Information Security Lab Module 2: Cryptographic Reductions

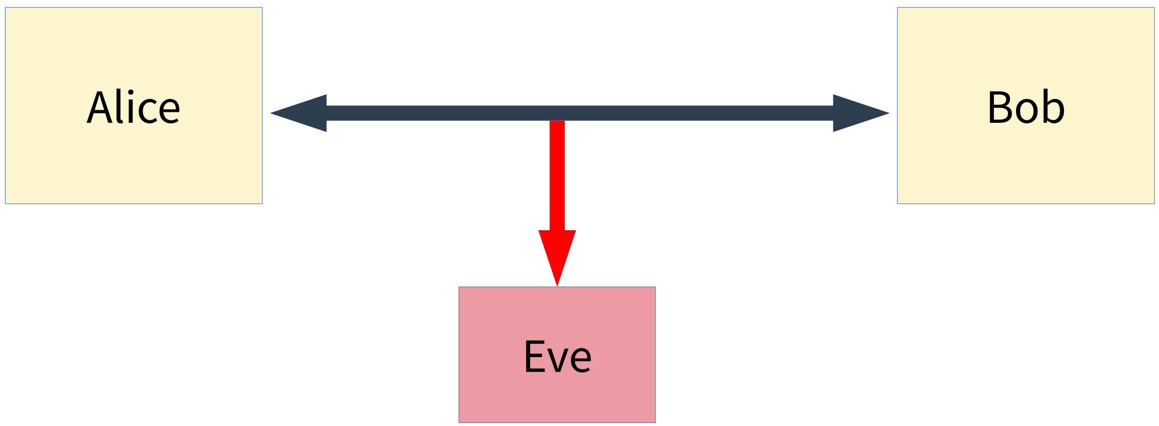
Dennis Hofheinz

Topic of this module

- **Cryptographic building blocks**
  - Examples: encryption/signature schemes, zero-knowledge proofs, …
  - Often not the solution, but a crucial part of the solution (e.g., TLS)
- **Question 1: how to construct them/reason about them?**
- **Question 2: how to use them in a larger system?**
- **Crucial: good interface between the two**
  - Achievable/useful security definition (e.g., for encryption)
- **This module: Question 1**
  - Goal 1.1: achievable (and universal/useful) security definition
  - Goal 1.2: construct a secure cryptographic scheme **with proof**
- **What about Question 2? (→ remarks/outlook)**

Motivation for public-key encryption

- **Running example building block: public-key encryption**
  - Surprising, relevant, well-studied, instructive
- **Scenario: Alice and Bob want to communicate over insecure channel**



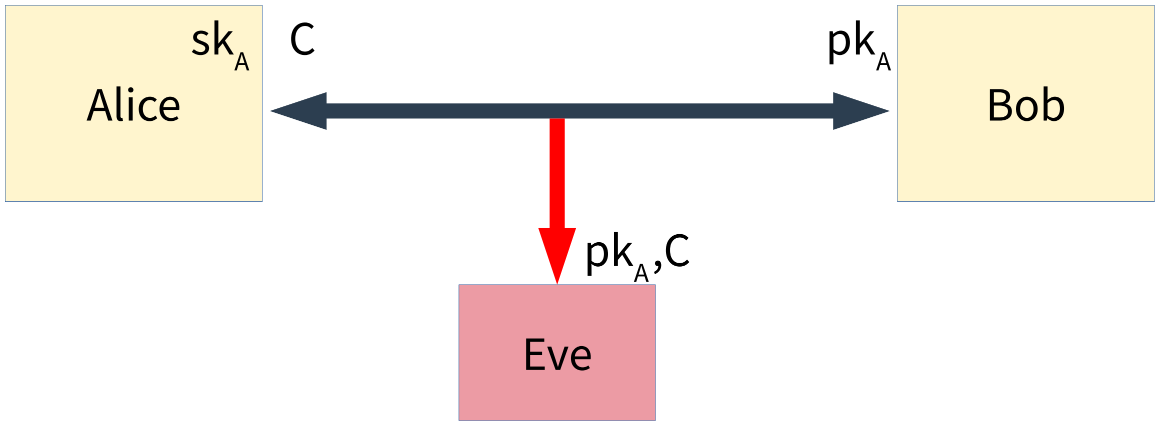
  - Possible solution: perform a key exchange, use symmetric encryption
  - Simplified setting: channel authenticated (i.e., Eve can only eavesdrop)

Motivation for public-key encryption

- **Public-key encryption to the rescue!**
  - A public-key encryption (PKE) scheme consists of the following algorithms:
    - Key generation: Gen() outputs a keypair (pk,sk)
      pk is called "public key", sk is called "secret key"
    - Encryption: Enc(pk,M) outputs a ciphertext C for message M
    - Decryption: Dec(sk,C) outputs the message M
  - pk allows to encrypt ("hide" the message M), sk allows to decrypt
  - Correctness: Dec(sk,Enc(pk,M)) = M always
  - Intuitively, should be hard to obtain M from C without sk
    - Not trivial how to formally capture this, we'll get into that
  - Invented 1976 by Diffie and Hellman (or 1970 by Ellis, or 1974 by Merkle)
  - First scheme 1977 by Rivest, Shamir, and Adleman (or 1973 by Cocks, or 1974 by Merkle)

Motivation for public-key encryption

- **Scenario: Alice and Bob want to communicate over insecure channel**



  - Let‘s say Alice wants to send a message M to Bob
  - Here is how public-key encryption helps (informally):
    - First step: Alice generates a keypair (pk_A,sk_A) and sends pk_A to Bob
    - Second step: Bob encrypts M and sends the ciphertext $C=Enc(pk_A,M)$ to Alice
    - Third step: Alice decrypts C to retrieve $M=Dec(sk_A,C)$

Motivation for security definitions

- **So are we there yet?**
  - Not quite, we should have a definition of security for PKE schemes
    - Provides interface between PKE builders/users
    - Could come in different flavors (e.g., against active/passive adversaries)
    - Allows to argue benefits/shortcomings of different schemes
- **First goal: identify "good" security definition for PKE schemes**
  - First attempt: should be hard to compute sk from pk
    - Necessary, but what if it's possible to compute sk' that also decrypts?
  - Second attempt: should be hard to compute M from C
    - Not well-defined: for what M? Fixed? Uniform? Application-dependent?
      For uniform M, this is often called "one-way security"
    - Also, what if it's possible to retrieve the first half of M?
  - Need to work harder

Semantic security

- **Goldwasser-Micali 1984: "semantic security"**
  - "Everything that can be computed efficiently with C about M…
    …should also be computable efficiently without C."
  - Intuition: C does not help in any computations involving M
  - More formally:

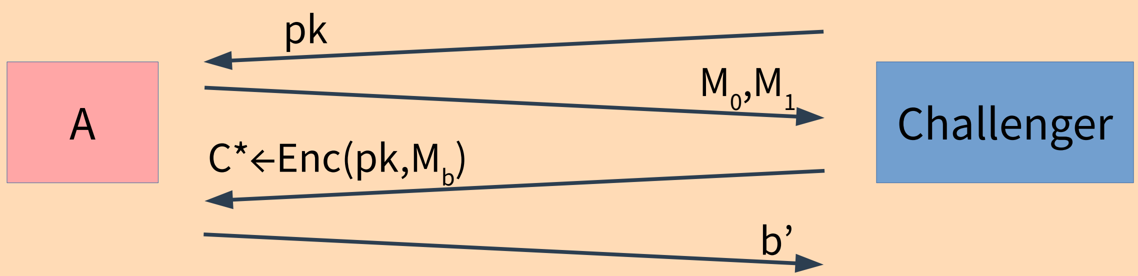
> A PKE scheme PKE=(Gen,Enc,Dec) is <u>semantically secure</u>, if for every distribution D on equal-length messages, every predicate P, and every efficient algorithm A, there exists an efficient simulator S such that
>
> Pr[A(pk,C) = P(M)] < Pr[S(pk) = P(M)] + "small"
>
> where (pk,sk)←Gen(), M←D, C←Enc(pk,M)

("efficient" and "small" are not yet formally defined)

IND-CPA security

- **Semantic security intuitive, but somewhat complex**
  - Implies secure channels against passive adversaries in arbitrary contexts
  - But: four quantifiers, need to construct simulator S for every adversary A
- **Fortunately, equivalent, but technically simpler notion exists**

A PKE scheme PKE=(Gen,Enc,Dec) is <u>indistinguishable under chosen-plaintext attacks (IND-CPA secure)</u>, if no efficient adversary A can win the following experiment with probability significantly larger than ½:



A finally wins iff $|M_0|=|M_1|$ and b=b'

  - Intuition: hard to distinguish ciphertexts of self-chosen messages

More on (IND-CPA) security

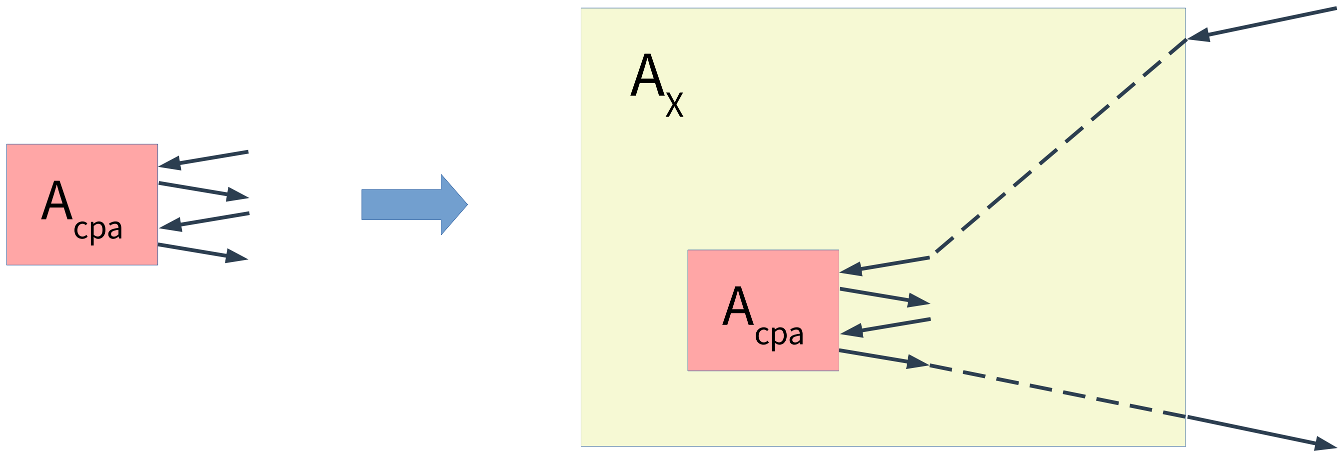
- **IND-CPA security perhaps less intuitive, but more manageable**
  - Theorem (without proof): PKE semantically secure iff PKE IND-CPA secure
  - $|M_0| = |M_1|$ requirement concession to correctness (large $M \Rightarrow$ large C)
- **Observation: IND-CPA security requires randomized encryption**
  - Assume Enc is deterministic (i.e., same $(pk,M) \Rightarrow$ same C)
  - Then adversary A can encrypt M_0, M_1 using pk and compare with C
  - Side remark: do not use textbook RSA ($C=M^e \bmod N$) unless you know exactly what you are doing!
- **Many IND-CPA secure schemes known on many different platforms (groups, factorization, lattices, coding theory, multivariate equations, …)**
- **For active attacks, stronger notions possible (more on this later)**

Towards achieving security

- **Problem: any reasonable form of PKE security requires assumptions**
  - Inefficient attacks against any PKE scheme exist (run Gen() until you get pk)
  - In fact, if P=NP, then a poly-time algorithm to get sk from pk exists
  - Hence, must base any reasonable form of PKE security on assumptions
- **Assumption should be "easier to check" than security of scheme**
- **Popular assumptions:**
  - Factoring: given N=PQ for primes P,Q, find P,Q
  - Discrete logarithm (in group G = <g>): given (g,g^x), find x
  - Computational Diffie-Hellman (in G): given (g,g^x,g^y), find g^{xy}
  - Decisional Diffie-Hellman (in G): given (g,g^x,g^y), distinguish g^{xy} from random
  - Learning with errors: given (A, Ax+e) for $A \in Z_q^{m \times n}$, small $e \in Z_q^m$, find $x \in Z_q^n$

Towards achieving security

- **Goal: results of the form "if X holds, then PKE is IND-CPA secure"**
  - PKE schemes known for $X \in \{$ factoring, CDH, DDH, LWE $\}$ (but not dlog!)
  - Technical means to achieve this: reduction
    - Given any adversary A_{cpa} on PKE's IND-CPA security…
      … construct an adversary A_X on assumption X
    - Most of the time, black-box reductions will be possible:

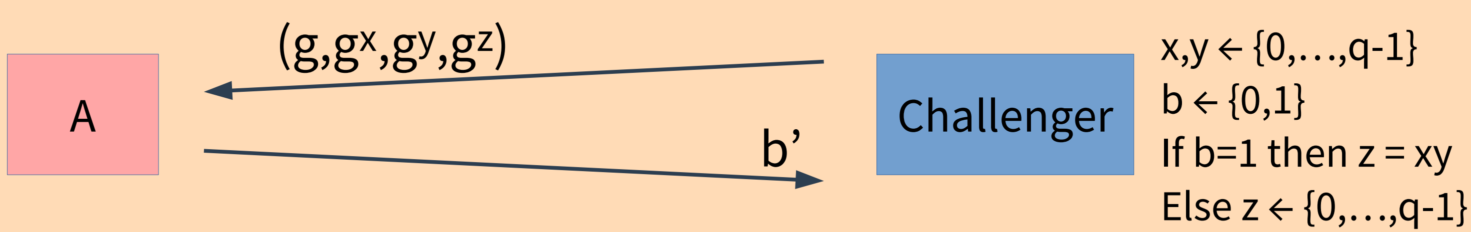


(black-box means that A_X uses A_{cpa} as a black box, without looking inside)

The Decisional Diffie-Hellman assumption

- **Example: “if DDH holds, then ElGamal is IND-CPA secure”**
- **… but wait, first let’s talk about groups**
- **Popular platform in cryptographic constructions: cyclic groups**
  - Given: cyclic group **G=<g>** of (not necessarily prime) order **q**
  - Useful: hard problems in **G**, e.g., “Decisional Diffie-Hellman” (DDH):

The <u>Decisional Diffie-Hellman (DDH) assumption</u> in a group G=<g> of order q says that no efficient adversary A can win the following experiment with probability significantly larger than ½:

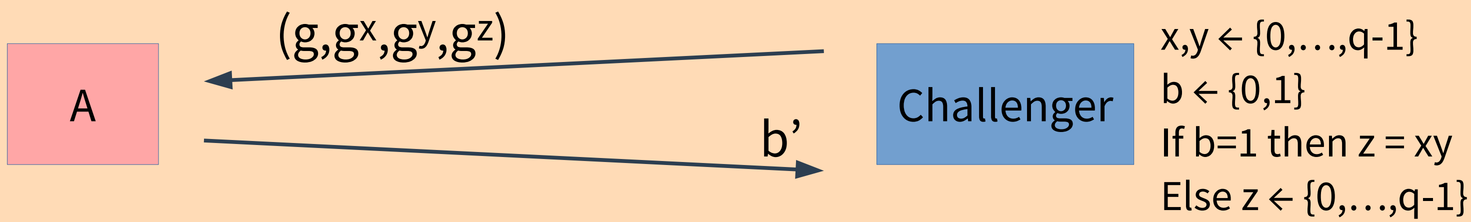


A finally wins iff b=b’

  - Intuition: “hard to recognize products in the exponent”

More on DDH

The Decisional Diffie-Hellman (DDH) assumption in a group G=<g> of order q says that no efficient adversary A can win the following experiment with probability significantly larger than ½:



A finally wins iff b=b'

- **(Un)realistic platforms for the DDH assumption**
  - Plausible: DDH holds in subgroups of elliptic curves (size around 2^{200} elements)
  - DDH does **not** hold in Z_p^* (multiplicative group of prime field with p elements)
  - Plausible: DDH in prime-order subgroups of Z_p^* (size of p around 2000 bits)
  - Fun fact: DDH does **not** hold in commuting subgroups of braid groups

More on DDH

The Decisional Diffie-Hellman (DDH) assumption in a group $G=\langle g\rangle$ of order q says that no efficient adversary A can win the following experiment with probability significantly larger than ½:

A

(g, g^x, g^y, g^z)

b'

Challenger

$x,y \leftarrow \{0,\ldots,q\text{-}1\}$
$b \leftarrow \{0,1\}$
If $b=1$ then $z = xy$
Else $z \leftarrow \{0,\ldots,q\text{-}1\}$

A finally wins iff $b=b'$

- **Best algorithms (on plausible platforms) on DDH assumption**
  - Often: best known DDH-solvers actually solve discrete logarithm (DL) problem
    - Compute x from g^x, then compare $(g^y)^x = g^{xy}$ with g^z
    - Notable exceptions: "gap groups" in which DDH is easy, but DL (presumably) hard
  - Best known DL-solvers: index calculus (for Z_p^*-subgroups), generic algorithms

The ElGamal PKE scheme

- Gen() picks generator g and $x \leftarrow \{0,\ldots,q\text{-}1\}$, outputs

$$pk = (\, g,\, X = g^x \,), \qquad sk = x$$

- Enc(pk,M) picks $y \leftarrow \{0,\ldots,q\text{-}1\}$, outputs

$$C = (\, Y = g^y,\, D = X^y M \,)$$

- Dec(sk,(Y,D)) outputs

$$M = D/Y^x$$

- Message space is G, correctness holds because $X^y = g^{xy} = Y^x$

More on the ElGamal PKE scheme

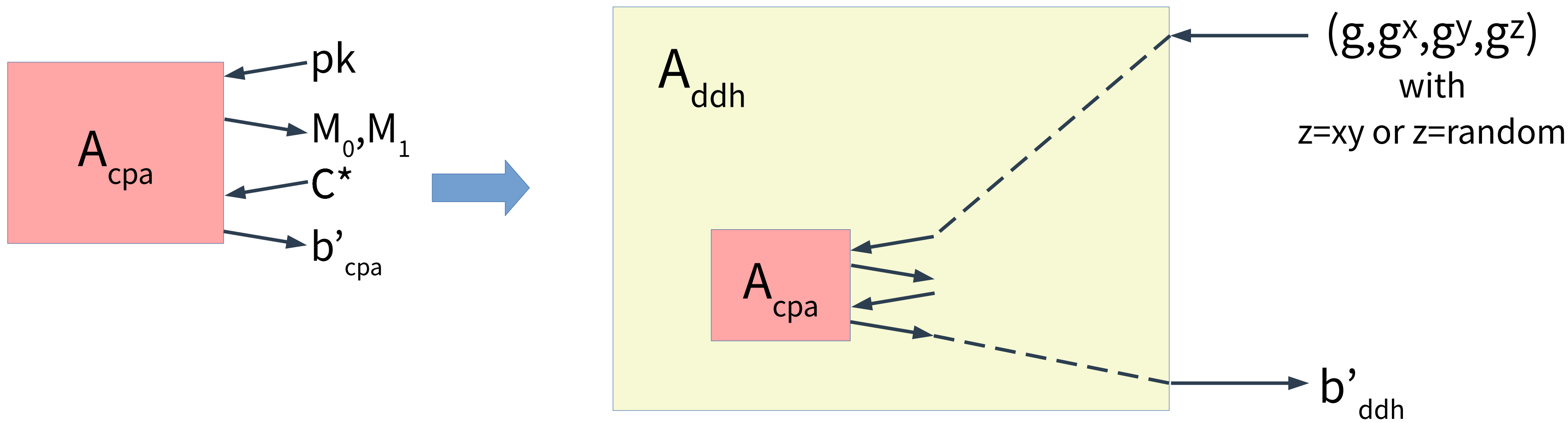
$pk = (g, X=g^x)$, $sk = x$, $C = (Y=g^y, D=X^y M)$

- **Invented in 1985 (but really Diffie-Hellman key exchange in disguise)**
- **Randomized (unlike textbook RSA)**
- **Can drop D, then acts as "key encapsulation mechanism"**
- **Security of ElGamal:**
  - IND-CPA secure under the DDH assumption (more on this here)
  - One-way secure under the CDH assumption
  - "Malleable" (homomorphic): ciphertexts can be altered (more on this later)
  - "Tightly secure": many instances as secure as one instance

Security of ElGamal

$pk = (g, X=g^x)$, $sk = x$, $C = (Y=g^y, D=X^y M)$

- **Security of ElGamal: IND-CPA secure under the DDH assumption**
  - We need to show: IND-CPA adversary A_{cpa} implies DDH-solver A_{ddh}:

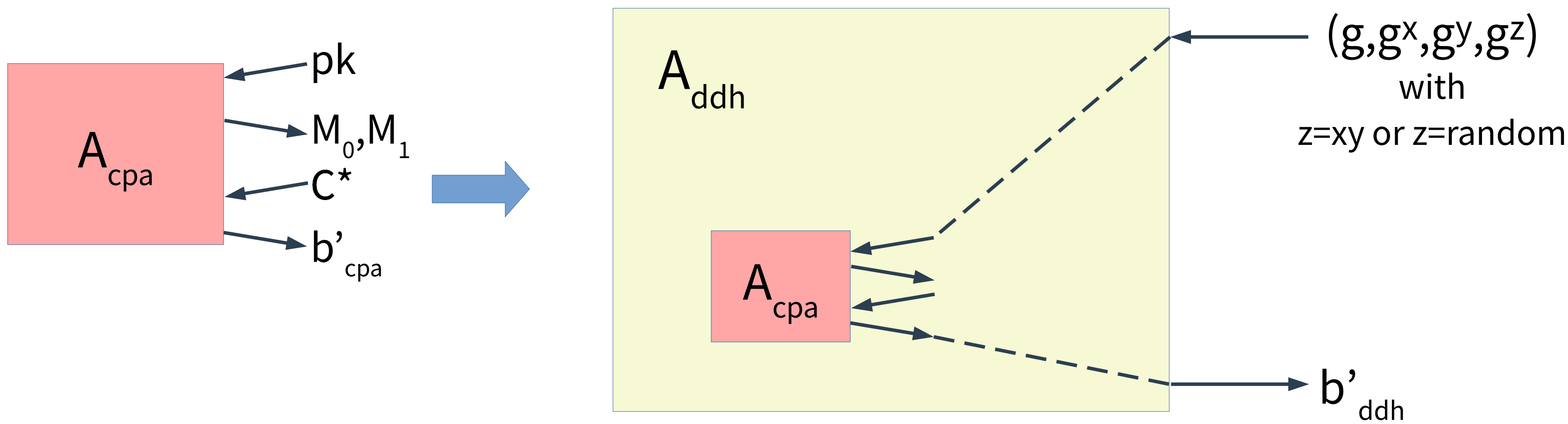


  - Idea: A_{ddh} sets $pk = (g, g^x)$, $C^* = (g^y, g^z M)$
  - But for which M? And what if z=random? Then C not encryption of M_0 or M_1!
  - Also: how do we set b'_{ddh}? Same as b'_{cpa}? Would that help?

Security of ElGamal

$pk = (g, X{=}g^x)$, $sk = x$, $C = (Y{=}g^y, D{=}X^y M)$

- **Security of ElGamal: IND-CPA secure under the DDH assumption**
  - We need to show: IND-CPA adversary A_{cpa} implies DDH-solver A_{ddh}:



  - Idea: A_{ddh} sets $pk = (g, g^x)$, $C^* = (g^y, g^z M_{b''})$ for $b'' \leftarrow \{0,1\}$, and $b'_{ddh} = [\, b'' == b_{cpa} \,]$
  - If $z{=}xy$, then A_{cpa} sees IND-CPA game and $\Pr[\, b'_{ddh}{=}1 \,] = \Pr[\, A_{cpa}$ wins IND-CPA $]$
  - If $z{=}$random, then A_{cpa}'s view independent of b'', and $\Pr[\, b'_{ddh} = 1 \,] = ½$
  - So: if A_{cpa} wins IND-CPA w.prob. $\gg ½$, then A_{ddh} wins w.prob. $\gg ½$

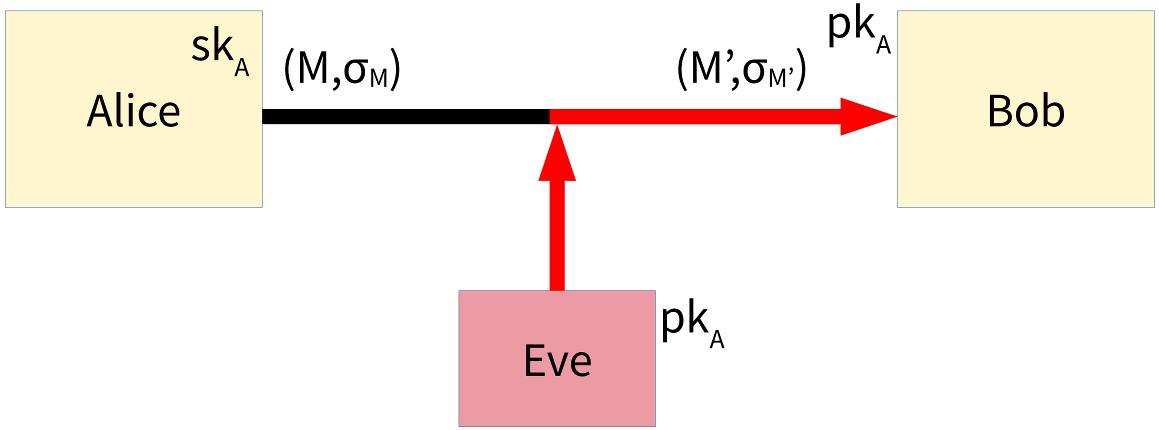
More on the security of ElGamal

- **So what have we gained?**
  - Security reduced to “simpler” assumption about problem in groups
  - Hope: simpler assumption easier to check/verify (than scheme directly)
  - Also: simpler assumption may imply security of many schemes
  - Intuitive statement: only way to break scheme is to solve DDH
  - “Win-win” situation: scheme secure or algorithmic progress

- **Does that mean every use of ElGamal is secure?**
  - Depends on application, but ElGamal will do its job (of being IND-CPA secure)
  - Example of “wrong use” of ElGamal: auctions (in a few slides)

Interlude: digital signatures

- **Another intuitive and important building block: digital signatures**
  - Motivation: digital analogue of hand-written signature
  - Intuitive goal: tie (digital) document to signer, preserve integrity

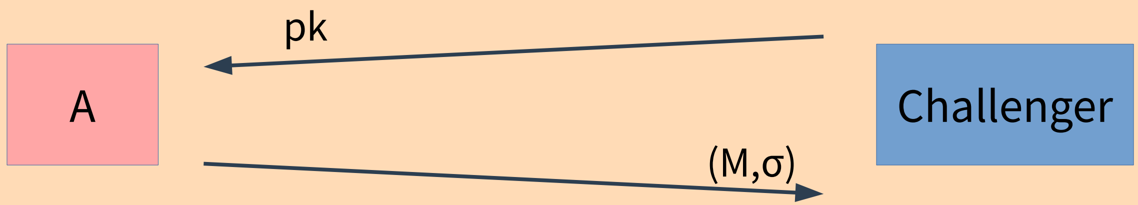


  - Formally, a digital signature scheme SIG=(Gen,Sig,Ver) consists of:
    - Key generation: Gen() outputs a keypair (pk,sk)
    - Signing: Sig(sk,M) outputs a signature σ for message M
    - Verification: Ver(pk,M,σ) outputs a bit ("verdict")
  - Correctness: Ver(pk,M,Sig(sk,M))=1 always

Interlude: digital signatures

- **Security of digital signatures**
  - Intuition: should be hard to forge signatures for yet-unsigned messages
  - First attempt at security definition:

A signature scheme SIG=(Gen,Sig,Ver) is <u>existentially unforgeable under no-message attacks (EUF-NMA secure)</u>, if no efficient adversary A can win the following experiment with significant probability:



A finally wins iff Ver(pk,σ,M)=1

  - Not very useful:
    - Intuitive reason: application probably has some honestly signed messages
    - Technical reason: trivial schemes (e.g., with σ=sk for any message M!) secure

Interlude: digital signatures

- **Security of digital signatures (second attempt)**
  - Intuition: should be hard to forge signatures for yet-unsigned message
    - … even if adversary already knows honestly generated signatures

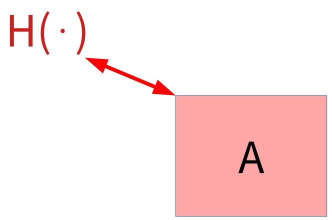
A signature scheme SIG=(Gen,Sig,Ver) is existentially unforgeable under chosen-message attacks (EUF-CMA secure), if no efficient adversary A can win the following experiment with significant probability:

A finally wins iff Ver(pk,σ,M)=1 and M fresh (i.e., never queried to Sig(sk,·))

  - Sig-oracle allows A to get a view like in an application with many signatures
    - EUF-CMA standard security notion for digital signatures, achievable and useful
    - More (in particular on constructions) in "Digital Signatures" lecture

Interlude: random oracle model

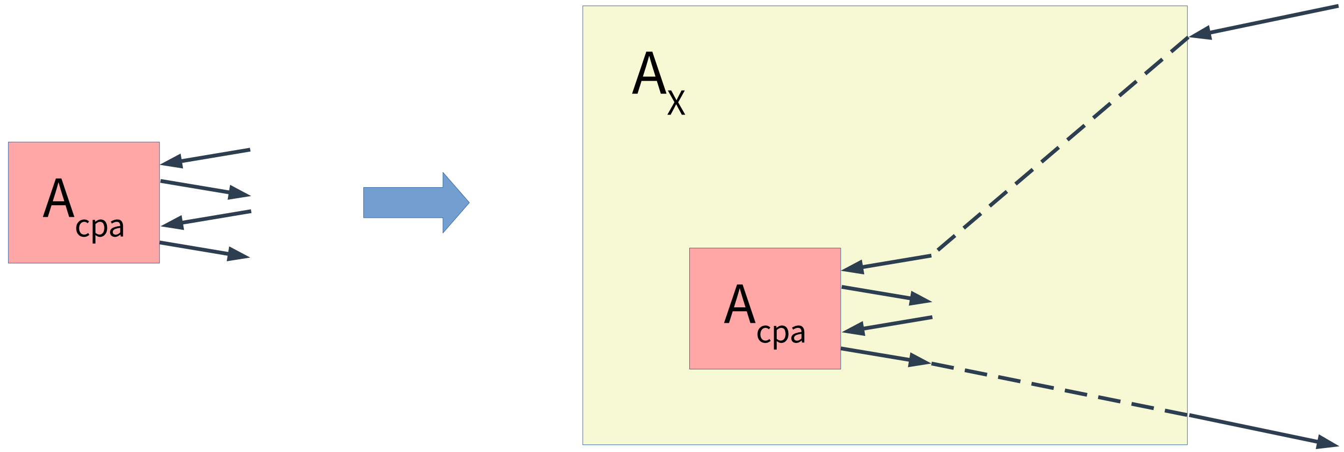
- **Sometimes difficult to prove (PKE or signature) schemes secure**
  - Compromise: consider “idealized” hash function (“random oracle”)
    - Scheme uses hash function H
    - In real world, H is implemented with a concrete hash function (e.g., SHA-3)
    - But in analysis/reduction, H is treated as truly random function (“random oracle”)



    - Both adversary and challenger/scheme have only has black-box access to H
    - Enables reductions for simple and elegant schemes (→ task!)
    - … but analysis now only shows security for over-idealized version of scheme

Cryptographic reductions: recap

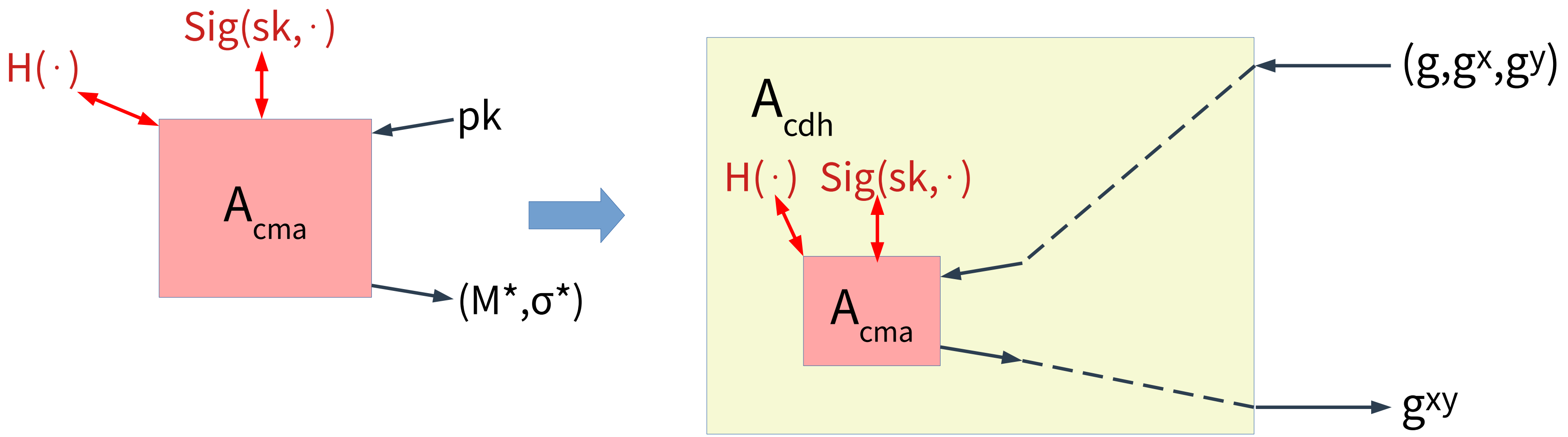
- **Motivation for security definitions**
  - Goal: interface between building block designers and users
- **Our goal here: designing/analyzing building blocks (not: using them)**
  - Basic tool: cryptographic reduction
  - Convert (hypothetical) adversary into problem-solver



  - Examples: ElGamal is IND-CPA secure under DDH assumption
- **Today: examples of more complex definitions/reductions**

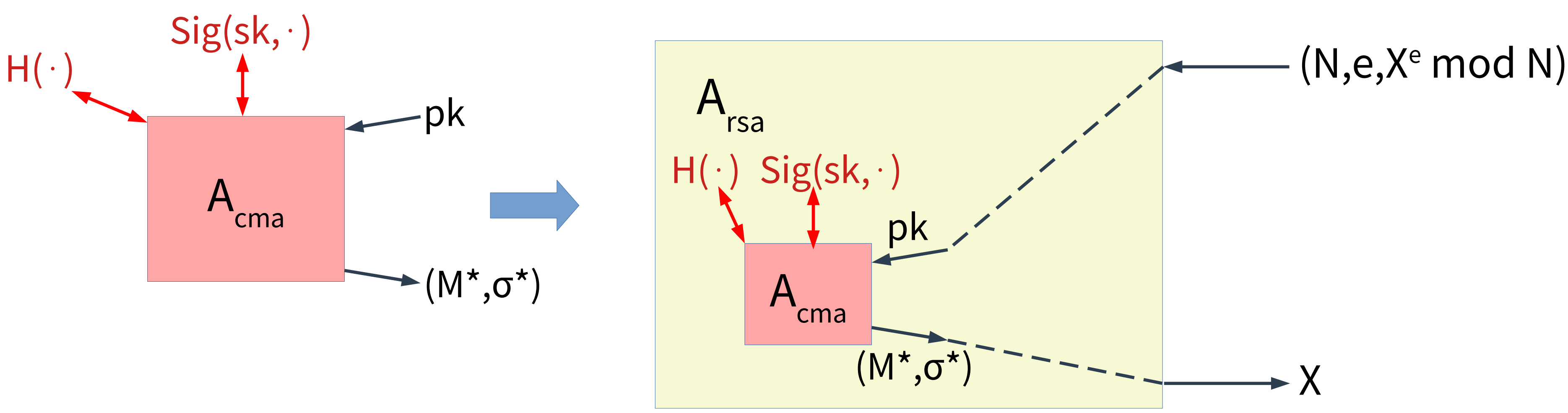
Interlude: digital signatures

- **Hence: strategy to prove security…**
  - …in the sense of EUF-CMA security for a given signature scheme…
  - …in the random oracle model…
  - …under the CDH (Computational Diffie-Hellman) assumption
- **Need to construct/implement A_{cdh} in the following diagram**
  - This requires implementing also Sig(sk,·) and H(·) for given A_{cpa}

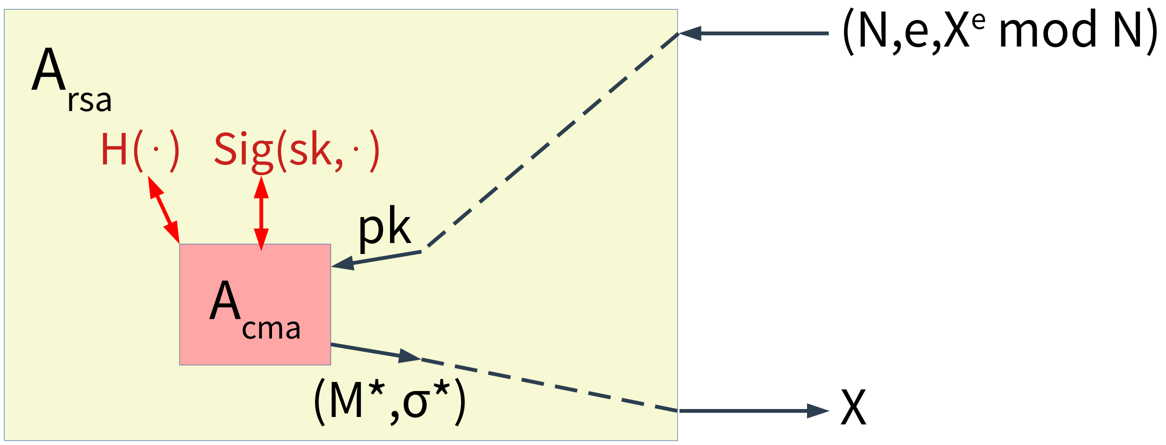


Interlude: digital signatures

- **Example: RSA-FDH ("RSA full domain hash") signature scheme:**
  - pk = (N,e), sk = (N,d), for N = PQ and $d = e^{-1} \bmod (P-1)(Q-1)$
  - $\mathrm{Sig}(sk,M) = H(M)^d \bmod N$, $\mathrm{Ver}(pk,M,\sigma) = 1$ iff $\sigma^e = H(M) \bmod N$
- **RSA assumption: given $(N,e,X^e \bmod N)$, hard to find X**
  - To prove that RSA-FDH EUF-CMA secure under RSA assumption in ROM:



  - Need to set **pk**, convert **(M*,σ*)** to **X**, and implement **H** and **Sig** oracles



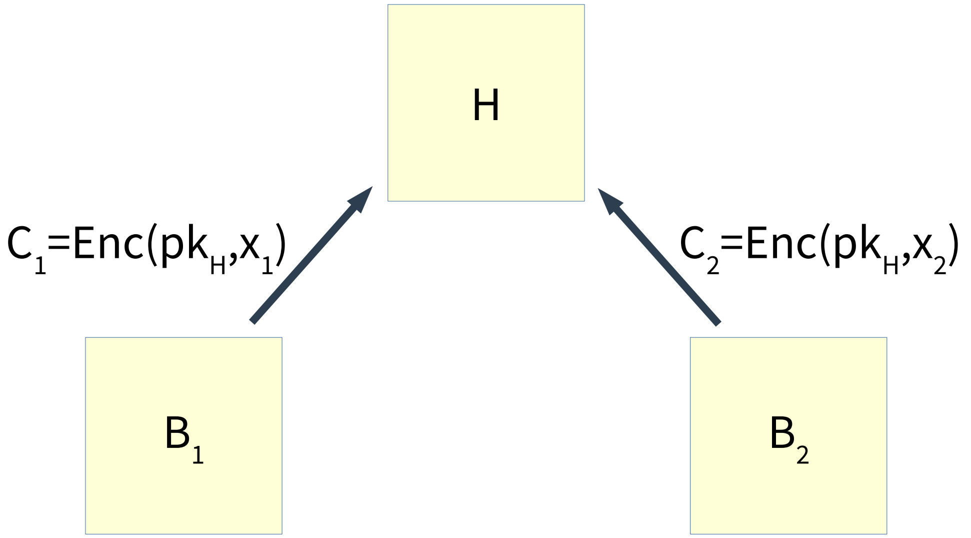
- **Specifically: implementation of A_{rsa}:**
  - Easy: set $pk = (N,e)$ for (N,e) from RSA assumption input
  - Problem: do not know sk, cannot implement Sig oracle directly
  - Solution: answer $H(M)$ queries such that $Sig(M)$ can be computed
  - Concretely (specific for RSA-FDH, different strategy for other schemes):
    - If $H(M) = R^e \bmod N$ for random, known R, then $Sig(M) = R$
    - If $H(M) = X^e \bmod N$ for given challenge X^e, then only valid signature is X
    - $\rightarrow$ guess which H-query refers to M^*, set $H(M^*)=X^e$, else $H(M_i)=R_i^e$

Interlude: digital signatures

… back to our main program…

When not to use ElGamal/IND-CPA security

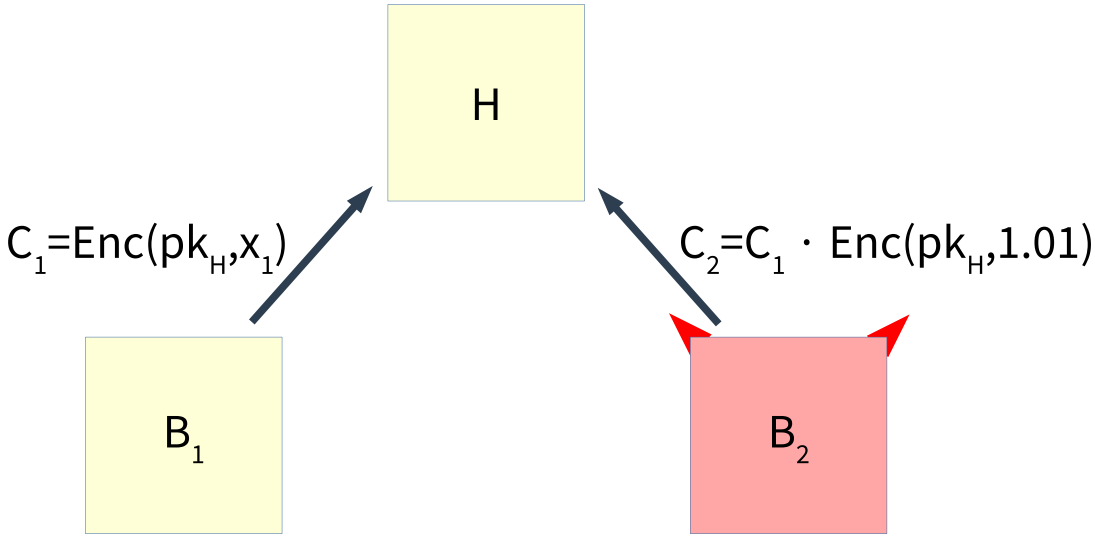
- **Imagine a digital auction with host/auctioneer H and bidders B_1, B_2:**
  - H wants to sell an item, bidders want to spend as little money as possible
  - Bids are encrypted numbers (amounts) under H's public key pk_H
  - Channel to/from H itself insecure (authenticated but not secret)



  - H sells to bidder B_i with $x_i > x_{3-i}$

When not to use ElGamal/IND-CPA security

- **Now imagine B_2 is malicious**
  - Assume encryption scheme is multiplicatively homomorphic
  - That is: $\mathrm{Enc}(pk, x) \cdot \mathrm{Enc}(pk, y) = \mathrm{Enc}(pk, x \cdot y)$ (for a suitable $\cdot$ operation)
  - Then, B_2 could wait for B_1's bid and choose its own bid adaptively:



  - Note: x_1 hidden, but $x_2 = x_1 \cdot 1.01 > x_1$, and B_2 gets the item
  - This attack works for ElGamal (depending on encoding of bids as $\mathbb{G}$-elements)

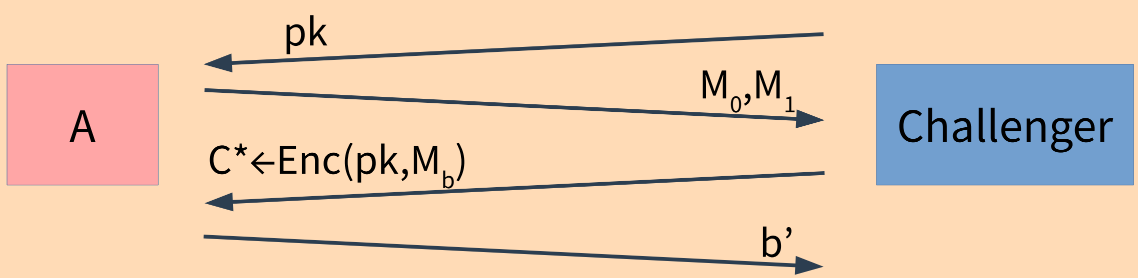
… now what?

- **What is the lesson learned from this auction example?**
  - Is IND-CPA security the wrong notion of security?
    - … in this particular context, yes
    - … but sometimes, only security against passive adversaries required
  - Avoid multiplicatively homomorphic schemes?
    - … but sometimes, homomorphic properties can be useful
  - Need security notion that guarantees security against active adversaries?
    - … by all means! (Next up)

Security against active adversaries

- **Recall our definition of IND-CPA security:**

A PKE scheme PKE=(Gen,Enc,Dec) is <u>indistinguishable under chosen-plaintext attacks (IND-CPA secure)</u>, if no efficient adversary A can win the following experiment with probability significantly larger than ½:



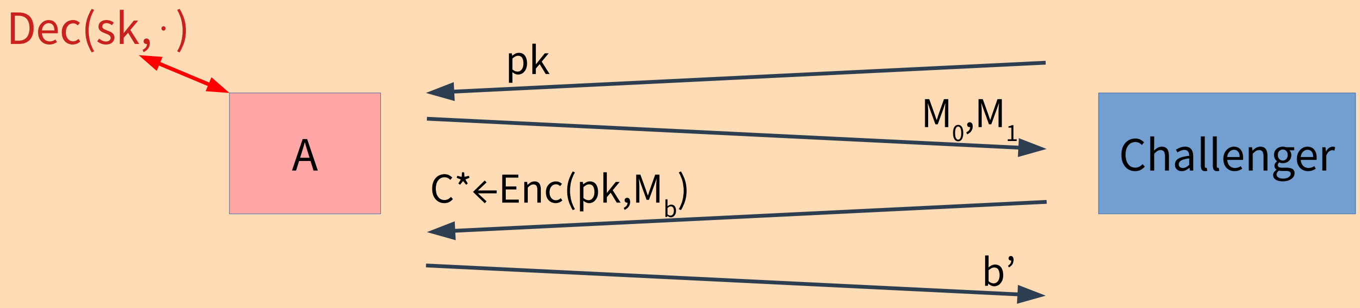
A finally wins iff $|M_0|=|M_1|$ and b=b'

  – We will start from this notion since it's simpler than semantic security
  – Here, A only listens, gets no feedback on how modified ciphertexts decrypt
    - First attempt: give A decryption oracle (i.e., access to Dec(sk,·))
    - Problem: A could decrypt challenge ciphertext C* with this oracle
    - Second attempt: give A decryption oracle Dec(sk,·) that does not work on C*

Security against active adversaries

- **Our new definition of IND-CCA security** (changes in red):

A PKE scheme PKE=(Gen,Enc,Dec) is <u>indistinguishable under chosen-ciphertext attacks (IND-CCA secure)</u>, if no efficient adversary A can win the following experiment with probability significantly larger than ½:

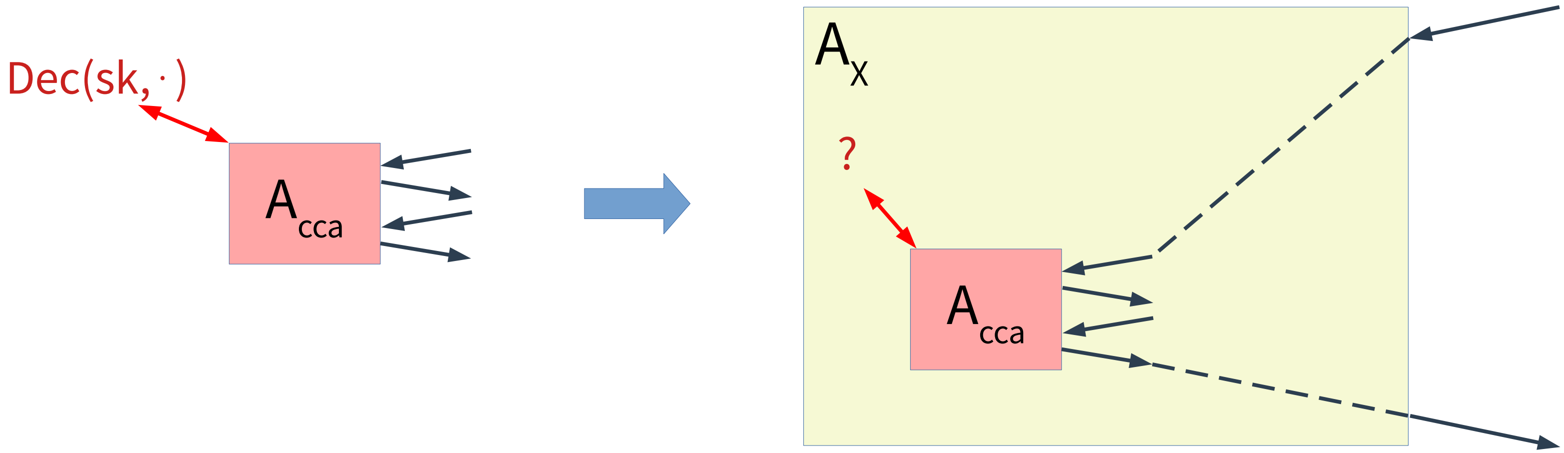


A finally wins iff $|M_0|=|M_1|$ and b=b'

  - Decryption oracle will reject query **Dec(sk,C*)**
  - IND-CCA security equivalent to semantic security against active adversaries
  - IND-CCA security implies secure channels (from authenticated channels) against active adversaries
  - Most popular notion for new PKE scheme proposals

The trouble with IND-CCA security

- **IND-CCA security is hard to achieve:**
  - Proposed in 1989 (in weaker form), first efficient scheme 1998 (or 1993, depending on whether you accept proof heuristics)
  - Implementing the decryption oracle during a reduction is difficult:



  - Dilemma:
    - If A_X knows **sk**, then A_{cca} is not very helpful, since A_X could break scheme on its own
    - If A_X does not know **sk**, then it is not clear how A_X can answer A_{cca}'s **Dec** queries

The trouble with IND-CCA security

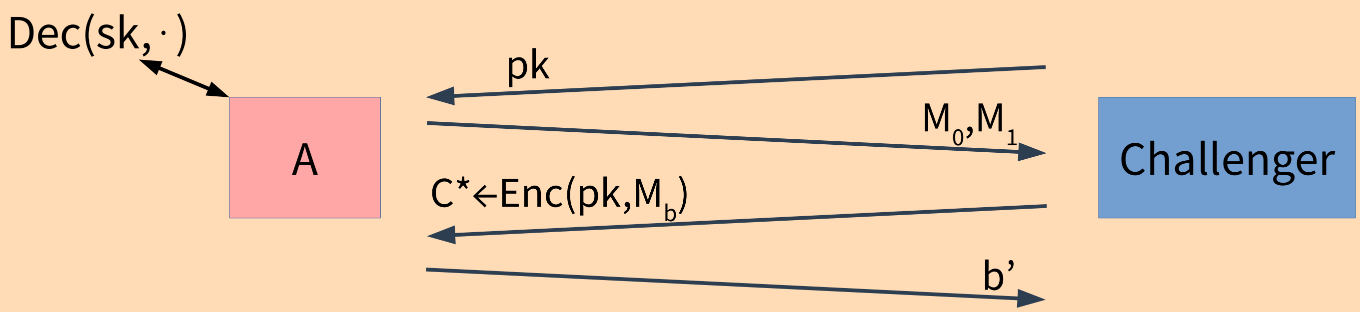
- **Possible ways to overcome IND-CCA dilemma:**
  - Dilemma:
    - If A_x knows **sk**, then A_{cpa} is not very helpful, since A_x could break scheme on its own
    - If A_x does not know **sk**, then it is not clear how A_x can answer A_{cpa}'s **Dec** queries
  - Kobayashi-Maru solution: random oracle model (→ RSA-FDH assignment)
  - Possible: A_x knows "all-but-one" secret keys that allow to decrypt all $C \neq C^*$
    - Difficulty: need a lot of structure of produce such keys
  - Or: give "special" C^* to A_x, while Dec only decrypts "normal" C correctly
    - Difference to "all-but-one" **sk**: **Dec** works on all C, but gives wrong result on some
    - Difficulty: need a lot of structure to define "special" and "normal" C

- **Next up: example of (almost) IND-CCA secure scheme and reduction**

Towards IND-CCA security: IND-CCA1

- **Stepping stone: IND-CCA1 security (changes to IND-CCA in red):**

A PKE scheme PKE=(Gen,Enc,Dec) is indistinguishable under lunchtime chosen-ciphertext attacks (IND-CCA1 secure), if no efficient adversary A can win the following experiment with probability significantly larger than ½:



A finally wins iff $|M_0|=|M_1|$ and b=b'

Here, Dec(sk,·) is only available to A before receiving C*

  - Naming: victim left its computer (decryption ability) unlocked during lunch
  - Guarantee between IND-CPA and IND-CCA
  - Achieving IND-CCA1 already requires solving the IND-CCA dilemma

Simplified Cramer-Shoup

- **Simplified version of Cramer-Shoup cryptosystem:**
  - Setting: group $G=\langle g \rangle$ (as with ElGamal)
  - Public key: $pk = (\, g,\ X=g^x,\ U=g^c X^d\,)$
  - Secret key: $sk = (c, d)$
  - Assume mapping H: $G \to \{0,1\}^{2n}$
    - Requirement: H(uniform input) = close-to-uniform output
    - Write $H(x) = H_1(x) \,||\, H_2(x)$ for $H_1, H_2 : G \to \{0,1\}^n$
  - Ciphertext for $M \in \{0,1\}^n$: $C = (\, R=g^r,\ S=X^r,\ h=H_1(U^r),\ P=M \oplus H_2(U^r)\,)$
  - Decryption computes $T=R^c S^d$ ($=U^r$), checks $h==H_1(T)$, outputs $P \oplus H_2(T)$

More on simplified Cramer-Shoup

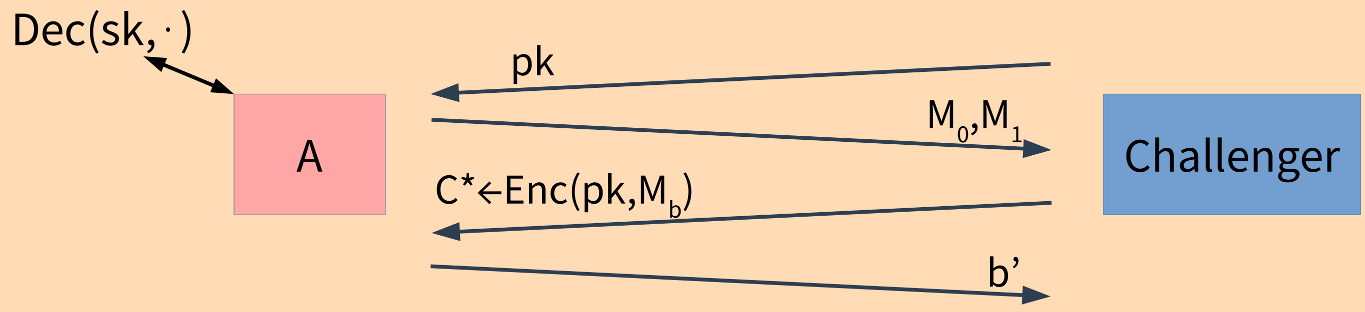
- **Simplified version of Cramer-Shoup cryptosystem:**
  - Public key: $pk = (g, X{=}g^x, U{=}g^cX^d)$
  - Secret key: $sk = (c, d)$
  - Ciphertext for $M \in \{0,1\}^n$: $C = (R{=}g^r, S{=}X^r, h{=}H_1(U^r), P{=}M \oplus H_2(U^r))$
  - Decryption computes $T{=}R^cS^d$ $(=U^r)$, checks $h{=}{=}H_1(T)$, outputs $P \oplus H_2(T)$
  - Intuition: "hashed" ElGamal with additional S and authentication tag h
    - If R or S or h is tampered with, something breaks, and decryption check fails
    - But: still malleable, since P can be tampered with (XOR-homomorphic)
    - Hence: not IND-CCA secure ("full" Cramer-Shoup also protects P)
  - Correctness: clear (since $T{=}R^cS^d{=}U^r$)
  - Important observation: pk does not uniquely fix sk

Game hopping

- **Strategy for proof (DDH ⇒ IND-CCA1 security):**
  - Several arguments necessary
  - Single reduction to DDH possible, but complex
  - Instead: “game hopping” technique (several small steps)

- **Game hopping:**
  - Start from IND-CCA1 experiment (with challenger and A)
  - Make small refinements, show that A’s winning probability is preserved
  - … until A’s winning probability is ½ by definition of changed experiment

Game 0: the original IND-CCA1 experiment

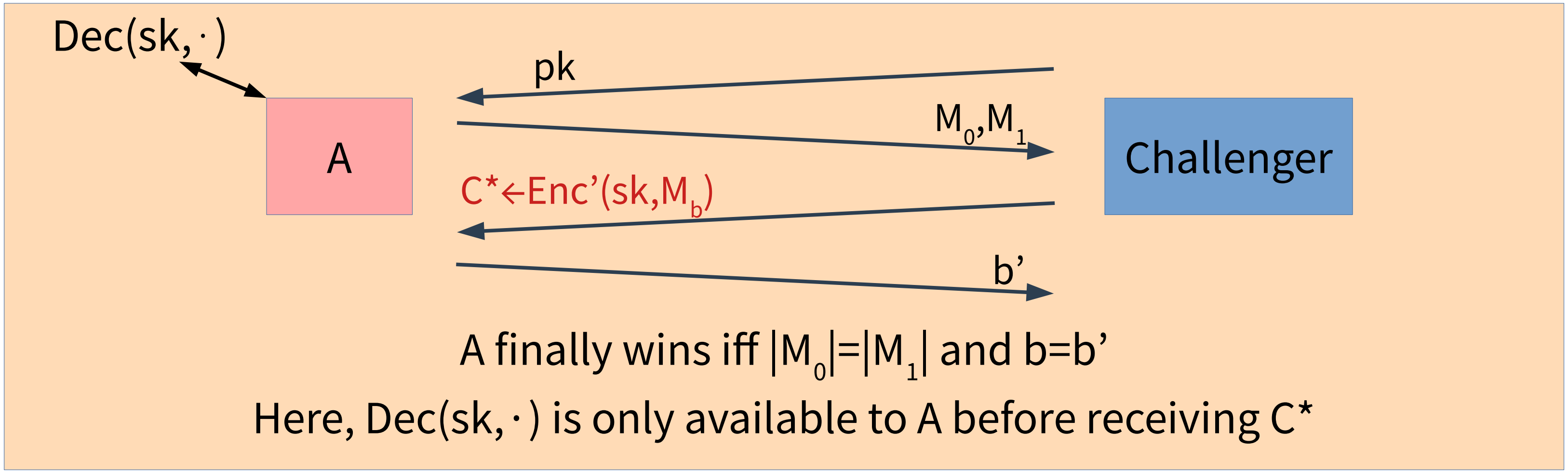


A finally wins iff $|M_0|=|M_1|$ and b=b'

Here, Dec(sk, ·) is only available to A before receiving C*

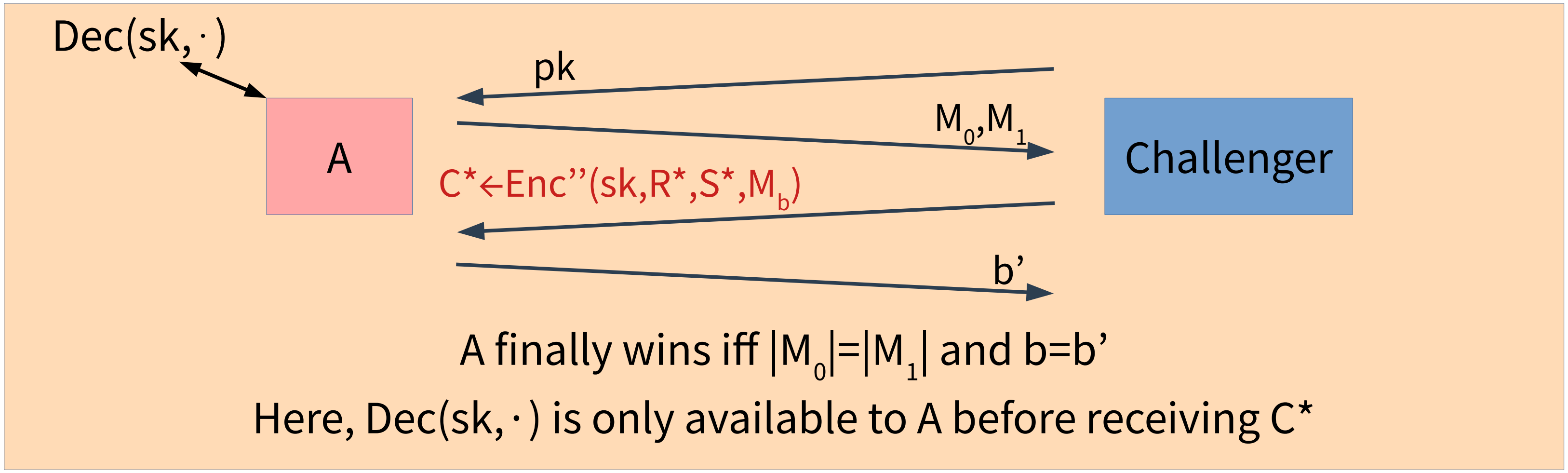
- **Rules:**
  - $pk = (g, X=g^x, U=g^cX^d)$, $sk = (c, d)$
  - $C^*=(R^*=g^{r^*}, S^*=X^{r^*}, h^*=H_1(U^{r^*}), P^*=M_b \oplus H_2(U^{r^*}))$
  - Dec(sk,(R,S,h,P)) computes $T=R^cS^d$ $(=U^r)$, checks $h==H_1(T)$, outputs $P \oplus H_2(T)$
- **Goal: want to bound Pr[A wins]**

Game 1: challenge ciphertext computed differently



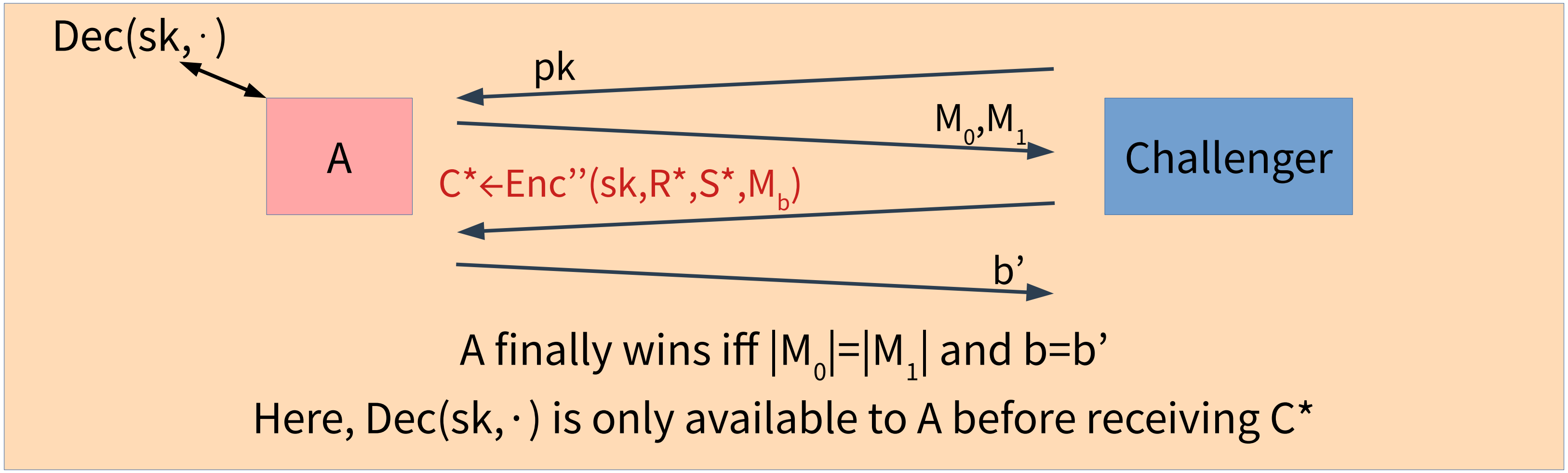
- **First modification:**
  - $pk = (g, X=g^x, U=g^c X^d),\quad sk = (c, d)$
  - $C^*=(R^*=g^{r^*}, S^*=X^{r^*}, h^*=H_1({R^*}^c {S^*}^d), P^*=M_b \oplus H_2({R^*}^c {S^*}^d))$
  - $Dec(sk,(R,S,h,P))$ computes $T=R^c S^d$ ($=U^r$), checks $h==H_1(T)$, outputs $P \oplus H_2(T)$
- **Reason for change: need r* only for computing R*,S* (but not h*,P*)**
- **Only conceptual change, no change in A's view or winning probability**

Game 2: changed challenge ciphertext



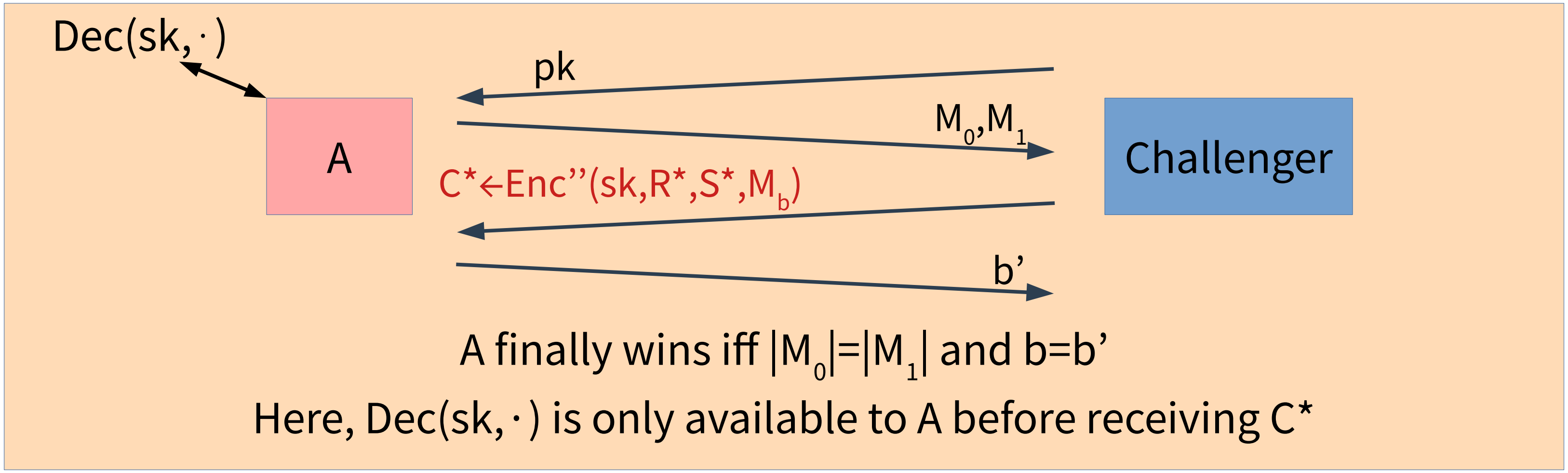
- **Second modification:**
  - $pk = (g, X=g^x, U=g^cX^d)$, $sk = (c, d)$
  - $C^*=(R^*=g^{r^*}, S^*=X^{s^*}, h^*=H_1(R^{*c}S^{*d}), P^*=M_b \oplus H_2(R^{*c}S^{*d}))$ for fresh s^*
  - Dec(sk,(R,S,h,P)) computes $T=R^cS^d$ $(=U^r)$, checks $h==H_1(T)$, outputs $P \oplus H_2(T)$
- **Intuition and justification: coming up**

Game 2: changed challenge ciphertext



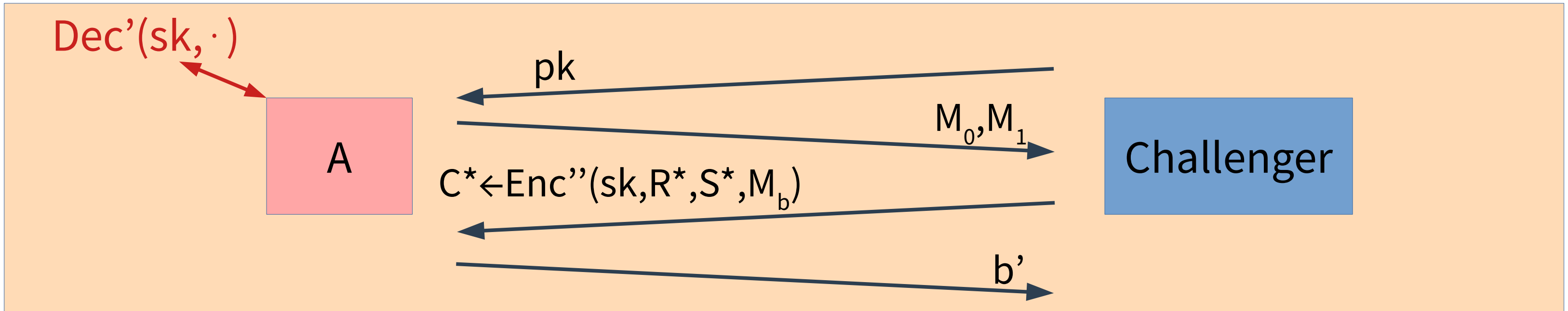
- **Justification: reduction to DDH**
  - Game 1: $R^*=g^{r^*}$, $S^*=X^{r^*}$, Game 2: $R^*=g^{r^*}$, $S^*=X^{s^*}$
  - Reduction to DDH interprets its input as (g,X,R^*,S^*), runs game above
    - If (g,X,R^*,S^*) is of the form (g,g^x,g^y,g^{xy}), then $S^*=X^{r^*}$, and this runs Game 1
    - If $(g,X,R^*,S^*) = (g,g^x,g^y,g^z)$ for random z, then $S^*=X^{s^*}$, and this runs Game 2
  - Reduction outputs whether A wins or not
  - A wins more often in Game 2 ⇒ reduction wins DDH with probability ≫ ½

Game 2: changed challenge ciphertext



- **Intuition: challenge ciphertext C* is now "special"**
  - $R^*=g^{r^*}$, $S^*=X^{s^*}$, with $r^* \neq s^*$ with overwhelming probability
  - Observation: this randomizes $T^*=R^{*c}S^{*d}$ computed during encryption of M_b:
    - Public key reveals only $U=g^cX^d=g^{c+xd}$ (i.e., one linear equation) about $sk=(c,d)$
    - For $r^* \neq s^*$, the value $T^*=R^{*c}S^{*d}=g^{r^*c+s^*xd}$ reveals different linear equation about sk
  - This blinds T^* and $H_2(T^*)$ and $P^*=M_b \oplus H_2(T^*)$ → A's view independent of b
  - Problem: Dec(sk,·) may reveal additional information about sk

Game 3: changed decryption oracle

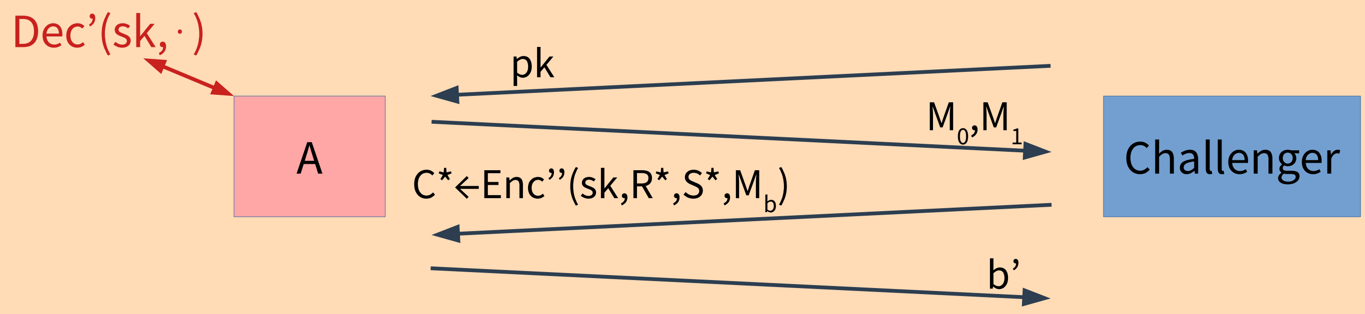


A finally wins iff $|M_0|=|M_1|$ and b=b'

Here, Dec(sk,·) is only available to A before receiving C*

- **Third modification:**
  - Dec'(sk,(R,S,h,P)) rejects ciphertext whenever $R=g^r$, $S=X^s$ for $r\neq s$
  - This makes the experiment inefficient
- **Intuition: suppress additional leakage of information about sk**
- **Justification: the original Dec would reject such ciphertexts anyway**
  - Dec would check $h==H_1(R^cS^d)$, where R^cS^d is independently random

Game 3: changed decryption oracle



A finally wins iff $|M_0|=|M_1|$ and b=b'

Here, Dec(sk,·) is only available to A before receiving C*

- **Analysis of Game 3:**
  - Dec'(sk,(R,S,h,P)) rejects ciphertext whenever $R=g^r$, $S=X^s$ for $r \neq s$
  - A now only gets information from pk, Dec about c+xd, but not r*c+s*xd
  - $P^*=M_b \oplus H_2(R^{*c}S^{*d})$ completely blinded by $H_2(R^{*c}S^{*d})=H_2(g^{r^*c+s^*xd})$
  - Hence, A's view statistically independent of b
  - Thus, A can win (b=b') only with probability exactly ½

Summary of proof strategy

- **Proof steps:**
  - Game 0: original IND-CCA1 experiment
  - Game 1: C* computed differently (using sk)
  - Game 2: S* randomized (reduction to DDH)
  - Game 3: Dec-leakage about sk prevented
- **A's winning probability preserved (up to small changes) during games**
- **A's winning probability in Game 3 exactly ½**

  ⇒ **A's winning probability in IND-CCA1 experiment close to ½**

  ⇒ **The modified Cramer-Shoup scheme IND-CCA1 secure**

Theorem: Under the DDH assumption, the modified Cramer-Shoup scheme is IND-CCA1 secure.

More on full IND-CCA security

- **Full IND-CCA security achieved by Cramer-Shoup and other schemes**
  - Idea: “authentication tag” also for message-dependent part P
  - Optimizations ⇒ $|C| = |M| + |2\ \mathbb{G}\text{-elements}| + |1\ \text{MAC tag}|$ ($|\text{MAC tag}| \approx 100$ bits)
  - Open: can we do better?
  - IND-CCA secure PKE schemes known from factoring/DDH/LWE/…
  - … but not from general one-way functions (or hash functions)
- **Does this scale to many ciphertexts/users?**
  - We have considered simple one-user, one-ciphertext definitions
  - Many-user, many-ciphertext definitions asymptotically equivalent
  - But: generic equivalence loses factor in reduction success
  - Hence: look for “tightly secure” schemes

Where are we now?

- **Achievable/intuitive security definitions (IND-CPA/IND-CCA)**
  - Intuitive because of relation to semantic security
  - Achievable by ElGamal/Cramer-Shoup (and many other schemes)

- **How about the usefulness of such definitions?**
  - Intuitive because of relation to semantic security
  - Useful in larger contexts: implies security (but not authenticity)
  - IND-CPA/IND-CCA default notions in literature

- **Next up: overview over other cryptographic building blocks**

Other cryptographic building blocks

- **Digital signatures**
  - Shameless plug: upcoming lecture on digital signatures!
  - Interesting phenomenon: generically implied by one-way functions…
  - … but efficient signatures apparently harder to achieve than efficient PKE
  - Example open problem: efficient signatures from factoring
- **Key exchange**
  - Conceptually similar to PKE, but additional properties like "forward secrecy"
  - Example open problem: non-interactive key exchange that scales well
- **More encryption variants: symmetric, identity-based, homomorphic**
  - Another shameless plug: lecture on advanced encryption schemes!

Even more cryptographic building blocks

- **Not (directly) related to secure communication**
  - Zero-knowledge protocols/schemes
    - Convince someone of the fact that you know information without revealing it
    - … e.g.: you own a certified digital passport that confirms that you are above 18
    - Highly useful in mixnets, anonymous credentials, e-voting, blockchain, …
  - Multi-party computation protocols
    - Replace trusted party with interaction for auctions, negotiations, elections, …
    - Solves many cryptographic problems with interaction, can be complex
  - Smaller building blocks: commitments, hash functions, one-way functions, …
- **Current hot topic: code obfuscation**
  - Old concept, cryptographic formalization new and very versatile
  - Open problem: (efficient) code obfuscation from standard assumptions