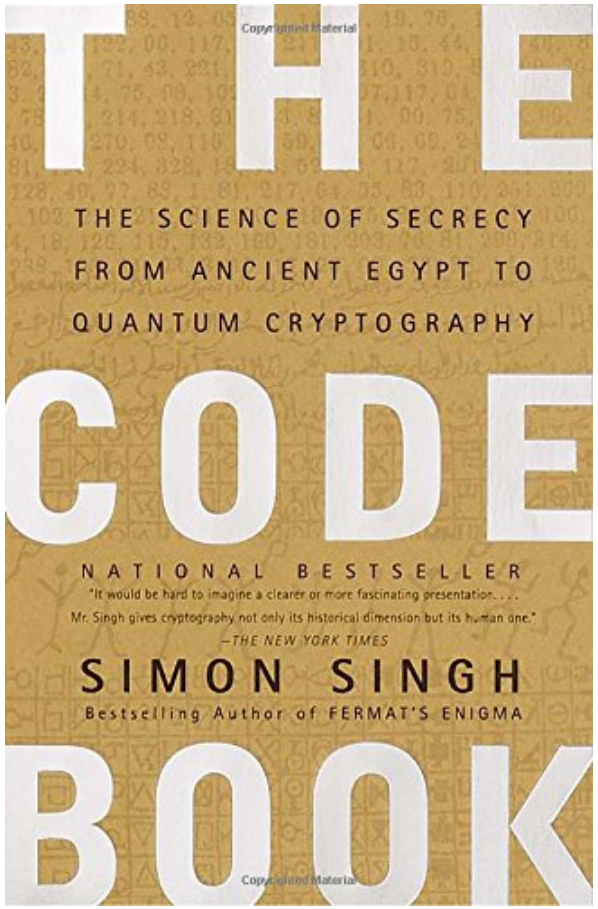
-- *Concise Oxford English Dictionary*

1. Much more of a **science** than *art*
   - security via mathematics
2. Much more than "*secret writing*"
   - **Public-key cryptography**, …

Modern Cryptography involves the study of *mathematical* techniques for securing {*digital information, systems and computations*} against adversarial attacks – [*KL]*

Good reads on crypto history





Source: amazon.com

Today

1. History
2. Principles of modern Cryptography
3. Symmetric encryption: Perfect secrecy
4. Review: mathematical background

Principles of modern crypto

1. Formal **definitions** of security
   - What is the "security" that you want to achieve exactly?
   - Guide the design and properly evaluate a construction
   - Know better what you need

Principles of modern crypto

2. Rigorous **proofs** of security
   - The only known method to reason against (infinitely) many possible attacking strategies
   - Never rely on your pure impression

Principles of modern crypto

3. Precise **assumptions**
   - Unconditional security is often impossible to prove
   - Be **precise**, for validating and comparing schemes

How about:

Assume "my construction satisfies the definition"	*VS.*	Assume "factoring 1000-bit integer cannot be done in less than 1000 steps" ✓

- More confidence in well-studied than ad hoc assumption
- Easier to test simple-stated assumptions
- Modularity: replace a building block when needed

Principles of modern crypto

1. Formal **definitions** of security
2. Rigorous **proofs** of security
3. Precise **assumptions**

Recall: historical crypto is unclear about

- is a cipher secure?
- under what circumstances?
- what'es "secure" even mean?

Provable security & real-world

A scheme has been proven secure

$\not\Rightarrow$

it's secure in the real world

- Is the definition right?
  - Not match what is needed
  - Not capture attackers' true abilities
- Is the assumption meaningful?

But you (the *designer/defender*), instead of *attackers*, have more in charge

- You'll know exactly where to look at and improve: refine defs, test assumptions, …

Formal approach is powerful

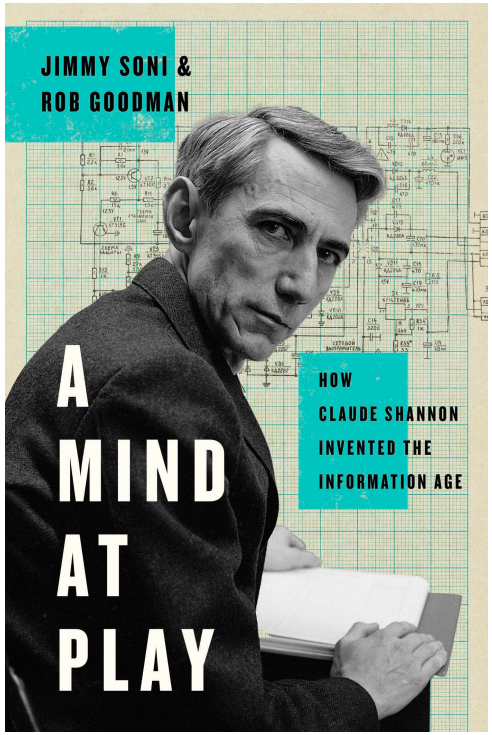
A Symbolic Analysis of Relay and Switching Circuits

By CLAUDE E. SHANNON
ENROLLED STUDENT AIEE

Most important Master's thesis in 20th century"

- Howard Gardner

- Electrical circuit →Boole algebra (for logic)
  - Designing elec. circuit: art to science
- Boole algebra → Electrical circuit
  - Circuits can *decide* T/F: AI
  - Circuits can *compute*: digital computers



Source: amazon.com

Today

1. History
2. Principles of modern Cryptography
3. Symmetric encryption: Perfect secrecy
4. Review: mathematical background

Today

1. History
2. Principles of modern Cryptography
3. Symmetric encryption: Perfect secrecy
4. Review: mathematical background

Sets

- Basic notations
  - $\exists$ "exist", $\forall$ "for all",
  - $\{0,1\}$ a bit, $\{0,1\}^k$ string of k bits
  - s.t. "such that", iff. "if & only if"
  - $:=$ assign or define

- (Discrete) Sets: A, B, X, Z usually capital letters
  - Union "$\cup$", intersection "$\cap$", subtraction "$A \backslash B$"
  - $|A|$ denotes the size of A
  - $\mathbb{N}$: natural numbers
  - $\mathbb{Z}$: set of integers

Functions

- Functions $f: A \to B$
  - One-to-one (injective) → $|A| \leq |B|$
  - Onto (surjective) → $|A| \geq |B|$
  - One-one correspondence (bijective) → $|A| = |B|$
  - $\log x$ base 2, $\ln x$ natural logarithm base e
- **Asymptotic notation**
  - Let $f, g: \mathbb{N} \to \mathbb{N}$ be functions. $f = O(g)$ iff. $\exists c$ constant s.t. $f(n) \leq c \cdot g(n)$ for sufficiently large n.
  - $f = o(g), f = \Omega(g), f = \omega(g)$
  - Review if necessary: Chapter 3 Introduction to Algorithms, By Cormen, Leiserson, Rivest and Stein.

Probability

- (Discrete) Sample space Ω
  - set of all possible outcomes of a random experiment
- Probability distribution $p: \Omega \to \mathbb{R}$,
  - $\forall \omega \in \Omega, \Pr(\omega) \coloneqq p_\omega \geq 0; \quad \sum_{\boldsymbol{\omega} \in \Omega} p_\omega = 1.$
- **Event** $E \subseteq \Omega$: a subset of the sample space
  - $\Pr(E) = \sum_{\omega \in E} p_\omega$: probability of an event occurs
  - $\bar{E} \coloneqq \Omega \backslash \mathrm{E}$ complement event, $\Pr(\bar{E}) = 1 - \Pr(E)$
- Ex. Roll a fair dice
  - $\Omega = \{1,2,3,4,5,6\}$
  - $p_\omega = \frac{1}{6}, \omega = 1, \dots, 6$; i.e., *uniform* distribution over Ω.
  - $E = \{1,3,5\}$ dice being odd, & $\Pr(E) = 1/2$

Probability cont'd

- Union bound

Let $E, F \subseteq \Omega$ be two events. Then $\Pr(E \cup F) \leq \Pr(E) + \Pr(F)$.

- **Conditional** probability
  - Assume $\Pr(A) > 0$. $\Pr(B|A) \coloneqq \Pr(A \cap B)/\Pr(A)$.

Bayes' theorem

Let $E, F \subseteq \Omega$ be two events and $\Pr(F) > 0$.

Then $\Pr(E|F) = \frac{\Pr(F|E) \cdot \Pr(E)}{\Pr(F)}$.

- **Independence**
  - Events A, B are independent iff. $\Pr(B|A) = \Pr(B)$.

i.e. $\Pr(A \cap B) = \Pr(A) \cdot \Pr(B)$

Probability cont'd

- Random variable $X: \Omega \to S$
  - S usually $\mathbb{R}$ (real num.); i.e. assign each outcome a number
  - In CS, often $S \subseteq \{0,1\}^k$, bit strings
  - "$X = x$" is the event $E = \{\omega \in \Omega: X(\omega) = x\}$
  - Independent random variables: X, Y are indep. iff. for all possible x and y, events $X = x$ and $Y = y$ are indep.
- **Expectation**: a weighed average
  - $\mathbb{E}[X] = \sum_{x \in S} \Pr(X = x) \cdot x$
  - **Linearity**: $\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y]$ (need no indep.)
- Ex. Ω = roll 4 dices indep,
  - Let $X(\omega)$ be the sum of 4 rolls; $S = \mathbb{Z}^+$
  - Ex. $\mathbb{E}[X] = 3.5 * 4 = 14$ (EX)

Algorithms

- Deterministic algorithm
  - A procedure that computes a function on an input
  - Computational models: circuit, Turing machine
- Randomized (probabilistic) algorithm
  - Unbiased random bits as auxiliary input
  - Model: Turing machine w. an extra uniformly random tape
  - Often more efficient (w. small error)
  - Ex. Quicksort with random pivot n^2 vs. $nlogn$

Will pick up more along the way:

linear algebra, number theory, group theory

…

- Useful references
  - MIT 6.042 Mathematics for computer science
  - A Computational Introduction to Number Theory and Algebra by Victor Shoup.
  - Probability and Computing: Randomized Algorithms and Probabilistic Analysis, by Michael Mitzenmacher & Eli Upfal.