Formal Methods for Cryptography

Santiago Zanella-Béguelin
santiago@microsoft.com

Microsoft®
Research
Cambridge, UK

2013.03.04–08
18th Estonian Winter School in Computer Science
EWSCS 2013

Cryptography

The goal of cryptography is to secure communications

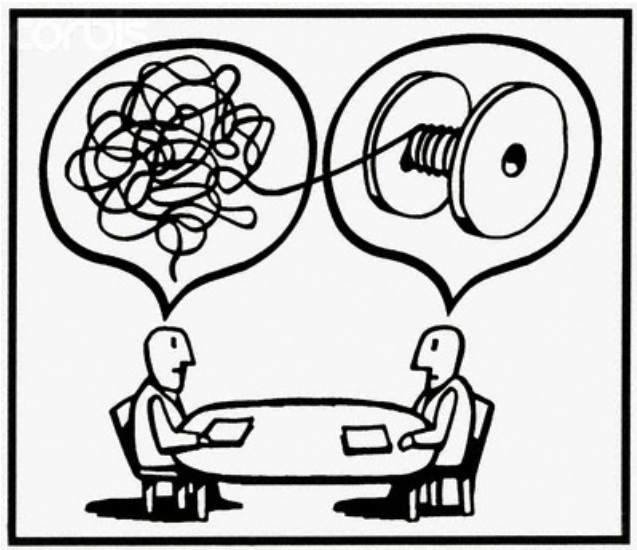
Alice Bob

- authentication: is it Alice?
- confidentiality: did someone else read my message?
- integrity: did someone tamper with my message?
- anonymity: is it Alice? Dunno!
- deniability: I did not say that!
- non-repudiation: you know you said it, I can prove it!

Cryptography

Through the use of primitives:

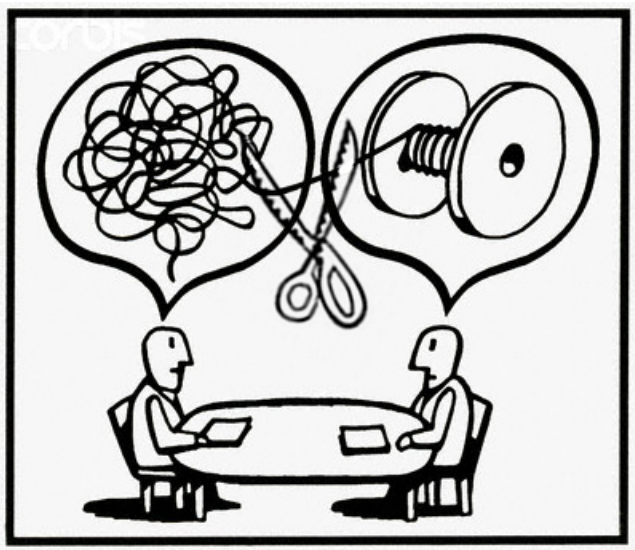
- Symmetric encryption
- Message Authentication Codes
- Hash functions
- Public-key encryption
- Digital signatures
- Zero-knowledge proofs



Cryptography

Many points of failure:

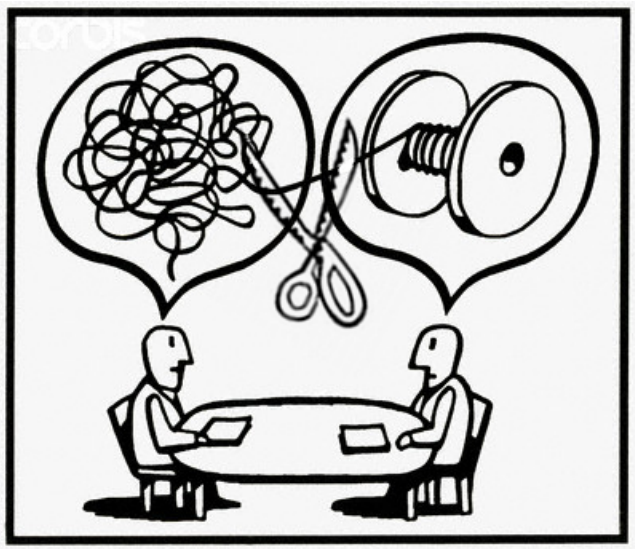
- primitives may not be secure
- protocols may be logically flawed
- implementations may introduce weaknesses
- applications may make an improper use of cryptography
- users may ignore security issues
  (search "RSA PRIVATE KEY" in pastebin)



Cryptography

Many points of failure:

- **primitives may not be secure** $\Leftarrow$ **solutions in this course**
- protocols may be logically flawed
- implementations may introduce weaknesses
- applications may make an improper use of cryptography
- users may ignore security issues
  (search "RSA PRIVATE KEY" in pastebin)



Course Outline

1. Introduction to provable security
   - Proof by reduction
   - Game-based proofs
   - Security goals and assumptions
2. Code-based cryptographic proofs
   - A programming language for games
   - A relational logic for proving equivalence
   - Approximate equivalence
3. Proof techniques
   - Reasoning about failure events
   - Eager/lazy sampling
   - Computer-aided proofs
4. Advanced applications...as time allows
   - Differential privacy
   - Zero-Knowledge proofs
   - Synthesis of encryption schemes

The Art and Science of Cryptography

Formal methods: A tale of two worlds

Symbolic World

$\{(\{(0, K_1)\}_{K_2}, 1)\}_{K_3}$

- Values as symbols
- Primitives as symbolic expressions
- Adversaries as inference engines
- Asymptotic security
- Indirect (computation soundness)
- ProVerif, PCL, AVISPA
- Better suited for protocols?

Computational World

$f(G(r) \oplus (m \| 0^k) \| H(G(r) \oplus (m \| 0^k)) \oplus r)$

- Values as bitstrings
- Primitives as functions on bitstrings
- Probabilistic Polynomial-time Adversaries
- Exact security bounds
- Direct
- CryptoVerif, CPCL, CIL
- Better suited for primitives?

This course is about the computational world

Formal methods: A tale of two worlds

Symbolic World $\{(\{(0, K_1)\}_{K_2}, 1)\}_{K_3}$	**Computational World** $f(G(r)\oplus(m\|0^k)\|H(G(r)\oplus(m\|0^k))\oplus r)$
• Values as symbols	• Values as bitstrings
• Primitives as symbolic expressions	• Primitives as functions on bitstrings
• Adversaries as inference engines	• Probabilistic Polynomial-time Adversaries
• Asymptotic security	• Exact security bounds
• Indirect (computation soundness)	• Direct
• ProVerif, PCL, AVISPA	• CryptoVerif, CPCL, CIL
• Better suited for protocols?	• Better suited for primitives?

This course is about the computational world

Perfect cryptography is not practical (we do not live in a symbolic world)

One-Time Pad achieves perfect secrecy

Recall absorption property of XOR: $b \oplus a \oplus b = a$

- To encrypt a plaintext $m \in \{0,1\}^n$ using key $k \in \{0,1\}^n$, compute $c = k \oplus m$.
- To decrypt a ciphertext $c \in \{0,1\}^n$ under key $k \in \{0,1\}^n$, compute $m = c \oplus k$.

NOT PRACTICAL: a key must be used only once and must be at least as long as the plaintext to be encrypted.

Shannon proved in the 40's that OTP is the essentially the only way to achieve information-theoretic secrecy

Perfect cryptography is not practical (we do not live in a symbolic world)

One-Time Pad achieves perfect secrecy

Recall absorption property of XOR: $b \oplus a \oplus b = a$

- To encrypt a plaintext $m \in \{0,1\}^n$ using key $k \in \{0,1\}^n$, compute $c = k \oplus m$.
- To decrypt a ciphertext $c \in \{0,1\}^n$ under key $k \in \{0,1\}^n$, compute $m = c \oplus k$.

NOT PRACTICAL: a key must be used only once and must be at least as long as the plaintext to be encrypted.

Shannon proved in the 40's that OTP is the essentially the only way to achieve information-theoretic secrecy

Practical cryptography

Modern cryptography is built on the assumption that there exist problems that are hard on the average.

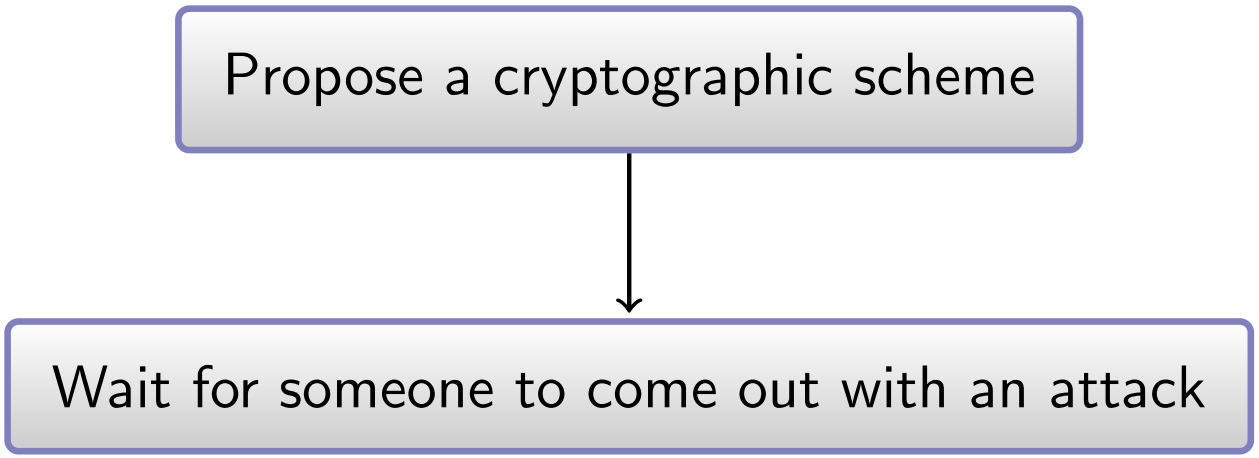
Efficiency is associated with the tasks that can be performed by probabilistic polynomial-time Turing machines (the class $\mathcal{BPP}$).

Thus, modern cryptography is meaningful only if

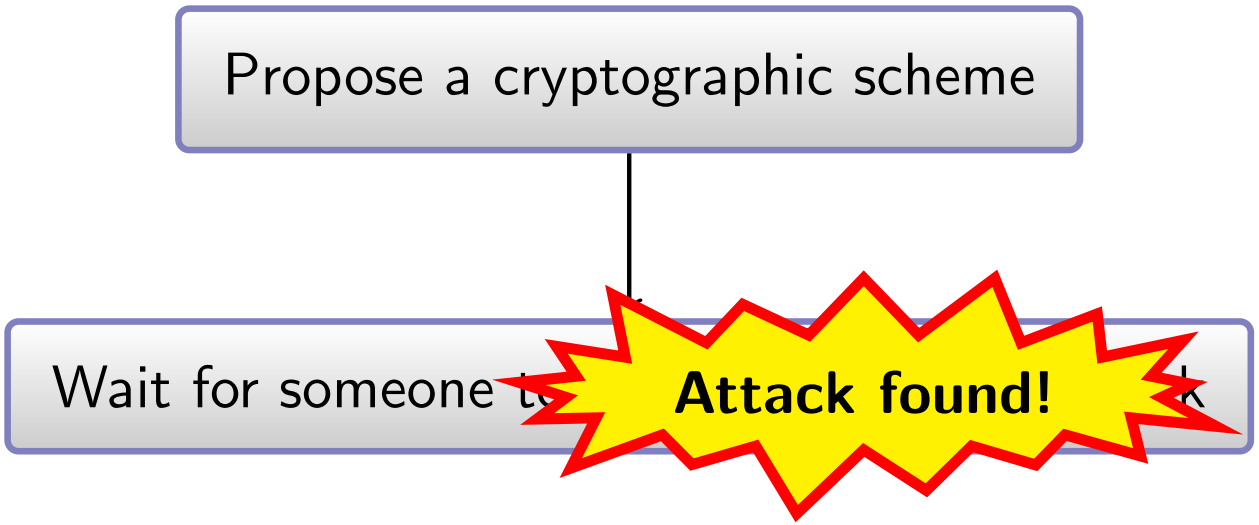
$$\mathcal{NP} \not\subseteq \mathcal{BPP}$$

The art: cryptanalysis-driven design

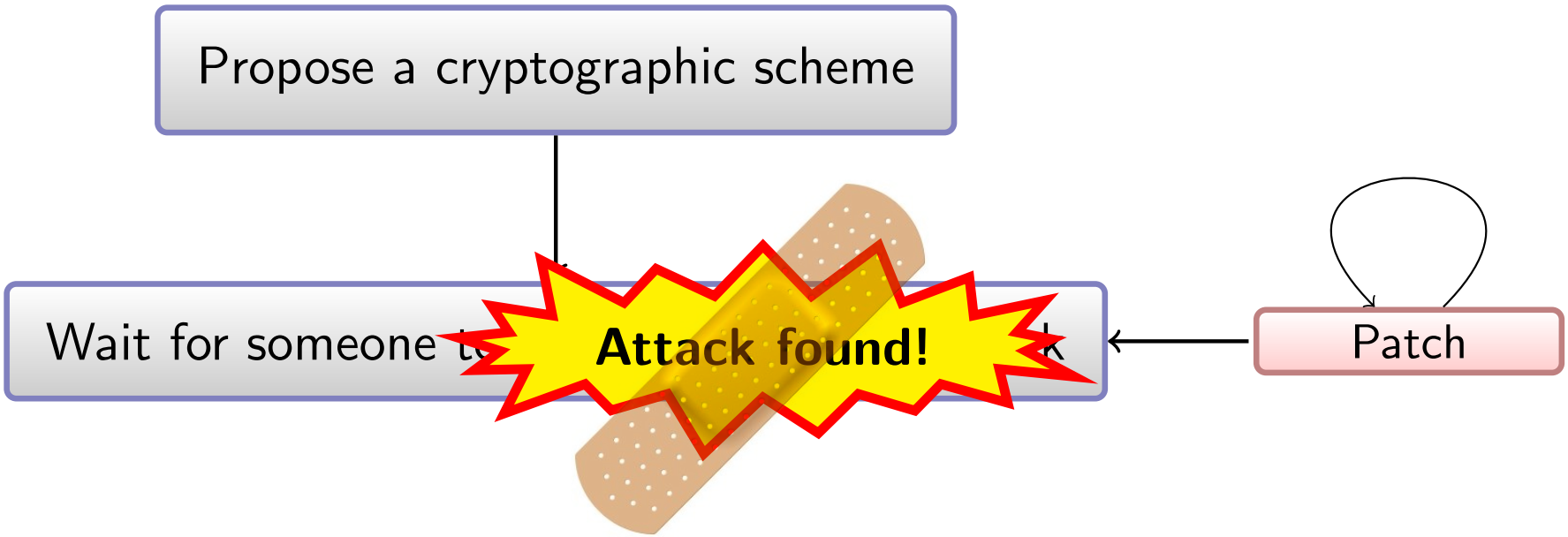
The art: cryptanalysis-driven design



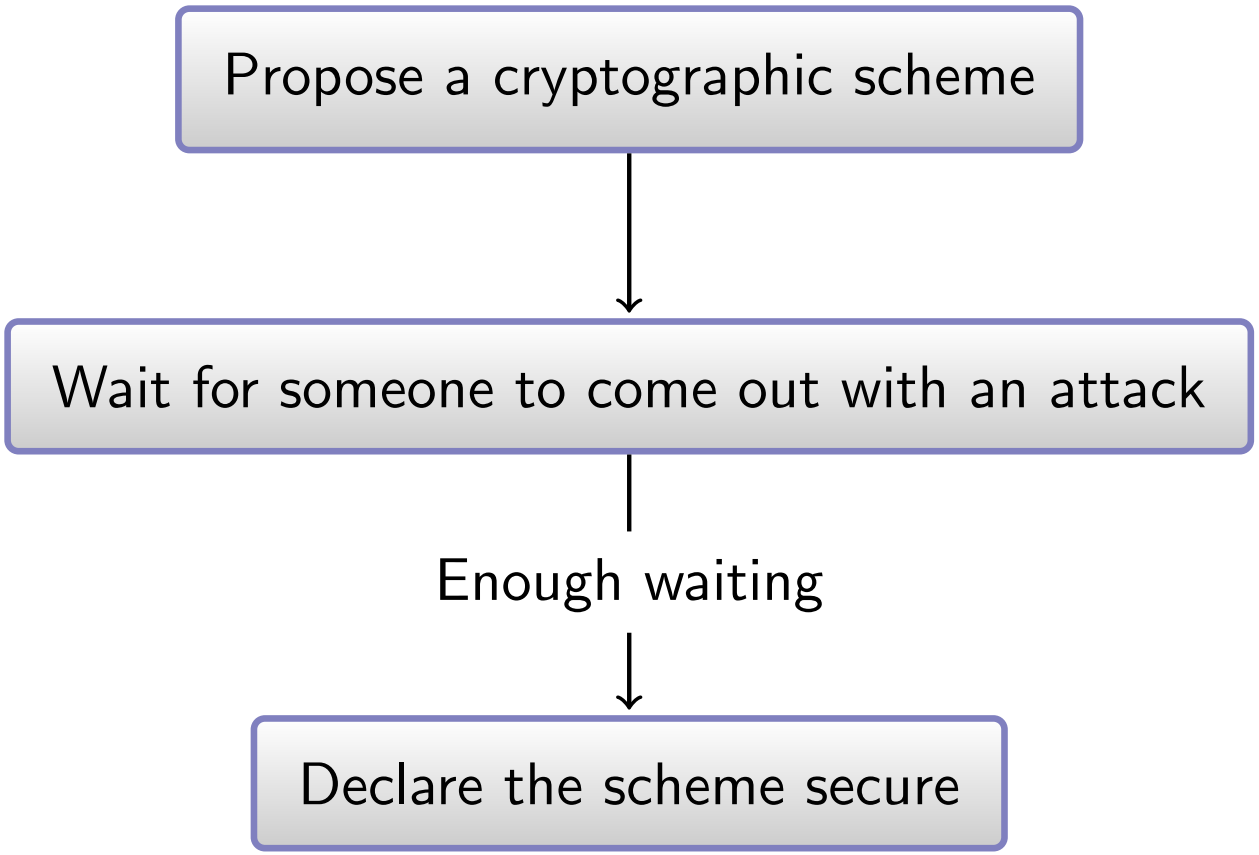
The art: cryptanalysis-driven design



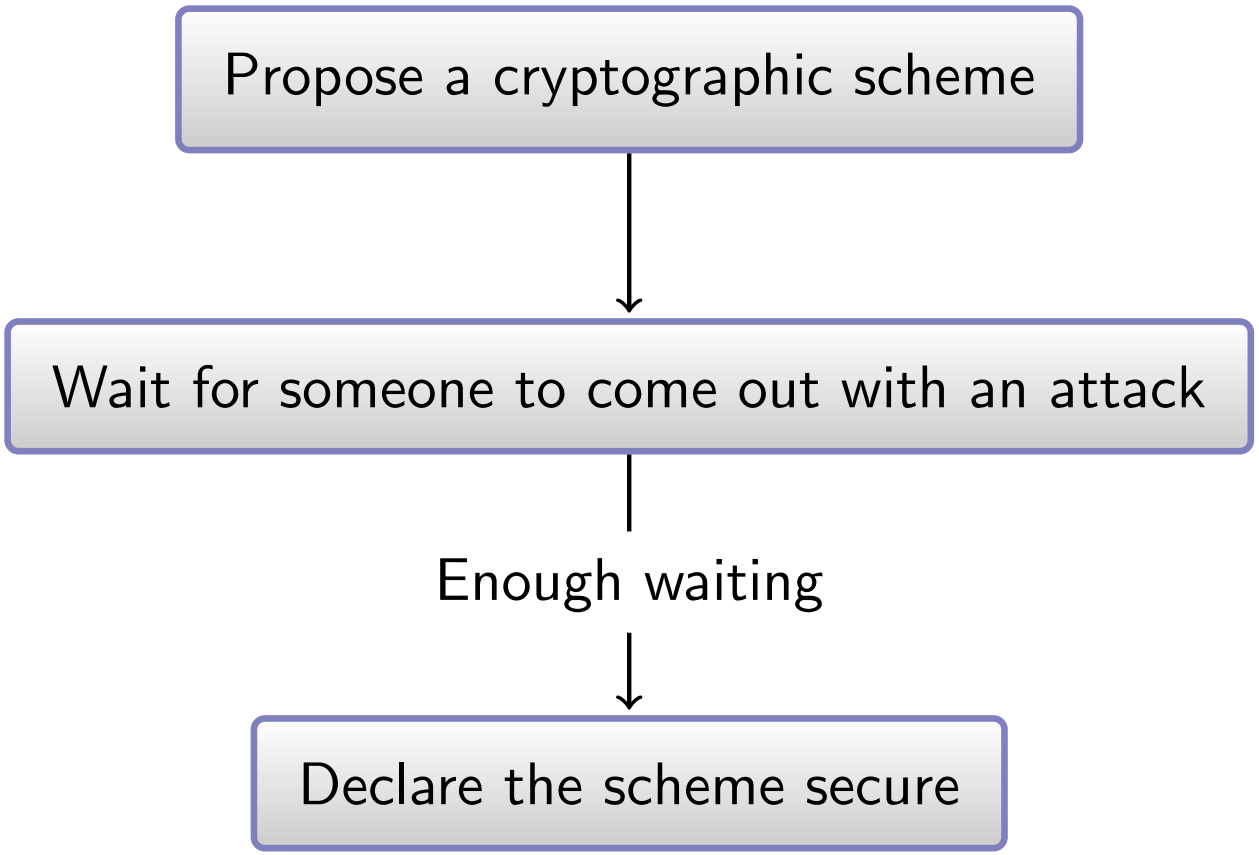
The art: cryptanalysis-driven design



The art: cryptanalysis-driven design

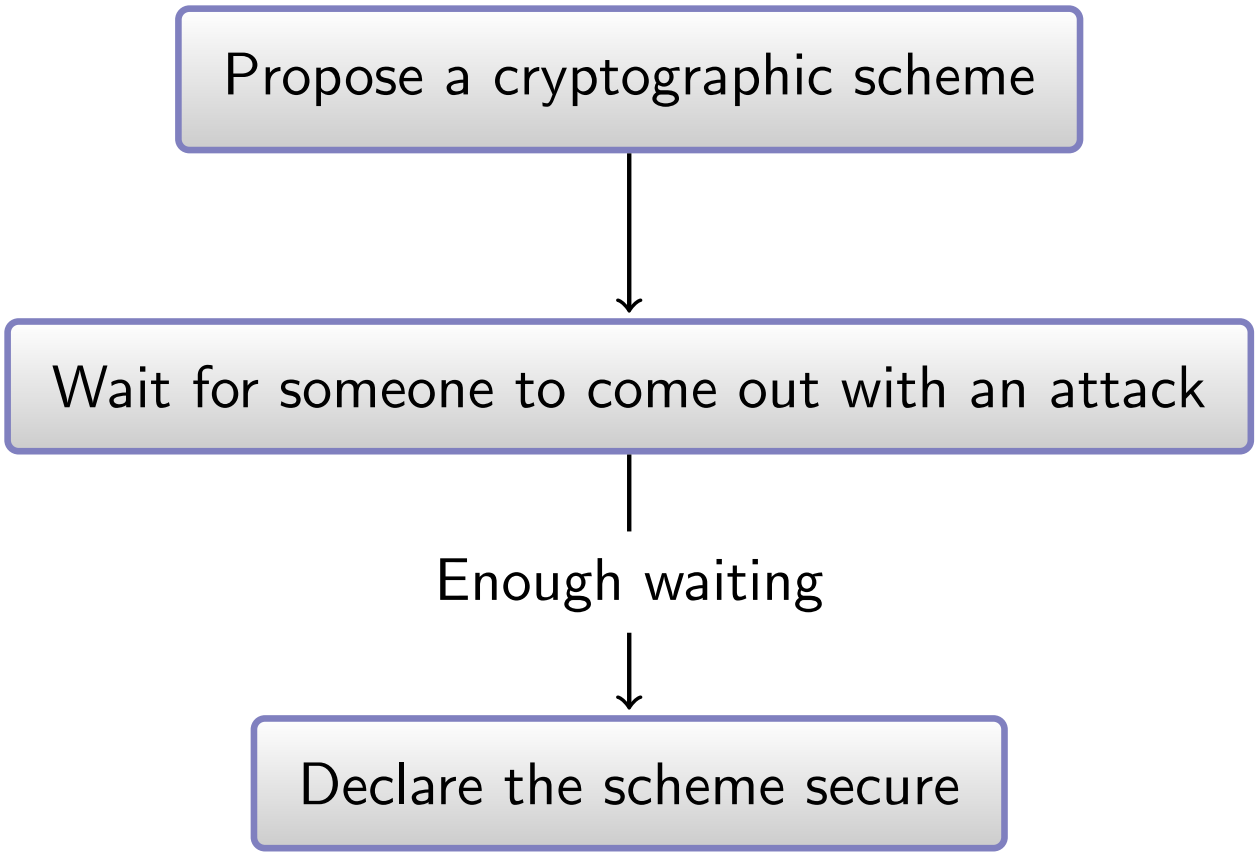


The art: cryptanalysis-driven design



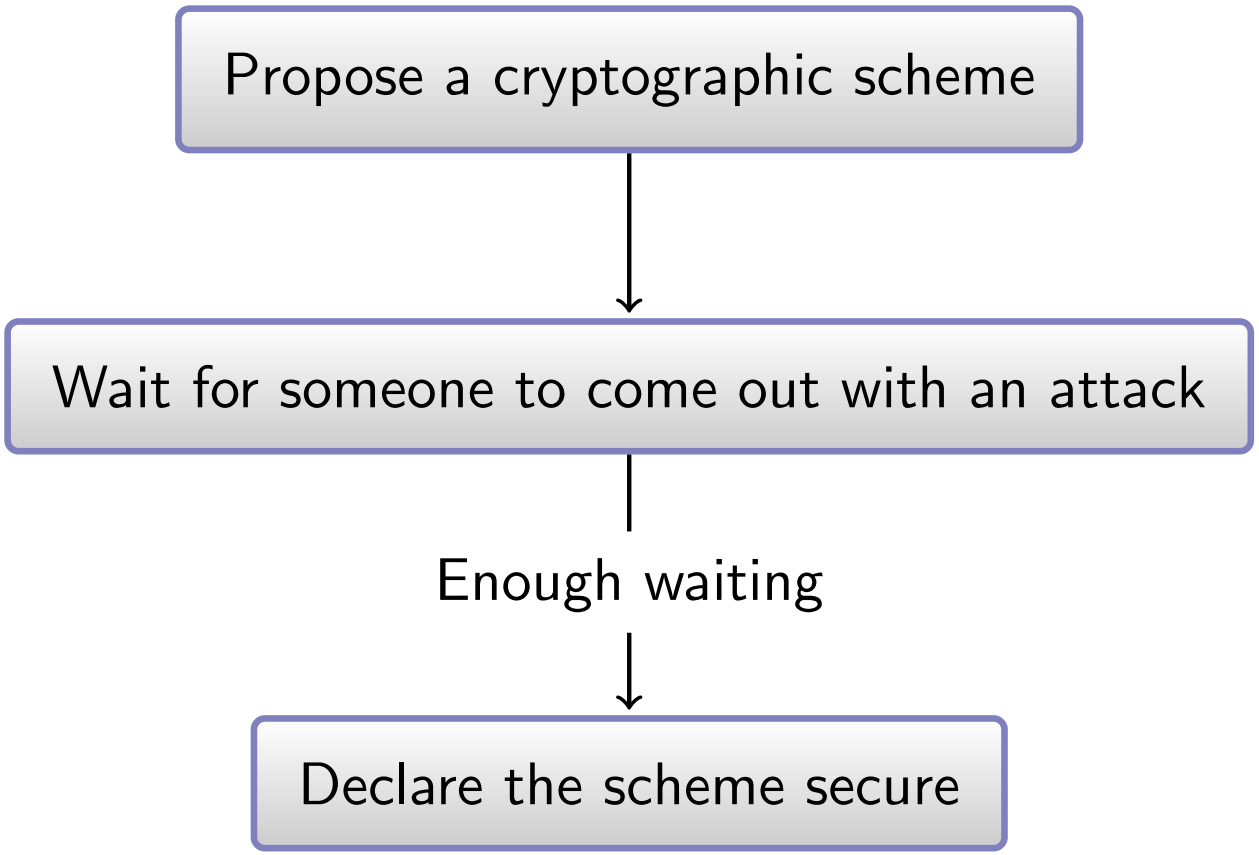
How much time is enough?

The art: cryptanalysis-driven design



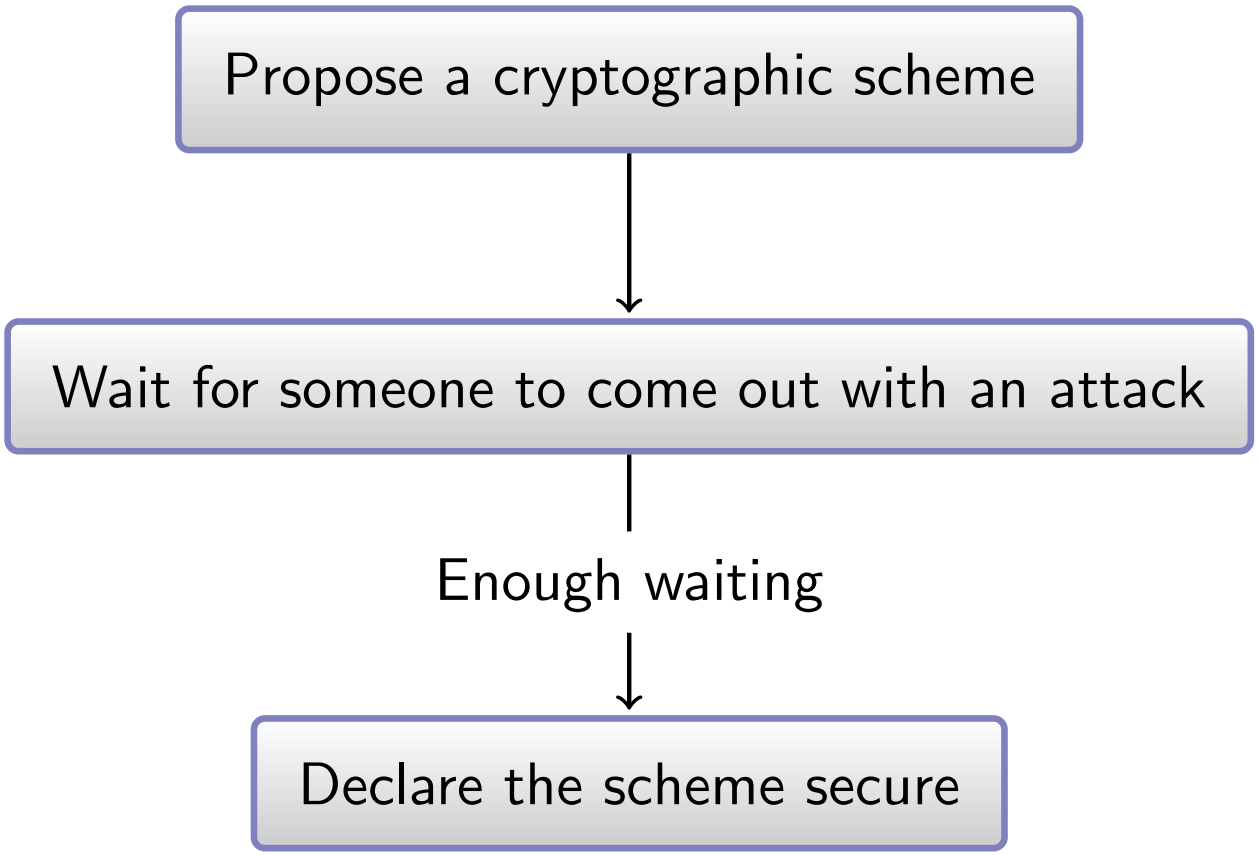
It took 5 years to break the Merkle-Hellman cryptosystem

The art: cryptanalysis-driven design



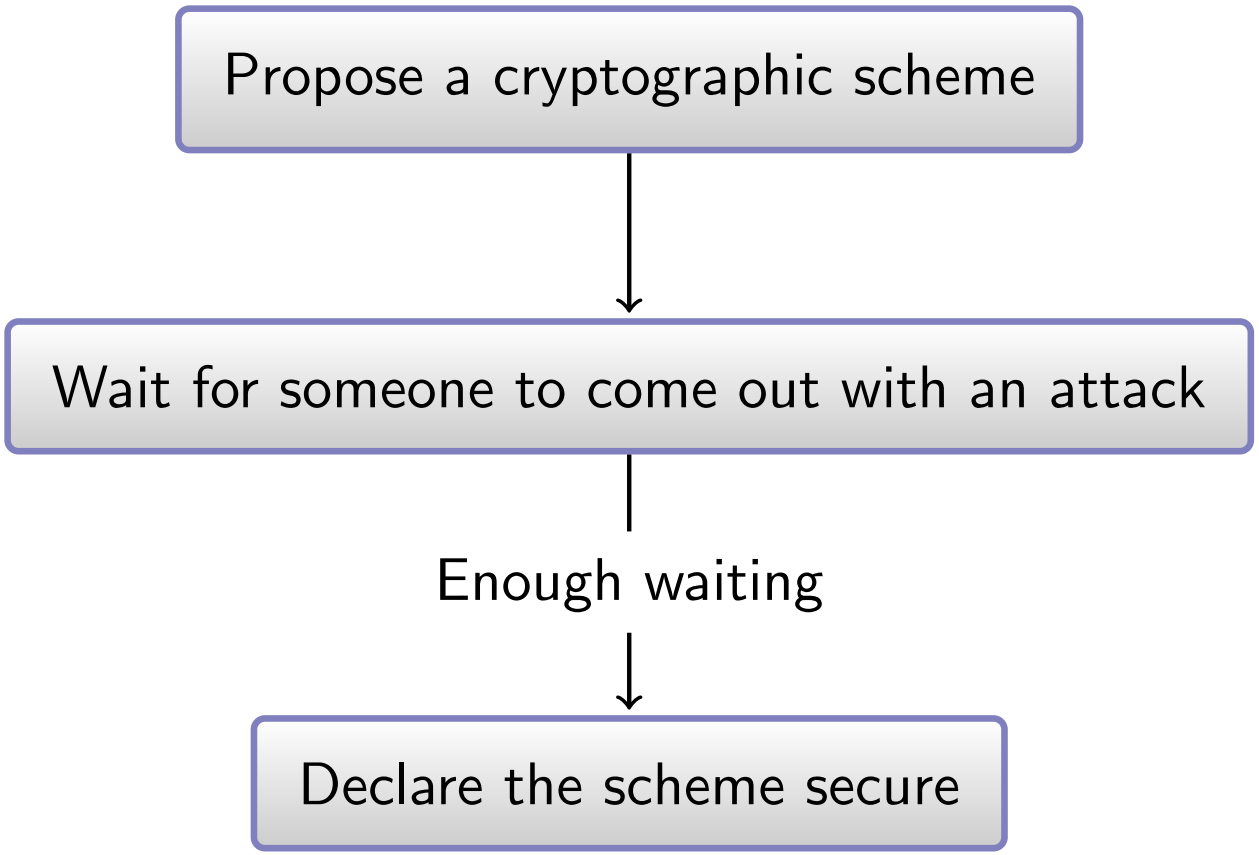
Ok, let's say 7 years

The art: cryptanalysis-driven design



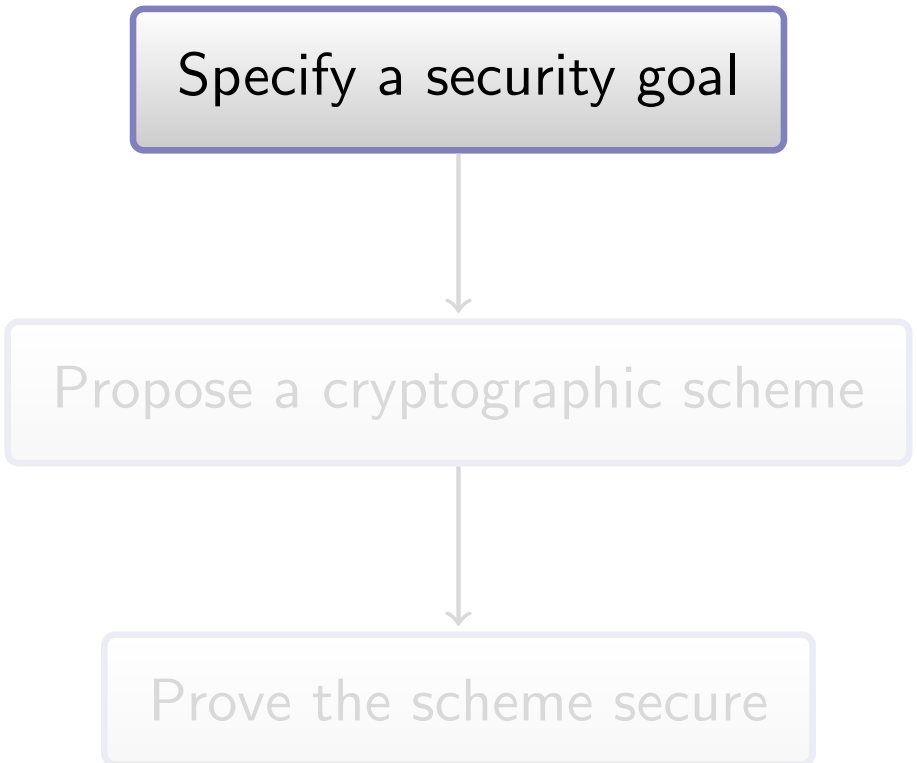
It took 10 years to break the Chor-Rivest cryptosystem

The art: cryptanalysis-driven design

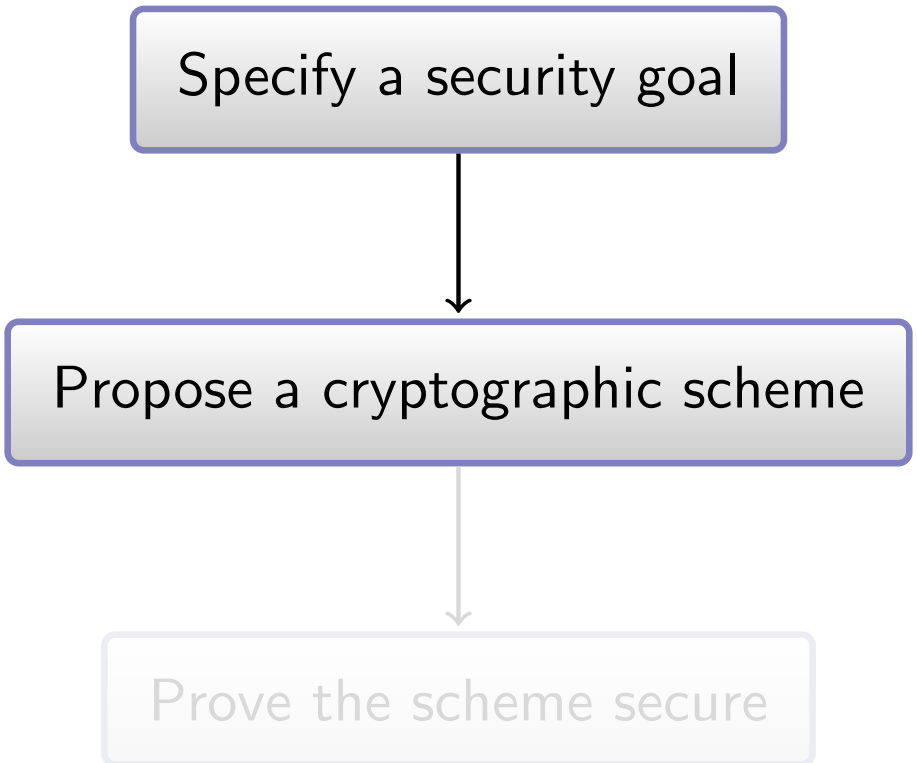


Can't we do better?

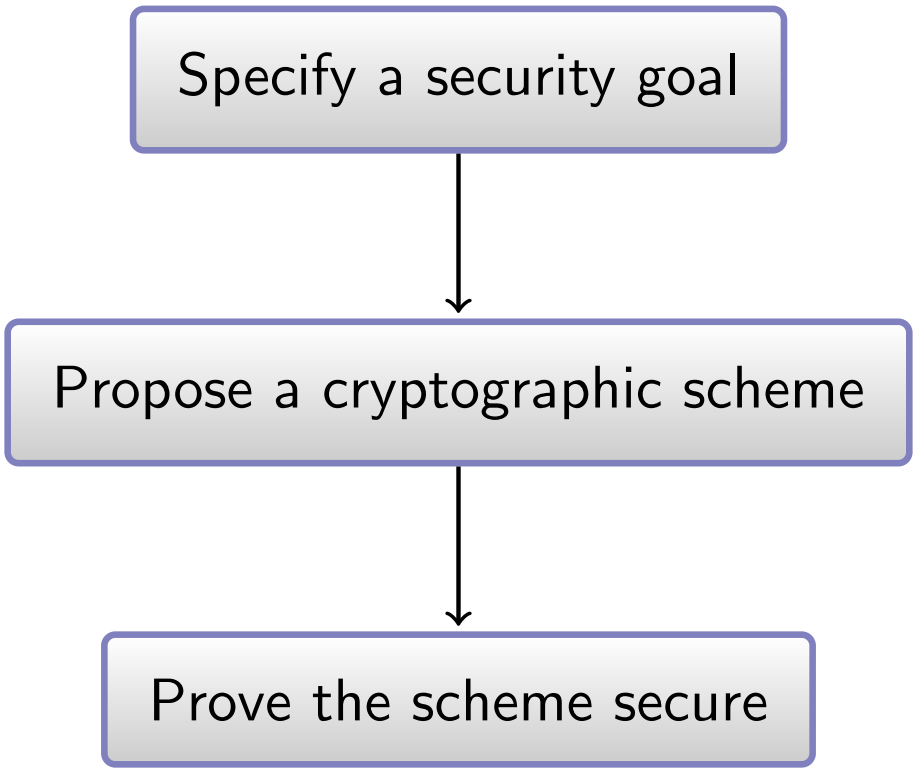
The science: provable security



The science: provable security



The science: provable security



But how?

The Provable Security paradigm

1. Define a security goal and a model for adversaries
2. Design a cryptographic scheme
3. Reduce security of the scheme to a computational assumption

IF an adversary $\mathcal{A}$ can break the security of the scheme
THEN the assumption can be broken with little extra effort

Conversely,

IF the security assumption holds THEN the scheme is secure

Proof by reduction

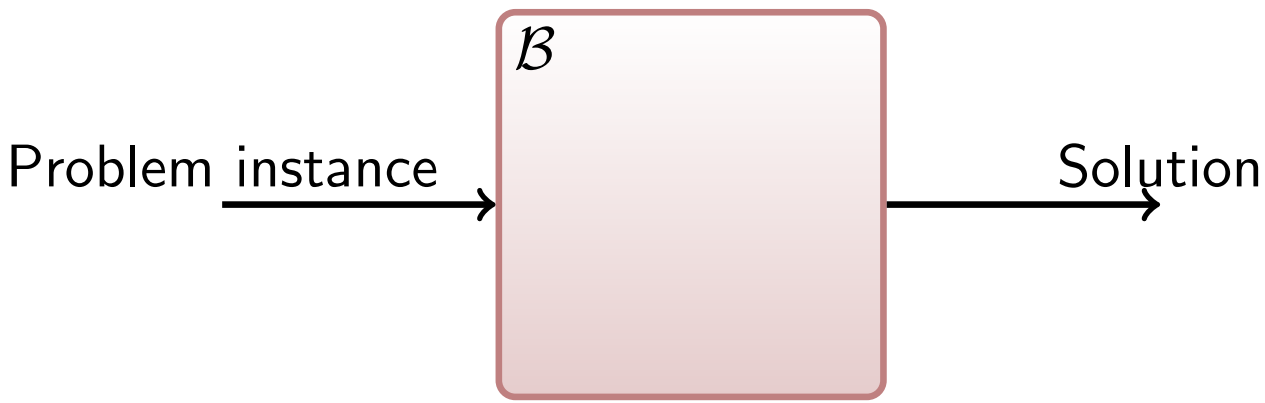
- Assume a polynomial adversary $\mathcal{A}$ breaks the security of a scheme
- Build a polynomial algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve a computational hard problem

IF the problem is intractable
THEN the cryptographic scheme is asymptotically secure

Proof by reduction

- Assume a polynomial adversary $\mathcal{A}$ breaks the security of a scheme
- Build a polynomial algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve a computational hard problem

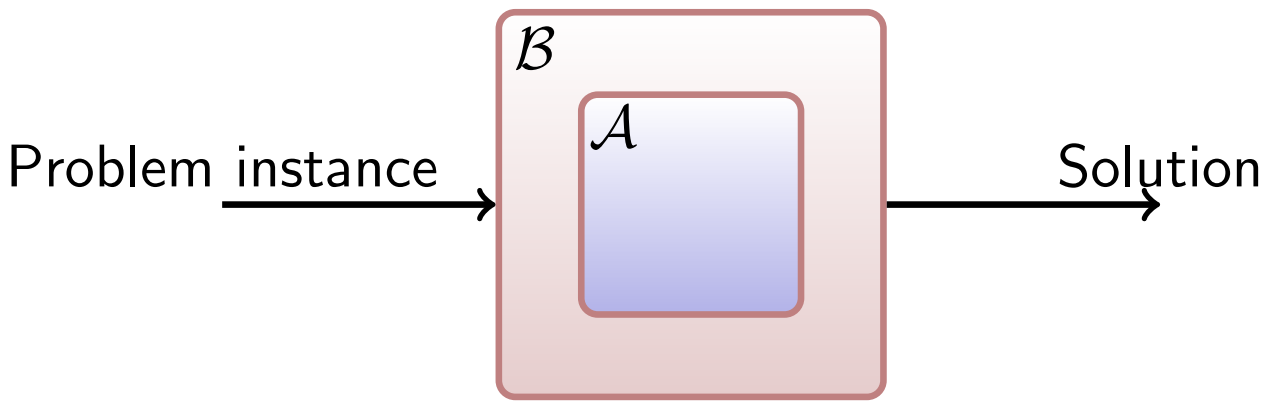
IF the problem is intractable
THEN the cryptographic scheme is asymptotically secure



Proof by reduction

- Assume a polynomial adversary $\mathcal{A}$ breaks the security of a scheme
- Build a polynomial algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve a computational hard problem

IF the problem is intractable
THEN the cryptographic scheme is asymptotically secure



Exact security

- Assume an adversary $\mathcal{A}$ breaks the security of a scheme within time t with probability ϵ
- Build an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve a computational hard problem with probability $\epsilon' \geq f(\epsilon)$ within time $t' \leq g(t)$

Bounds matter: the greater $f(\epsilon)$ and the smaller $g(t)$ are, the closer the security of the scheme is related to the problem.

Choosing scheme parameters

- What is the best known method to solve the problem?
- Choose parameters so that the reduction yields a better one

Exact security

- Assume an adversary $\mathcal{A}$ breaks the security of a scheme within time t with probability ϵ
- Build an algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve a computational hard problem with probability $\epsilon' \geq f(\epsilon)$ within time $t' \leq g(t)$

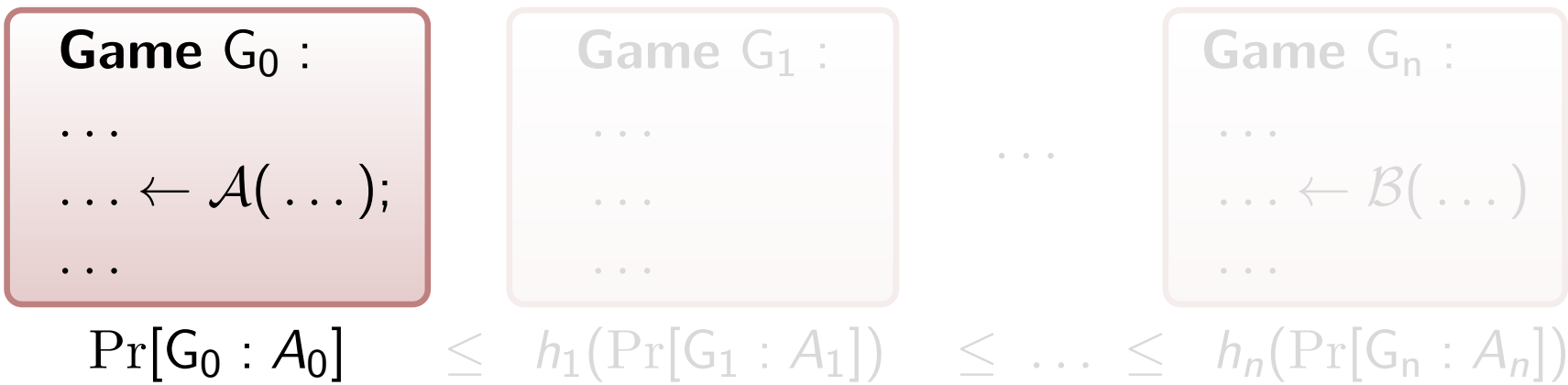
Bounds matter: the greater $f(\epsilon)$ and the smaller $g(t)$ are, the closer the security of the scheme is related to the problem.

Choosing scheme parameters

- What is the best known method to solve the problem?
- Choose parameters so that the reduction yields a better one

Game-based proofs

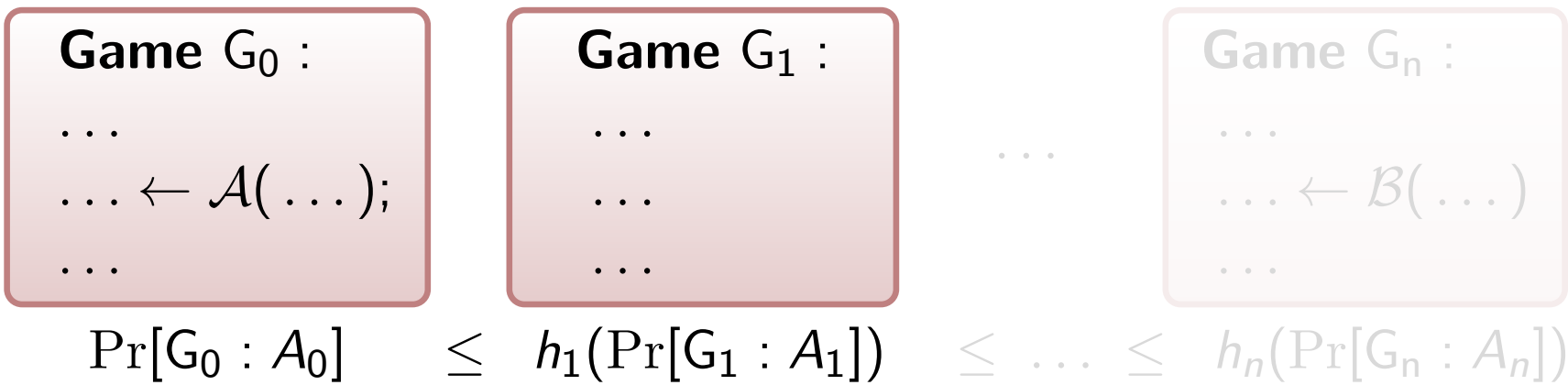
Security proofs in cryptography may be organized as sequences of games [...] this can be a useful tool in taming the complexity of security proofs that might otherwise become so messy, complicated, and subtle as to be nearly impossible to verify
V. Shoup



Start from an initial game encoding the security goal

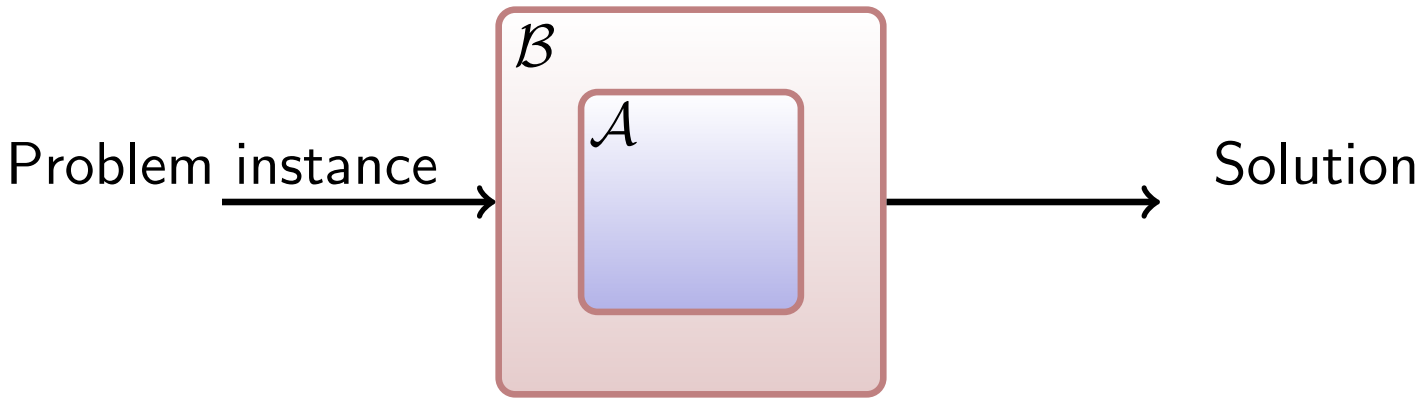
Game-based proofs

Security proofs in cryptography may be organized as sequences of games [...] this can be a useful tool in taming the complexity of security proofs that might otherwise become so messy, complicated, and subtle as to be nearly impossible to verify
V. Shoup



Stepwise transform the game keeping track of probabilities

Game-based proofs



Game G_0 :
$\ldots$
$\ldots \leftarrow \mathcal{A}(\ldots);$
$\ldots$

Game G_1 :
$\ldots$
$\ldots$
$\ldots$

$\ldots$

Game G_n :
$\ldots$
$\ldots \leftarrow \mathcal{B}(\ldots)$
$\ldots$

$$\Pr[\mathsf{G}_0 : A_0] \leq h_1(\Pr[\mathsf{G}_1 : A_1]) \leq \ldots \leq h_n(\Pr[\mathsf{G}_n : A_n])$$

Reach a final game encoding a computational assumption

Example: Public-Key Encryption

A public-key encryption scheme is a triple of algorithms $(\mathcal{KG}, \mathcal{E}, \mathcal{D})$:

Key Generation Given a security parameter $\eta : \mathbb{N}$, $\mathcal{KG}(\eta)$ generates a public key pk, used for encryption, and a secret key sk, used for decryption

Encryption Given a public key pk and a message m, $\mathcal{E}(pk, m)$ returns a ciphertext c

Decryption Given a secret key sk and a ciphertext c, $\mathcal{D}(sk, c)$ returns the decryption of c under sk, if c is a valid ciphertext or $\perp$ otherwise

$\mathcal{KG}$ and $\mathcal{E}$ are probabilistic, $\mathcal{D}$ is deterministic.

Correctness

$$\forall \eta, pk, sk, m.\ \mathcal{KG}(\eta) = (pk, sk) \implies \mathcal{D}(sk, \mathcal{E}(pk, m)) = m$$

Example: ElGamal encryption

- $\langle g \rangle$ is a cyclic group of order q
- q is exponential on the security parameter η

$$
\begin{array}{lcl}
\mathcal{KG}(\eta) & \stackrel{\text{def}}{=} & x \xleftarrow{\$} \mathbb{Z}_q;\ \text{return } (x, g^x) \\
\mathcal{E}(\alpha, m) & \stackrel{\text{def}}{=} & y \xleftarrow{\$} \mathbb{Z}_q;\ \text{return } (g^y, \alpha^y \times m) \\
\mathcal{D}(x, (\beta, \zeta)) & \stackrel{\text{def}}{=} & \text{return } \zeta \times \beta^{-x}
\end{array}
$$

Correctness:

Example: ElGamal encryption

- $\langle g \rangle$ is a cyclic group of order q
- q is exponential on the security parameter η

$$
\begin{array}{lcl}
\mathcal{KG}(\eta) & \stackrel{\text{def}}{=} & x \xleftarrow{\$} \mathbb{Z}_q;\ \text{return } (x, g^x) \\
\mathcal{E}(\alpha, m) & \stackrel{\text{def}}{=} & y \xleftarrow{\$} \mathbb{Z}_q;\ \text{return } (g^y, \alpha^y \times m) \\
\mathcal{D}(x, (\beta, \zeta)) & \stackrel{\text{def}}{=} & \text{return } \zeta \times \beta^{-x}
\end{array}
$$

Correctness:

$$\mathcal{D}(x, \mathcal{E}(g^x, m))$$

Example: ElGamal encryption

- $\langle g \rangle$ is a cyclic group of order q
- q is exponential on the security parameter η

$$
\begin{array}{lcl}
\mathcal{KG}(\eta) & \stackrel{\text{def}}{=} & x \xleftarrow{\$} \mathbb{Z}_q;\ \text{return } (x, g^x) \\
\mathcal{E}(\alpha, m) & \stackrel{\text{def}}{=} & y \xleftarrow{\$} \mathbb{Z}_q;\ \text{return } (g^y, \alpha^y \times m) \\
\mathcal{D}(x, (\beta, \zeta)) & \stackrel{\text{def}}{=} & \text{return } \zeta \times \beta^{-x}
\end{array}
$$

Correctness:

$$\mathcal{D}(x, (g^y, g^{xy} \times m))$$

Example: ElGamal encryption

- $\langle g \rangle$ is a cyclic group of order q
- q is exponential on the security parameter η

$$
\begin{array}{lcl}
\mathcal{KG}(\eta) & \stackrel{\text{def}}{=} & x \xleftarrow{\$} \mathbb{Z}_q;\ \text{return } (x, g^x) \\
\mathcal{E}(\alpha, m) & \stackrel{\text{def}}{=} & y \xleftarrow{\$} \mathbb{Z}_q;\ \text{return } (g^y, \alpha^y \times m) \\
\mathcal{D}(x, (\beta, \zeta)) & \stackrel{\text{def}}{=} & \text{return } \zeta \times \beta^{-x}
\end{array}
$$

Correctness:

$$(g^{xy} \times m) \times (g^y)^{-x}$$

Example: ElGamal encryption

- $\langle g \rangle$ is a cyclic group of order q
- q is exponential on the security parameter η

$$
\begin{array}{lcl}
\mathcal{KG}(\eta) & \stackrel{\text{def}}{=} & x \xleftarrow{\$} \mathbb{Z}_q;\ \text{return } (x, g^x) \\
\mathcal{E}(\alpha, m) & \stackrel{\text{def}}{=} & y \xleftarrow{\$} \mathbb{Z}_q;\ \text{return } (g^y, \alpha^y \times m) \\
\mathcal{D}(x, (\beta, \zeta)) & \stackrel{\text{def}}{=} & \text{return } \zeta \times \beta^{-x}
\end{array}
$$

Correctness:

$$(g^{xy} \times m) \times g^{-xy}$$

Example: ElGamal encryption

- $\langle g \rangle$ is a cyclic group of order q
- q is exponential on the security parameter η

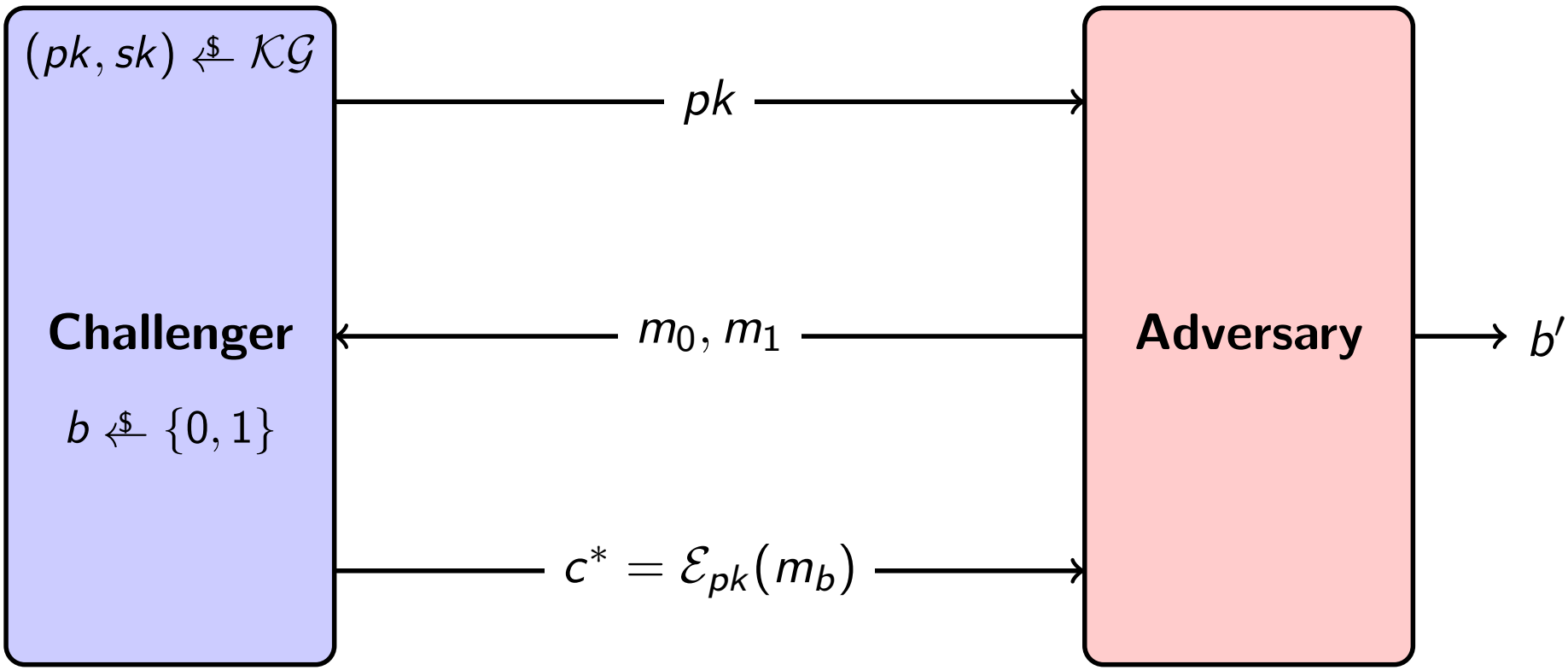
$$
\begin{array}{lcl}
\mathcal{KG}(\eta) & \stackrel{\text{def}}{=} & x \xleftarrow{\$} \mathbb{Z}_q;\ \text{return } (x, g^x) \\
\mathcal{E}(\alpha, m) & \stackrel{\text{def}}{=} & y \xleftarrow{\$} \mathbb{Z}_q;\ \text{return } (g^y, \alpha^y \times m) \\
\mathcal{D}(x, (\beta, \zeta)) & \stackrel{\text{def}}{=} & \text{return } \zeta \times \beta^{-x}
\end{array}
$$

Correctness:

$$m$$

Chosen-plaintext security

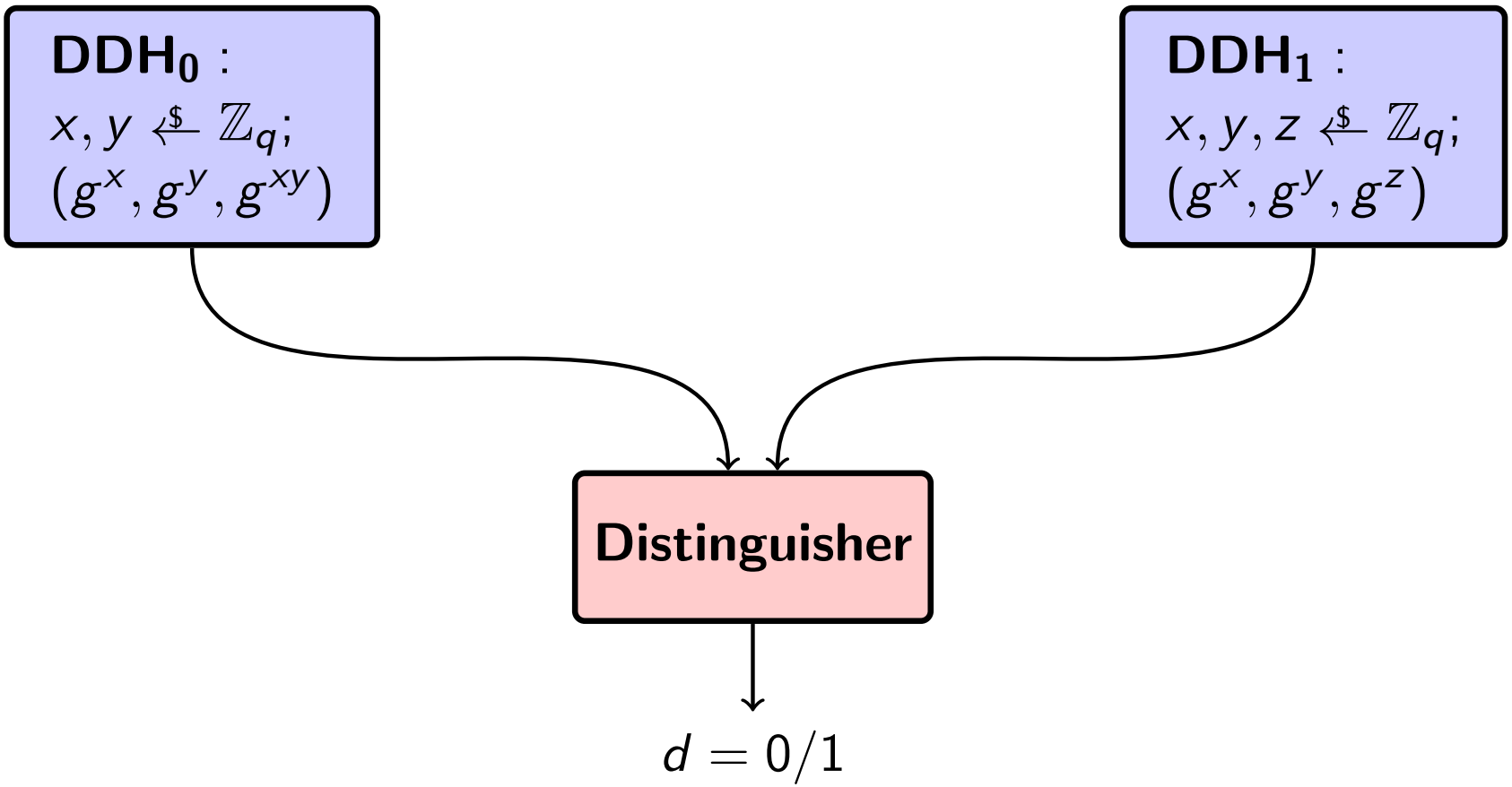
Define the experiment CPA:



Advantage of $\mathcal{A}$: $\mathbf{Adv}^{\mathcal{A}}_{\mathrm{CPA}} \stackrel{\mathrm{def}}{=} 2\ \Pr[\mathrm{CPA} : b = b'] - 1$

The Decisional Diffie-Hellman Assumption

Define the two experiments DDH_0 and DDH_1:



$$\mathbf{Adv}^{\mathcal{B}}_{\mathrm{DDH}} \stackrel{\mathrm{def}}{=} |\Pr[\mathrm{DDH}_0 : d = 1] - \Pr[\mathrm{DDH}_1 : d = 1]|$$

CPA-security of ElGamal

ElGamal is secure under the Decisional Diffie-Hellman assumption

DDH Assumption

For every polynomial-time distinguisher $\mathcal{B}$, $\mathbf{Adv}^{\mathcal{B}}_{\mathrm{DDH}}$ is negligible

One can prove:

$$\forall \mathcal{A}.\ \mathrm{PPT}(A) \implies \exists \mathcal{B}.\ \mathrm{PPT}(B) \wedge \mathbf{Adv}^{\mathcal{A}}_{\mathrm{CPA}} = \mathbf{Adv}^{\mathcal{B}}_{\mathrm{DDH}}$$

which implies, under the DDH assumption, that for any PPT adversary $\mathcal{A}$, $\mathbf{Adv}^{\mathcal{A}}_{CPA}$ is negligible

CPA-security of ElGamal

ElGamal is secure under the Decisional Diffie-Hellman assumption

DDH Assumption

For every polynomial-time distinguisher $\mathcal{B}$, $\mathbf{Adv}^{\mathcal{B}}_{\mathrm{DDH}}$ is negligible

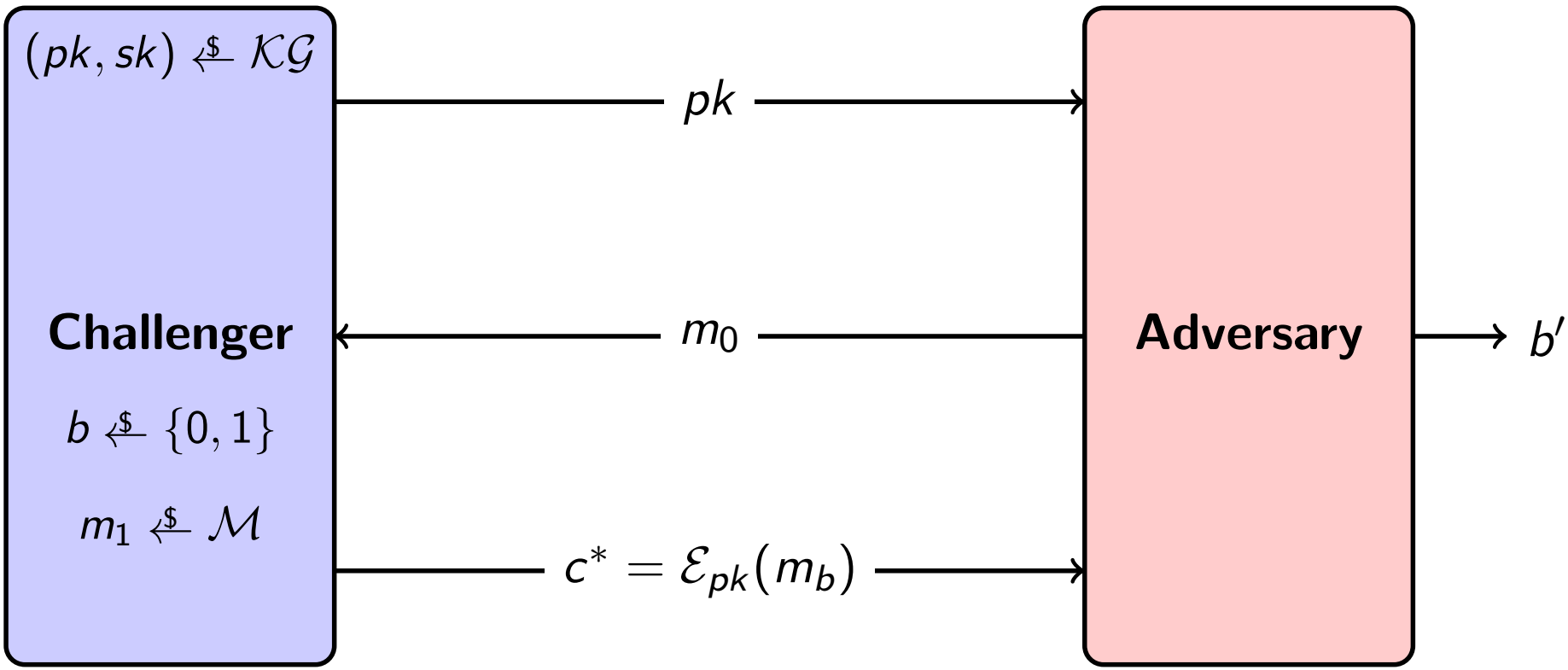
One can prove:

$$\forall \mathcal{A}.\ \mathrm{PPT}(A) \implies \exists \mathcal{B}.\ \mathrm{PPT}(B) \wedge \mathbf{Adv}^{\mathcal{A}}_{\mathrm{CPA}} = \mathbf{Adv}^{\mathcal{B}}_{\mathrm{DDH}}$$

which implies, under the DDH assumption, that for any PPT adversary $\mathcal{A}$, $\mathbf{Adv}^{\mathcal{A}}_{CPA}$ is negligible

Real-or-Random security

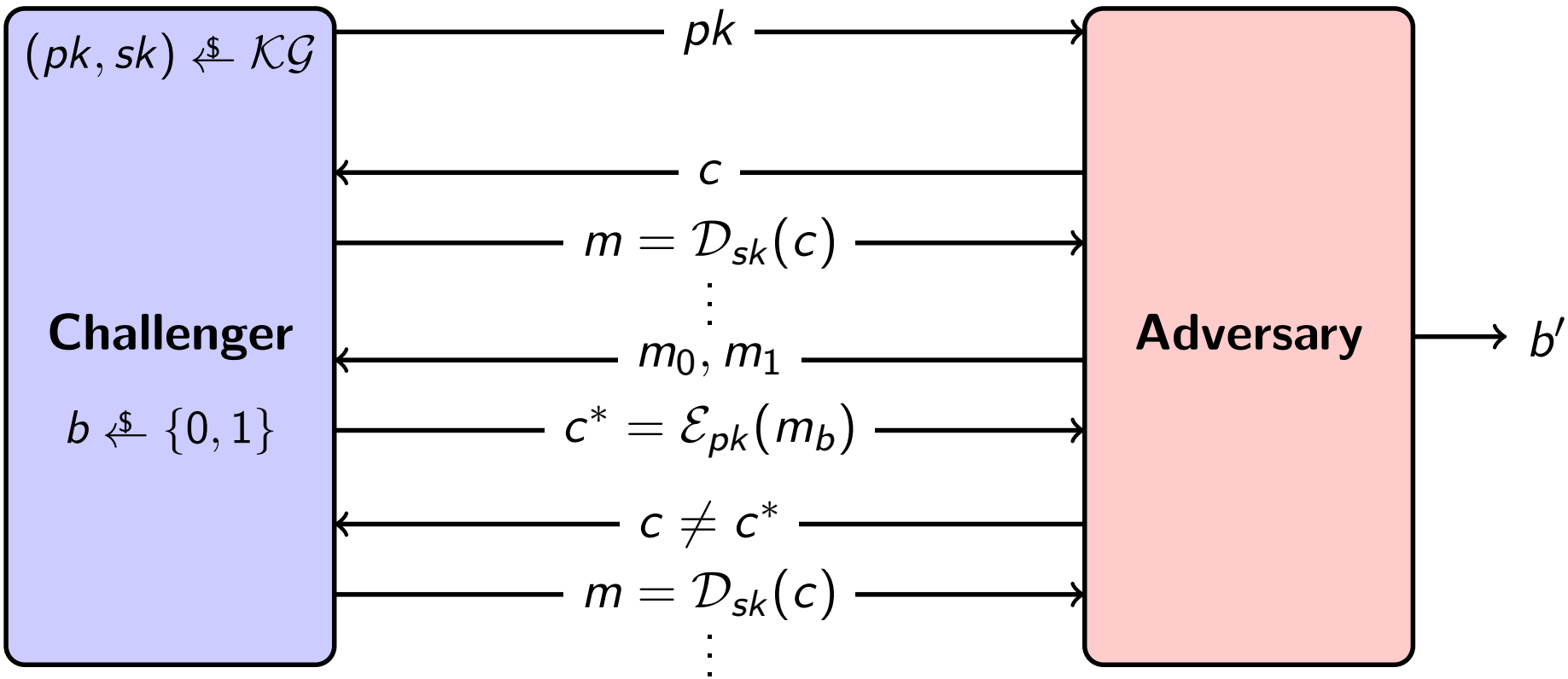
Define the experiment ROR:



Advantage of $\mathcal{A}$: $\mathbf{Adv}^{\mathcal{A}}_{\mathrm{ROR}} \stackrel{\mathrm{def}}{=} 2\ \Pr[\mathrm{ROR} : b = b'] - 1$

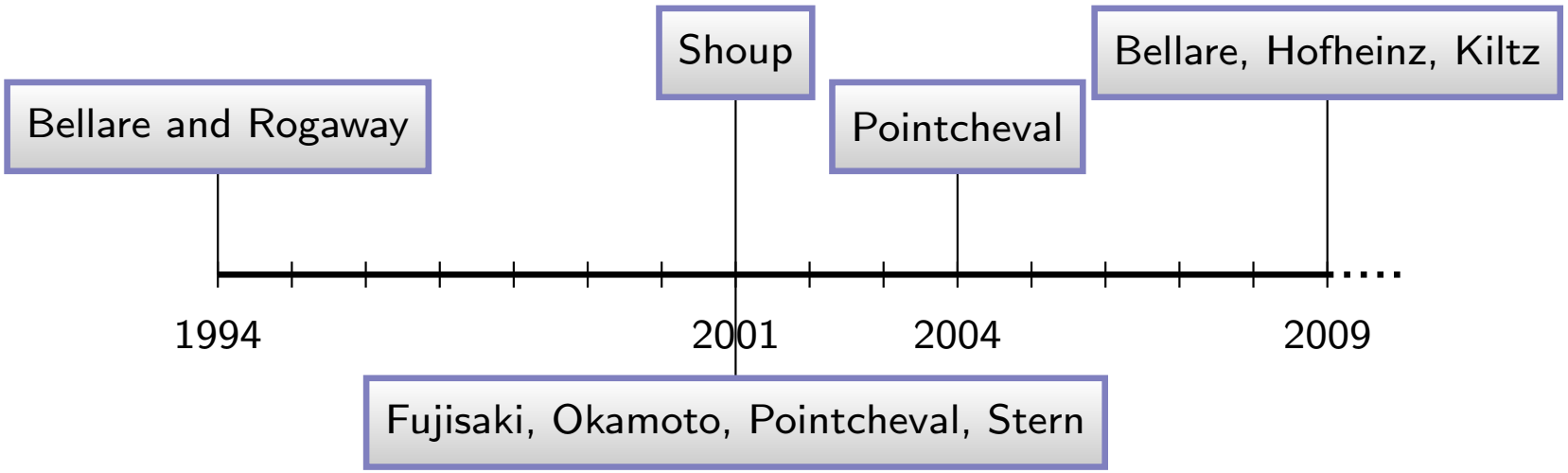
Chosen-ciphertext security

Define the experiment CCA:



Advantage of $\mathcal{A}$: $\mathbf{Adv}^{\mathcal{A}}_{\mathrm{CCA}} \stackrel{\mathrm{def}}{=} 2\ \Pr[\mathrm{CCA} : b = b'] - 1$

Things can still go wrong: RSA-OAEP



1994 Purported proof of chosen-ciphertext security

2001 Proof is flawed, but can be patched

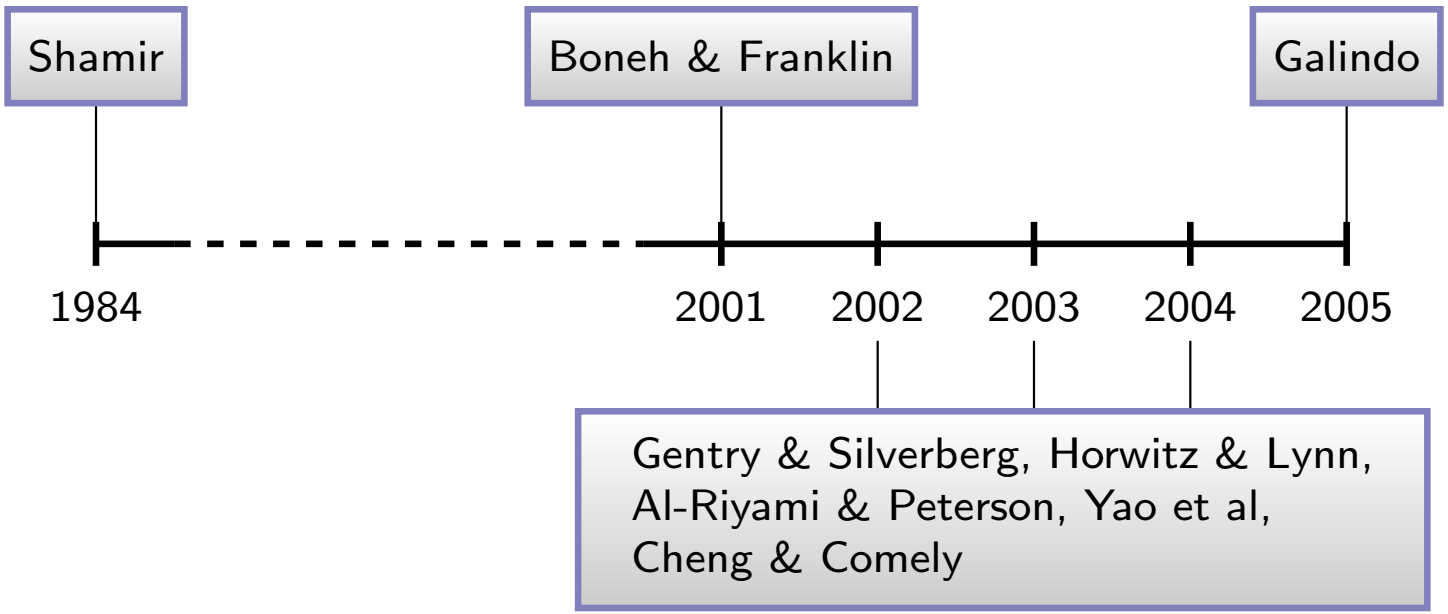
- ...for a weaker security notion, or
- ...for a modified scheme, or
- ...under stronger assumptions

2004 Filled gaps in Fujisaki et al. 2001 proof

2009 Security definition needs to be clarified

2011 Filled gaps and marginally improved bound in 2004 proof

Things can still go wrong: Identity-Based Encryption



1984: Conception of Identity-Based Cryptography

2001: First practical provably-secure IBE scheme.

2002-2005: Extensively used as a buliding block

2005: Proof is flawed, but can be patched

- ...for a weaker security bound

Things can still go wrong: cryptographers know it

- *In our opinion, many proofs in cryptography have become essentially unverifiable. Our field may be approaching a crisis of rigor*
  M. Bellare and P. Rogaway (2004)

- *Do we have a problem with cryptographic proofs? Yes, we do [...] We generate more proofs than we carefully verify (and as a consequence some of our published proofs are incorrect)*
  S. Halevi (2005)

Next class: code-based cryptographic proofs

Goal

Build a framework to formalize game-based cryptographic proofs

- Provide foundations to game-based proofs
- Notation as close as possible to cryptographer's
- Automate common reasoning patterns
- Support exact security
- Provide independently and automatically verifiable proofs

Excercises

1. Roll six traditional fair dice. What is the probability of getting exactly 4 different numbers?
2. Design a set of four 6-sided dice d_1, d_2, d_3, d_4 such that d_i rolls higher than d_{i+1} with probability more than $1/2$ for $i = 1, 2, 3$. What is the probability p that d_4 rolls higher than d_1? Can you design a set where $p < 1/2$?
3. Let x, y, z be independent uniform random variables over the set of bitstrings of length k. What is the distribution of $x \oplus y$? Now suppose y is constant, what is the distribution of $x \oplus y$?
4. Show by reduction the equivalence between ROR and CPA security for public-key encryption.