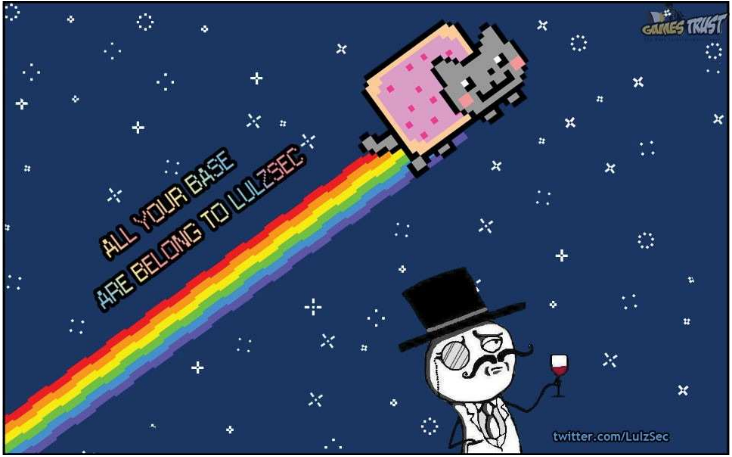
New directions in security by obscurity

Dusko Pavlovic

Royal Holloway, Oxford and Twente

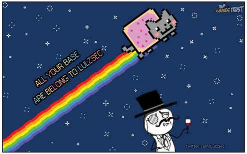
September 2011

Notation: Attack



Assumption: Security reduction

Suppose that you are given a system C and a proof



C

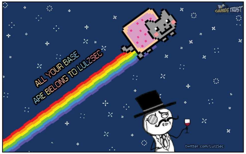
$$\Downarrow$$

$$P = NP$$

Would you consider system C secure?

Assumption: Security reduction

Suppose that you are given a system $\mathcal{D}$ and a proof



$\mathcal{D}$

$$\Downarrow$$

$$P \neq NP$$

Would you consider system $\mathcal{D}$ secure?

There is security by obscurity in cryptography

Theorem

System $\mathcal{D}$ is secure enough to protect an account with $1,000,000

Proof.

Proving $P \neq NP$ yields $1,000,000 from Clay Institute. $\square$

Directions

Background: What is obscurity in security?

Approach: Refining attacker models

X-Direction: Security by epistemic game theory

Y-Direction: Security by algorithmic information theory

Summary: Adaptive attacker meets adaptive defender

(Disclaimer)

I am **not** advocating or criticizing

- property rights over code or algorithms
- limitations of surveillance disclosure
- cryptography export controls
- ...

The policy issues are not addressed in this research.

I formalize "obscurity" as a technical concept,
and discuss its utility as a security resource.

Directions

Background: What is obscurity in security?

Approach: Refining attacker models

X-Direction: Security by epistemic game theory

Y-Direction: Security by algorithmic information theory

Summary: Adaptive attacker meets adaptive defender

What is security by obscurity?

Kerckhoffs' Principle

"The system must not be required to be secret, and it must be able to fall into the hands of the enemy without inconvenience."

Jean Guillaume Auguste Victor François Hubert Kerckhoffs

What is security by obscurity?

Shannon’s Maxim

"The enemy knows the system."

Claude Shannon

Secure key *vs* obscure system

Lock can only be opened using the correct key

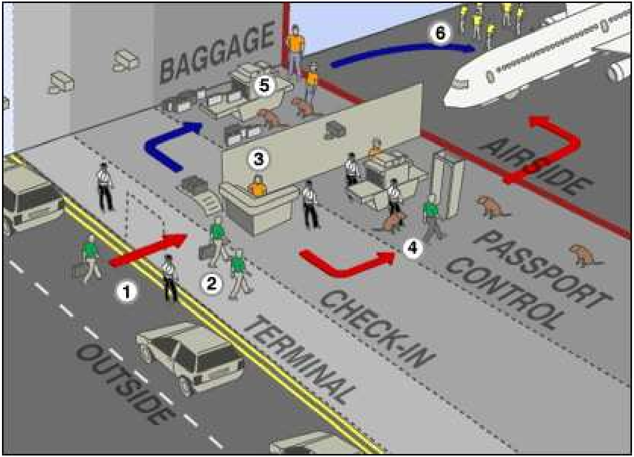
Secure key *vs* obscure system

. . . and **not** by breaking the system

Outside cryptography

there are systems with no key

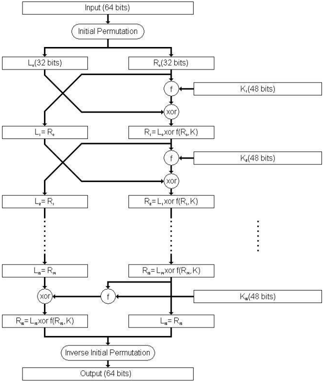
Outside cryptography



there is not much more to hide except the system

In cryptography

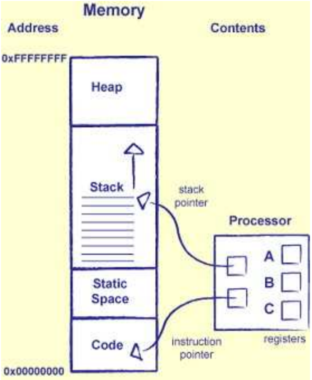
- keys = data
- system = program



In computation

(Gödel, Von Neumann, Kleene)

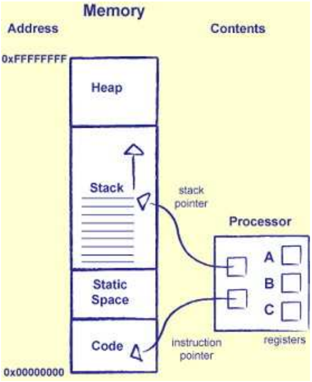
- keys = data = program
- system = program = data



In computation

(Gödel, Von Neumann, Kleene)

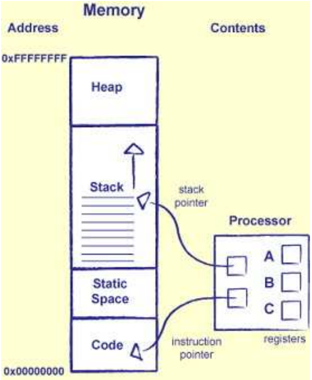
- keys = data = program
  - data $\rightsquigarrow$ encrypted
- system = program = data
  - programs $\rightsquigarrow$ obfuscated



In computation
(Gödel, Von Neumann, Kleene)

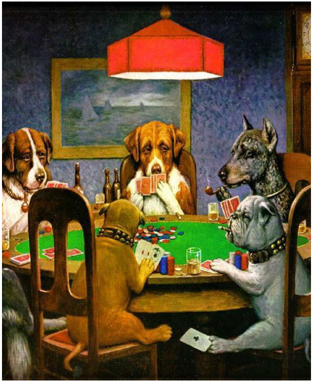
- keys = data = program
  - data $\rightsquigarrow$ encrypted
- system = program = data
  - programs $\rightsquigarrow$ obfuscated

Theorem [Barak et al]
Obfuscators do not exist.



In poker

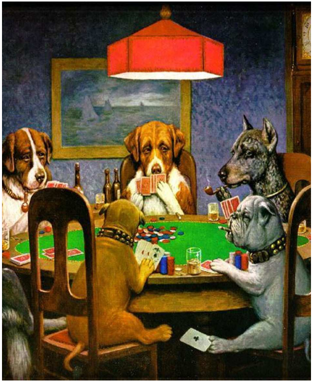
- keys = hands of cards
- system = tactics



In games

(Von Neumann-Morgenstern, Harsanyi, Aumann. . .)

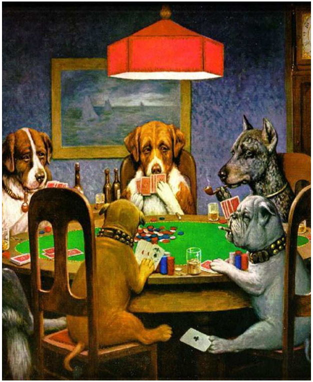
- keys = players' positions
- system = players' types



In games

(Von Neumann-Morgenstern, Harsanyi, Aumann. . .)

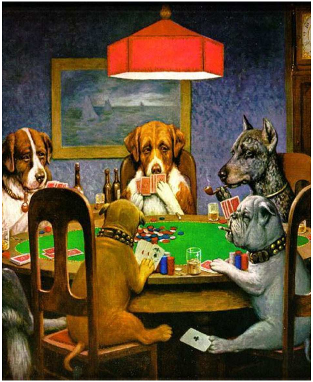
- keys = players' positions
  - (im)perfect information
- system = players' types
  - (in)complete information



In games

(Von Neumann-Morgenstern, Harsanyi, Aumann. . .)

- keys = players' positions
  - (im)perfect information
- system = players' types
  - (in)complete information

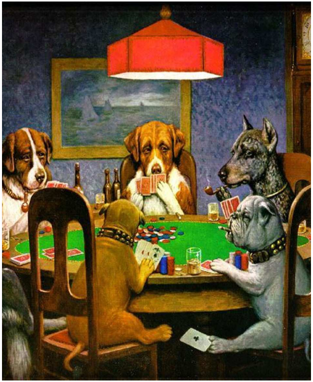


Kerckhoffs' Principle
Security is a game of imperfect information.

In security games
(Kerckhoffs, Shannon)

- keys $\leftsquigarrow$ cryptanalysis
  - hard
- system $\leftsquigarrow$ decompilation
  - easy

Kerckhoffs' Principle
Security is a game of imperfect information.



Directions

Background: What is obscurity in security?

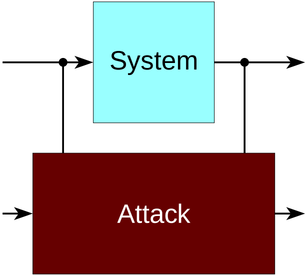
Approach: Refining attacker models

X-Direction: Security by epistemic game theory

Y-Direction: Security by algorithmic information theory

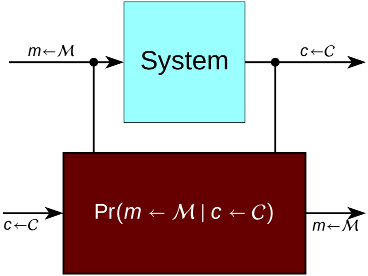
Summary: Adaptive attacker meets adaptive defender

Security is a game of information



Shannon's attacker: computationally unbounded

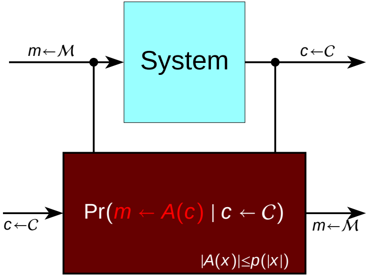
(omnipotent computer)



If a source conveys some information,
the attack will extract that information.

Diffie-Hellman's attacker: computationally bounded

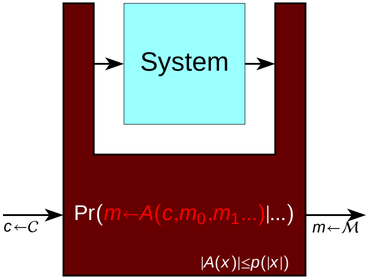
(real computer)



Public key determines the corresponding private key, but the attacker cannot compute one from the other.

Adaptive attacker: queries and controls the system

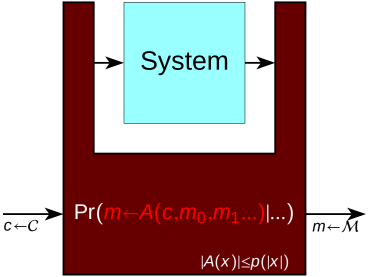
(still a real computer computer)



If there is a vulnerability,
an attack algorithm will use it.

Adaptive attacker: queries and controls the system

(still a real computer computer)

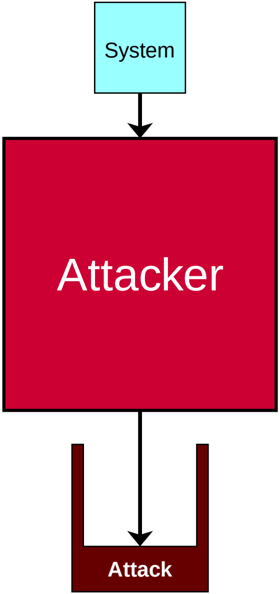


If there is a vulnerability,
an attack algorithm will use it.

But where do attack algorithms come from?

Kerckhoffs' attacker: logically unbounded

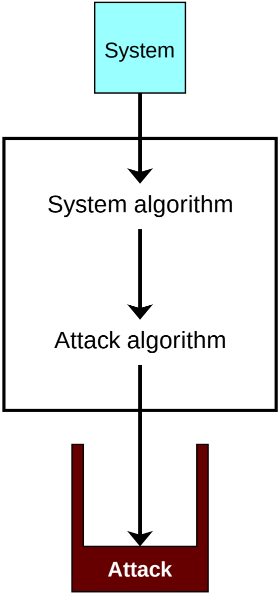
(omnipotent programmer)



If there is an attack,
the attacker will find it.

Kerckhoffs' attacker: logically unbounded

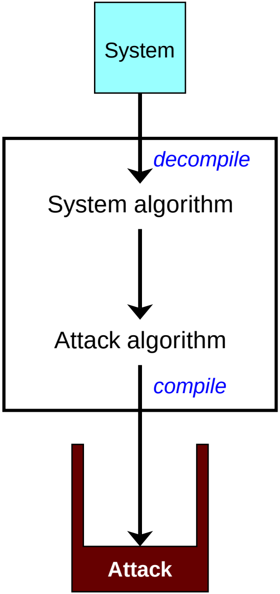
(omnipotent programmer)



If an attack exists,
the attacker will find it

Kerckhoffs' attacker: logically unbounded

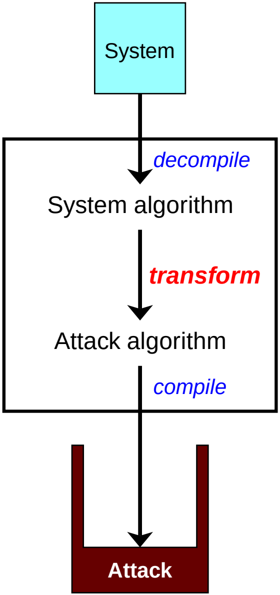
(omnipotent programmer)



If an attack exists,
the attacker will find it.

Kerckhoffs' attacker: logically unbounded

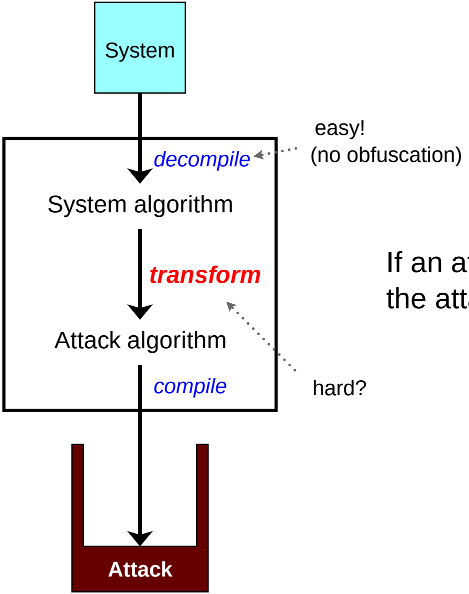
(omnipotent programmer)



If an attack exists,
the attacker will find it

Kerckhoffs' attacker: logically unbounded

(omnipotent programmer)



If an attack exists,
the attacker will find it

Two directions

- improve adaptation of system to attack
- hinder adaptation of attack to system

Two directions

- improve adaptation of system to attack
  - use **epistemic game theory** in security
- hinder adaptation of attack to system
  - use **algorithmic information theory** in security

Directions

Background: What is obscurity in security?

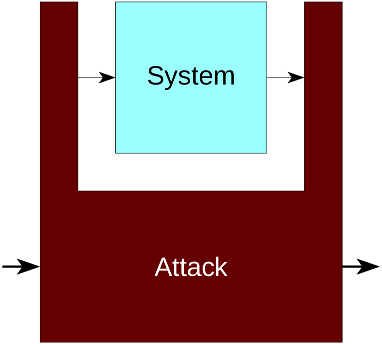
Approach: Refining attacker models

X-Direction: Security by epistemic game theory

Y-Direction: Security by algorithmic information theory

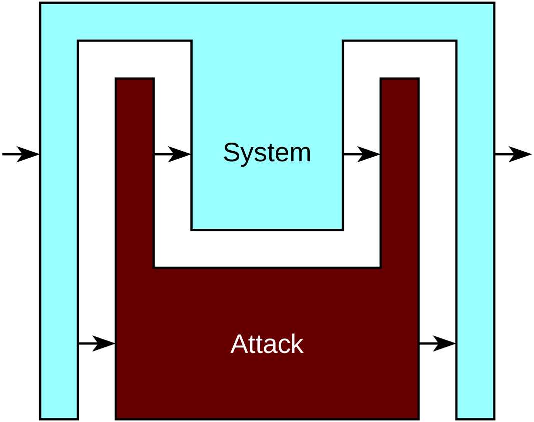
Summary: Adaptive attacker meets adaptive defender

X-Direction



If the attacker queries the system

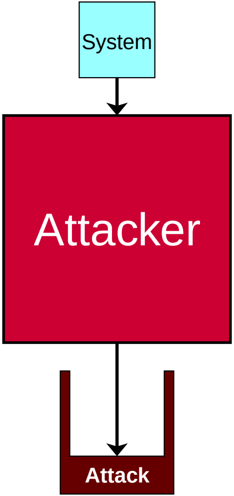
X-Direction



If the attacker queries the system
then the system should query the attacker

Adaptive attacker

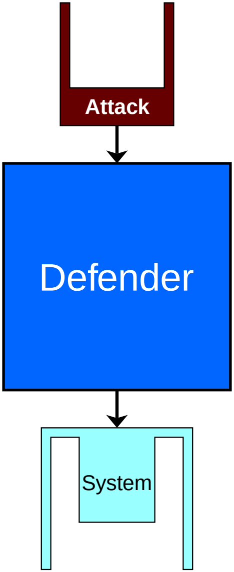
(logically limited)



If there is an easy attack, the attacker will find it.

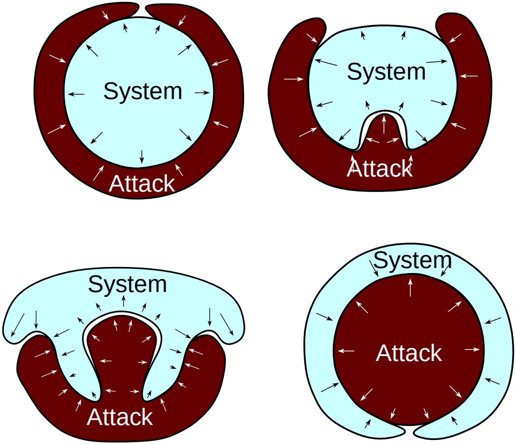
...should be met by an adaptive defender

(logically limited)



If there is an easy defense the defender will find it.

From fortification to adaptation



Obscurity is a problem and a tool.

Directions

Background: What is obscurity in security?

Approach: Refining attacker models

X-Direction: Security by epistemic game theory

Y-Direction: Security by algorithmic information theory

Summary: Adaptive attacker meets adaptive defender

Y-Direction

Take into account attacker's logical limitations.

power	*unbounded*	*bounded*
computational	Shannon	Diffie-Hellman
rationality	Cournot	Simon
logical	Kerckhoffs	?????

Y-Direction

Take into account attacker's logical limitations.

power	*unbounded*	*bounded*
computational	Shannon	Diffie-Hellman
rationality	Cournot	Simon
logical	Kerckhoffs	Bennett?

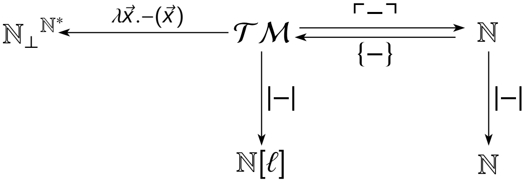
Y-Direction

Take into account attacker's logical limitations.

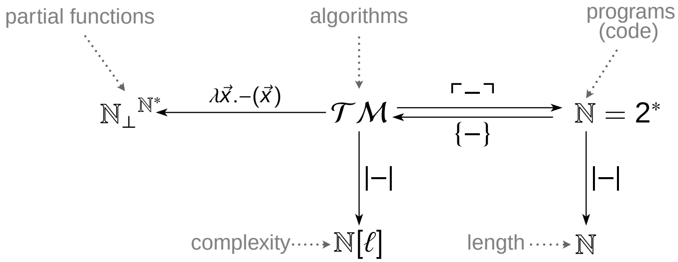
power	*unbounded*	*bounded*
computational	Shannon	Diffie-Hellman
rationality	Cournot	Simon
logical	Kerckhoffs	Bennett?

$$\frac{\text{computational complexity}}{\text{secrecy}} = \frac{\text{logical complexity}}{\text{obscurity}}$$

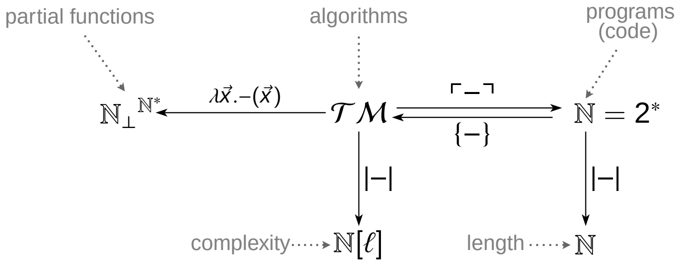
Algebra of algorithms



Algebra of algorithms



Algebra of algorithms



- programs represent algorithms

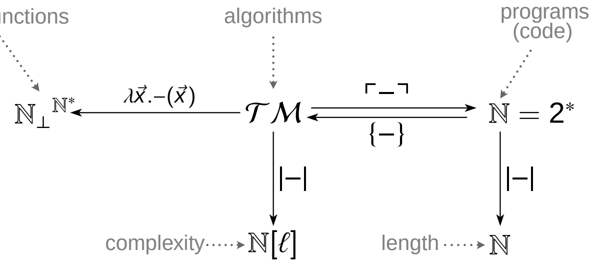
$$\ulcorner\{p\}\urcorner = p \qquad \{\ulcorner M \urcorner\} = M$$

Security by obscurity

Dusko Pavlovic

Background
Approach
X-Direction
Y-Direction
Summary

Algebra of algorithms



- programs represent algorithms

$$\ulcorner\{p\}\urcorner = p \qquad\qquad \{\ulcorner M\urcorner\} = M$$

- there is a *Universal Turing Machine* $U \in \mathcal{TM}$, such that for all $M \in \mathcal{TM}$ and all $\vec{x} \in \mathbb{N}^*$ holds

$$U(\ulcorner M\urcorner, \vec{x}) \quad \doteq \quad M(\vec{x})$$

Assumptions

- $\mathbb{N}$ is a partial combinatory algebra
- $\mathcal{T}\,\mathcal{M}$ are self-delimiting (i.e. the codes are prefix-free)

Algorithmic distance

Definition

A program $p \in \mathbb{N}$ is (a, b)-informative if $\{p\}(a) = b$.
Abbreviate $(\langle\rangle, a)$-informative to a-informative

Definition

Algorithmic distance between $a, b \in \mathbb{N}$ is the length of the shortest (a, b)-informative program

$$C(a,b) \quad = \quad \bigwedge_{\{p\}(a)=b} |p|$$

Algorithmic complexity

Definition (Solomonoff, Kolmogorov)

Algorithmic complexity of $a \in \mathbb{N}$ is the length of the shortest a-informative program

$$C(a) = \bigwedge_{\{p\}()=a} |p|$$

Logical complexity

Definition (~C.H. Bennett)

Logical complexity of $a \in \mathbb{N}$ is the complexity of the simplest *a-informative* program

$$D(a) \quad = \quad \bigwedge_{\substack{\{p\}()=a \\ C(p)=|a|}} |\{p\}|$$

Logical depth

Remarks

- Logical depth measures complexity of evolutionary processes as computational processes.
- Logical depth of an organism is the time it takes it to evolve
  - A virus may be computationally simple, but logically deep
- PRIMES is computationally simple but logically deep

Logical distance

Definition

Logical distance of $a, b \in \mathbb{N}$ is the complexity of the simplest (a, b)-*informative* program

$$D(a,b) = \bigwedge_{\substack{\{p\}(a)=b \\ C(a,b)=|p|}} |\{p\}|$$

Logical distance

Remark

D is almost a metric

$$\begin{aligned} D(a,a) &= 0 \\ D(a,b) + D(b,c) &\geq D(a,c) \end{aligned}$$

Logical distance

Remark

D is almost a metric

$$\begin{aligned} D(a,a) &= 0 \\ D(a,b) + D(b,c) &\geq D(a,c) \end{aligned}$$

in fact a *quasi-pseudo-metric*

$$\begin{aligned} D(a,b) &\neq D(b,a) \\ D(a,b) = 0 &\not\Rightarrow a = b \end{aligned}$$

Logical distance

Remark

D is almost a metric

$$\begin{aligned} D(a,a) &= 0 \\ D(a,b)+D(b,c) &\geq D(a,c) \end{aligned}$$

in fact a *quasi-pseudo-metric*

$$\begin{aligned} D(a,b) &\neq D(b,a) \\ D(a,b)=0 &\not\Rightarrow a=b \end{aligned}$$

provided that the constants are factored out

$$D \quad : \quad \mathbb{N}\times\mathbb{N}\to\mathbb{N}[\ell]\twoheadrightarrow\mathbb{N}[\ell]/\mathbb{N}$$

Background

- **Ray Solomonoff (1960):**
  Inductive interpretation (explanation) of a given observation is the smallest program that generates it.

Background

- **Ray Solomonoff (1960):**
  Inductive interpretation (explanation) of a given observation is the smallest program that generates it.
- **A. Kolmogorov (1965), G. Chaitin (1968):**
  Complexity of a bitstring is the length of the simplest program that outputs it.

Background

- **Ray Solomonoff (1960):**
  Inductive interpretation (explanation) of a given observation is the smallest program that generates it.

- **A. Kolmogorov (1965), G. Chaitin (1968):**
  Complexity of a bitstring is the length of the simplest program that outputs it.

- **Charles H. Bennett (1981):**
  Logical depth of an organism is the time complexity of the simplest evolutionary process that leads to it.

Security application

Assure that $D(s,a)$ is large for all attacks a on system s.

Obstacle

- Logical distance is not computable.

Obstacle

- Logical distance is not computable.
  - Chaitin proved Gödel-style incompleteness.

Upshot

- There is security by obscurity, but it is **not provable**.

Upshot

- There is security by obscurity, but it is **not provable**.
  - Kolmogorov: Most bitstrings are random
  - Martin-Löf: Most bitstrings cannot be proven random.

Directions

Background: What is obscurity in security?

Approach: Refining attacker models

X-Direction: Security by epistemic game theory

Y-Direction: Security by algorithmic information theory

Summary: Adaptive attacker meets adaptive defender

Summary

New directions in security by obscurity

- improve adaptation of system to attack
  - use **epistemic game theory** in security
- hinder adaptation of attack to system
  - use **algorithmic information theory** in security

Summary

Obstacles

- complexity of strategies with incomplete information
- incompleteness of theories of logical distance