CS 578 – Cryptography

Prof. Michael Backes

Public-key Encryption Schemes in Prime Order Groups

June 2, 2006

Administrative Announcements

- Handouts:
  - New exercise sheet
  - Lecture notes on public-key encryption in prime order groups (today)
  - Lecture notes on the Cramer-Shoup encryption scheme (treated next Tuesday, sophisticated stuff, please read in advance!)

Recall: Modular Arithmetic mod. primes

- Let p be a prime, p huge (approx 1024 bits)
- Notation: $Z_p = \{0,1,2,\ldots, p-1\}$ with addition and multiplication modulo p
- Set of invertible elements in Z_p is $Z^*_p := \{1,2,3,\ldots,p-1\}$
- Inverse of $g \in Z^*_p$: $g^{-1} = g^{p-2}$
- Classic theorem (Fermat's little theorem):
  $\forall\ a \in Z_p \setminus \{0\}$ (mod p): $a^{p-1} = 1$ (mod p)

Recall: Modular Arithmetic mod. primes

- Solving linear equations $ax+b = 0 \pmod p$ easy: Compute: $x = -b \cdot a^{p-2}$
- Theorem (Euler): Z^*_p is a cyclic group, i.e., $\exists g \in Z^*_p: \langle g \rangle := \{1,g,g^2,g^3,\ldots,g^{p-2}\} = Z^*_p$
- Such an element g is called a generator of Z^*_p
- Theorem (Lagrange) $\forall g \in Z^*_p : ord_p(g) \mid p - 1$
- Definition (Quadratic Residues): An element $g \in Z^*_p$ is called a quadratic residue if there exists $h \in Z^*_p$ such that $g = h^2$.
- $g \in Z^*_p$ has either 0 or 2 square roots.

Recall: Modular Arithmetic mod. primes

- Definition (Legendre Symbol). The Legendre symbol of g over p is defined as

$$\left(\frac{g}{p}\right) = \begin{cases} 1 & \text{if g is QR in } Z^*_p \\ -1 & \text{if g is not a QR in } Z^*_p \\ 0 & \text{if g=0} \end{cases}$$

- Euler → $\left(\frac{g}{p}\right) = g^{(p-1)/2}$ in Z_p

Recall: Modular Arithmetic mod. primes

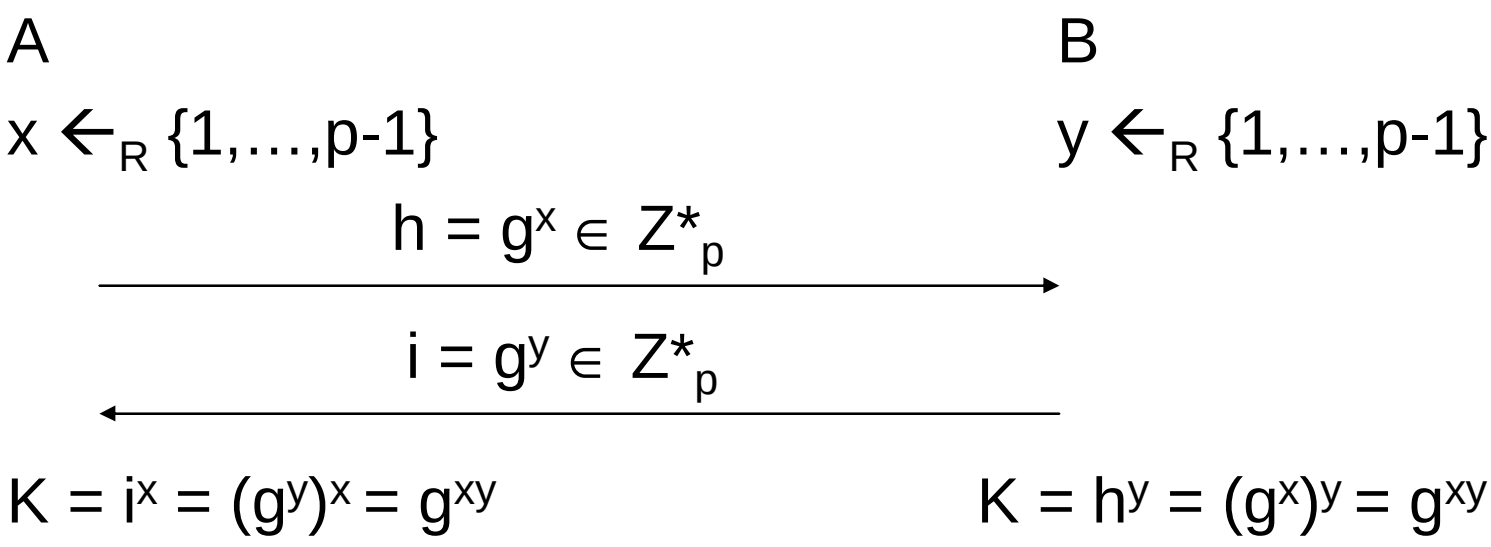
- Solving quadratic equations now easy:
  $ax^2 + bx + c = 0$ in Z_p

$$x_{1,2} = (-b \pm \sqrt{b^2 - 4ac})\,(2a)^{-1} \text{ in } Z_p$$

- Finding roots of any polynomial of degree d over Z_p doable in poly time in (d, log p)
- Complexity of basic operations:
  - Addition: O(log p), multiplication: $O(\log^2 p)$
  - Exponentiation g^x mod p (via repeated squaring): $O(\log x \cdot \log^2 p)$
- Finding random primes of bitsize n easy: Pick random n-bit number and do (randomized) primality test

Diffie-Hellman Key Exchange

- Basic Diffie-Hellman protocol, 1976
- Fix:
  - A prime p (approx. 1024 bits)
  - An element $g \in Z^*_p$ where g is a generator of Z^*_p (later over subgroups G_q of Z^*_p)



Diffie-Hellman Key Exchange

- Both parties obtain the same key
  $K = g^{xy} \in Z^*_p$
- Then K can be used to derive other keys, e.g.,
  - an AES key
  - a MAC key, …
- Called key derivation function (KDF)

Diffie-Hellman Key Exchange

- Why is basic DH secure against eavesdroppers?
- Question: Can Eve compute $K = g^{xy}$?
  More precisely, can Eve distinguish $K = g^{xy}$ from g^z for a random z?
- Can Eve compute K?
  - Eve sees: $h = g^x$, $i = g^y \in Z^*_p$
  - Eve's goal: Compute $g^{xy} \in Z^*_p$

Comp. Diffie-Hellman Problem

- Let $DH_g(g^x, g^y) := g^{xy}$ denote the Diffie-Hellman function
- Computational Diffie-Hellman Assumption (CDH):
  Given a random n-bit prime p, a random generator $g \in Z^*_p$ (or a subgroup G_q of large prime order q of Z^*_p), and random elements $x,y \in \{1,\ldots,p-1\}$ no efficient adversary (in n) can compute $DH_g(g^x, g^y)$.

Discrete Logarithm Problem

- Related function: Discrete Logarithm
- Fix a prime p (approx. 1024 bits), and element $g \in Z^*_p$ (or of a subgroup G_q of prime order q of Z^*_p),
- Definition (Discrete Logarithm). The discrete logarithm of h with respect to g, $DLog_g(h)$, is defined as the smallest integer $x>0$ s.t. $g^x = h$ in Z^*_p (or G_q),
  and $DLog_g(h) = \infty$ if no such x exists
- Examples: p = 7, g = 3
  - $Dlog_3(2) = 2$, $Dlog_3(4) = 5$

DLog and CDH

- Lemma: DLog in Z^*_p (or in G_q) easy → CDH in Z^*_p (or in G_q) also easy

→ For Diffie-Hellman Key Exchange to be secure, DLog has to be hard.

- Lots of example groups where DLog is believed to be hard.
- Converse direction: CDH easy → DLog easy?
- Open problem! (strong evidence this is true, Maurer'94)
- DLog not always hard: easy in Z^*_p for primes $p = 2^n + 1$

Random Self-reducibility

- Problem: What if DLog in Z^*_p is not hard in all of Z^*_p, i.e., what if 10% of DLog instances are easy?
- Lemma (Random Self-reducibility)
  Fix prime p and generator g of Z^*_p. Suppose there exists an algorithm A for computing $DLog_g(h)$ in time T whenever $h \in S \subseteq Z^*_p$ and $|S| = \varepsilon \cdot |Z^*_p|$.
  Then there exists an algorithm B for computing $DLog_g(h)$ for all h in expected time T/ε.

  [proof on the board]

→ even computing DLog in some fraction of Z^*_p is hard if DLog is hard

Random Self-reducibility

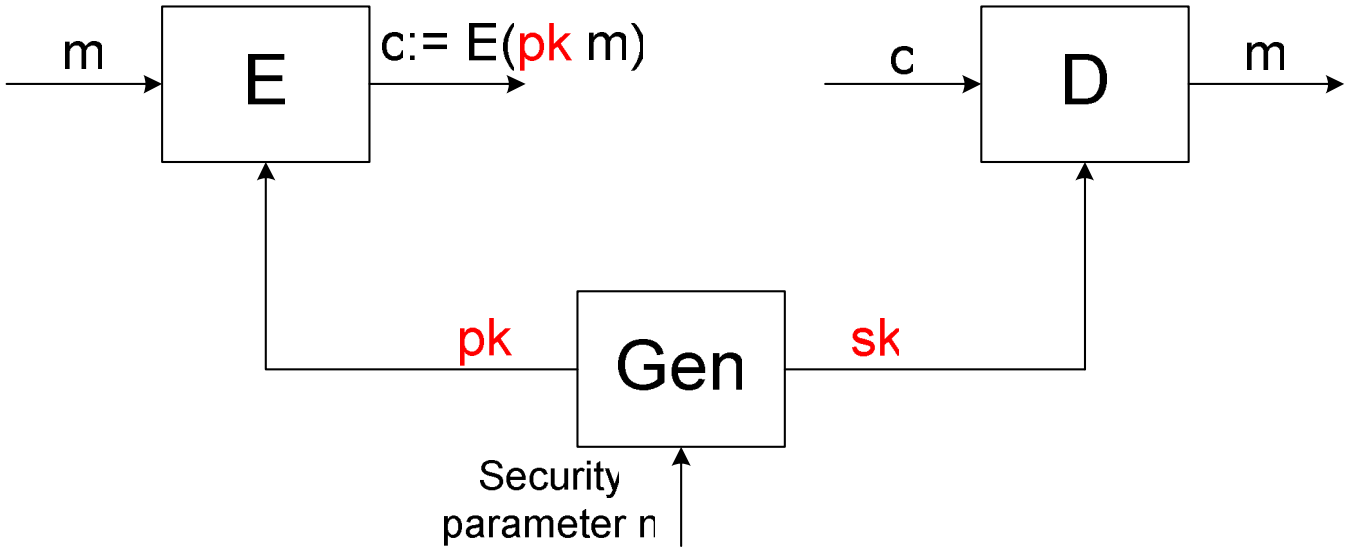
- Note: DH also has random self-reducibility
- Lemma: When $g \in Z^*_p$ is a generator, then anyone can compute $\text{lsb}(\text{DLog}_g(h))$ for all $h \in Z^*_p$

$$\left(\frac{h}{p}\right) = 1 \quad \text{iff} \quad \text{lsb}(\text{DLog}_g(h)) = 0$$

$$\left(\frac{h}{p}\right) = -1 \quad \text{iff} \quad \text{lsb}(\text{DLog}_g(h)) = 1$$

Public-key Encryption

- DH: Key exchange only !!
- Now build public-key encryption mechanisms out of it:



Public-key Encryption

- Definition (Public-Key Encryption Scheme): A public-key encryption scheme is a triple of efficient algorithms (Gen,E,D):
  - Gen(n): Generates a secret/public key pair (pk,sk) for security parameter n
  - E(pk,m) and D(sk,m) as usual

  such that for all (pk,sk) ← Gen(k), and for all m: D(sk,E(pk,m)) = m
- n is the security parameter, tacitly considered input to all algorithms (formally again sequences of encryption schemes and (uniform) adversaries, but much more natural for public-key encryption).

ElGamal Encryption System (1984)

- Generation in Z^*_p, for security parameter n:
  - Pick random n-bit prime p
  - Pick generator g of Z^*_p (sometimes publicly chosen p and g used)
  - Pick random $x \in \{1,\ldots,p-1\}$
  - Set pk := (p, g, h := g^x)
  - Set sk := (p, g, x)
  - Output (pk,sk)

ElGamal Encryption System (1984)

- Encryption Enc(pk,m) where pk = (p, g, h := g^x) and m $\in$ Z_p
  - Pick random y $\in$ {1,…,p-1}
  - Set i := g^y , k := h^y
  - Output c := (i, k $\cdot$ m) $\in$ $Z^*_p \times Z_p$
- Decryption Dec(sk,c) where sk = (p, g, x) and c = (A,B)
  - Output B / A^x
- B / A^x = B / $(g^y)^x$ = B / $(g^x)^y$ = B / h^y = m

ElGamal Encryption System for Subgroups

- Now in subgroup G_q of Z^*_p
- Technical nitpick: Now two security parameters: n for q, n* for p. Related by public function n_p: $n^* = n_p(n)$
- Generation in G_q, for security parameter n and function n_p):
  - Pick random n-bit prime q
  - Pick random $n_p(n)$-bit prime p such that q | p-1
  - Pick $g \in Z^*_p$ of order q (sometimes public q, p, g)
  - Pick random $x \in \{1,\ldots,q\}$
  - Set pk := (q, p, g, h := g^x)
  - Set sk := (q, p, g, x)
  - Output (pk,sk)

ElGamal Encryption System for Subgroups

- Encryption Enc(pk,m) where
  pk = (q, p, g, h = g^x) and m $\in$ <g> =G_q
  - Pick random y $\in$ {1,…,q}
  - Set i := g^y , k := h^y
  - Output c := (i, k $\cdot$ m) $\in$ $G_q \times G_q$

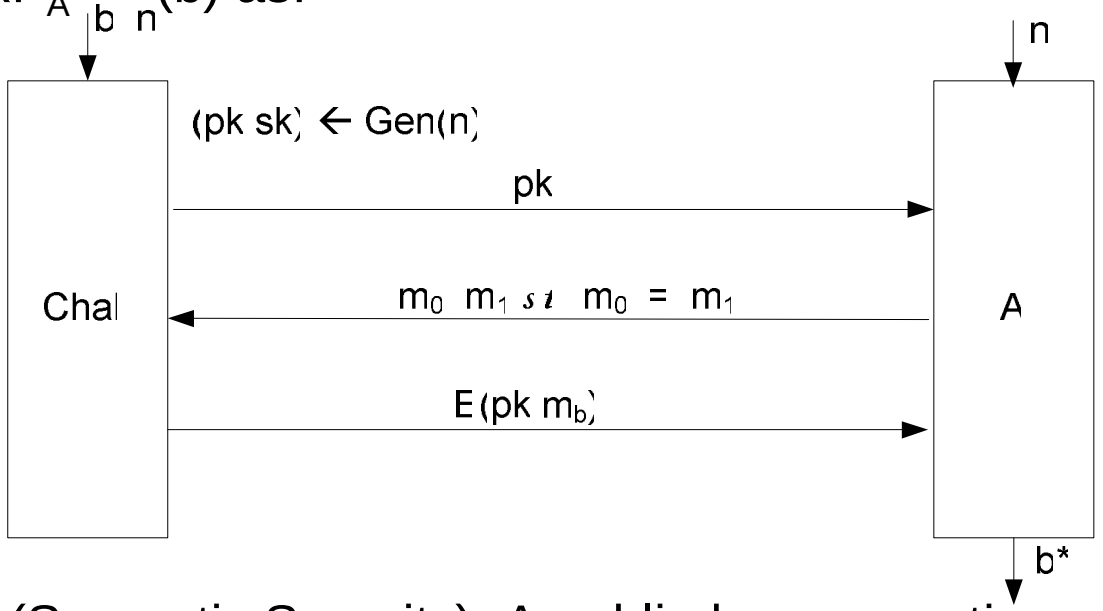
- Decryption Dec(sk,c)
  where sk = (q, p, g, x) and c = (A,B)
  - Output B / A^x

ElGamal Encryption System (cont'd)

- $B = g^{xy} \cdot m$ → DH secret is used as a OTP to encrypt m!
- Why is this secure?
  - First: Define security of public-key encryption systems
  - Then: Prove that ElGamal is secure

Semantic Security (CPA)

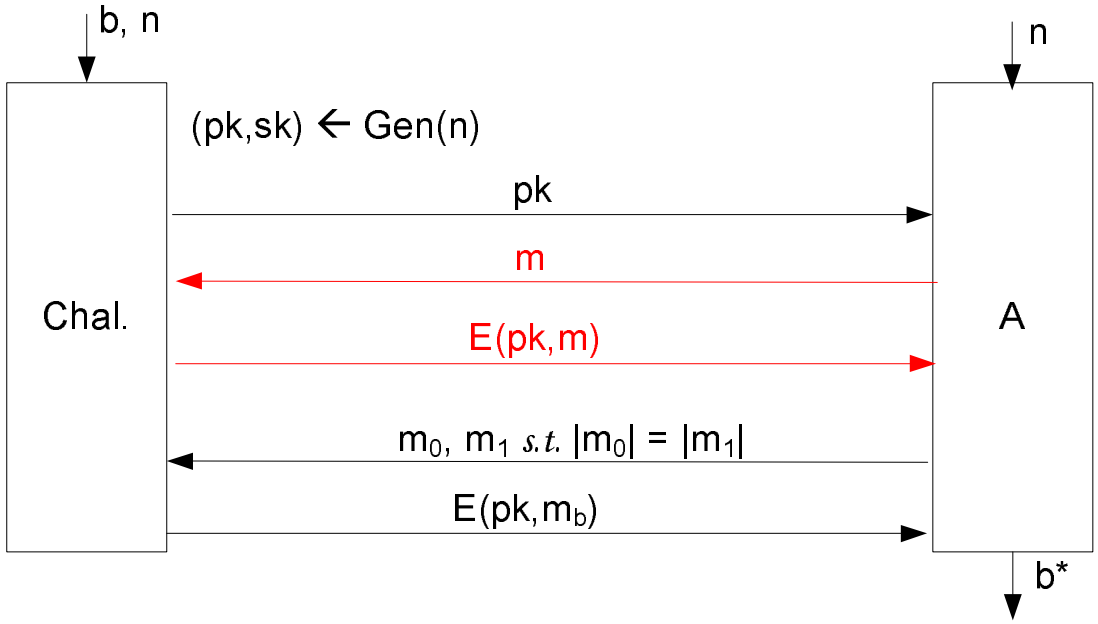
- Let PE = (Gen, E, D) be a public-key encryption scheme. Define $EXP_A^{CPA}(b)$ as:



- Definition (Semantic Security). A public-key encryption scheme PE = (Gen, E, D) is semantically secure under chosen-plaintext attack (CPA) if for all efficient adversaries A, we have that $Adv^{CPA}[A,PE] = |Pr[EXP_A^{CPA}(0)=1] – Pr[EXP_A^{CPA}(1)=1]|$ is negligible.

A Strengthening of Semantic Security?

- Does the following extended experiment strengthen the definition?



- No, since A can compute E(pk,m) itself for messages of its choice!

Proving ElGamal Semantically Secure

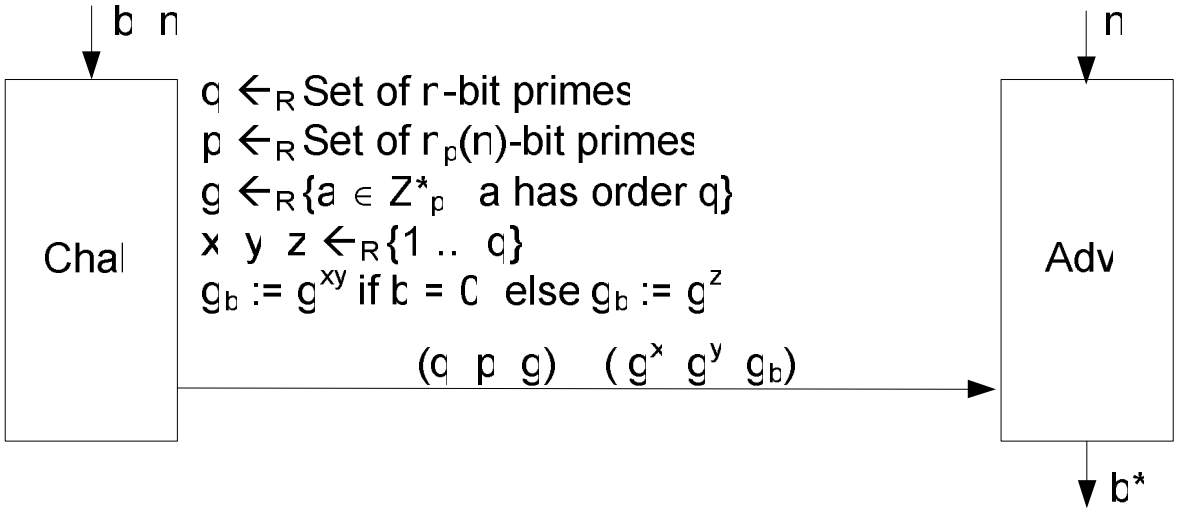
- ElGamal is semantically secure under CPA (in the subgroup G_q) if the following Decisional Diffie-Hellman assumption in G_q holds:
- Decisional Diffie-Hellman Assumption (DDH): Given n-bit prime q, $n_p(n)$-bit prime p with q | p-1, and $g \in Z^*_p$ of order q, no efficient adversary (in n) can distinguish
  (g^x, g^y, g^{xy}) and (g^x, g^y, g^z)
  for x, y, z random in {1,…,q}.
- Why decisional? CPA-security says it must be hard to distinguish, CDH that it is hard to compute. But distinguishing might be easier...

DDH as an Attack Game

- Definition (DDH Challenger, on input a security parameter n):
  - Challenger randomly chooses an n-bit prime q, an $n_p(n)$-bit prime p with q | p-1, and $g \in Z^*_p$ of order q and outputs (q, p, g)
  - Challenger chooses a bit b
  - Challenger chooses x, y, $z \in \{1,\ldots,q\}$ random and outputs
    - (g^x, g^y, g^{xy}) if b = 0 and
    - (g^x, g^y, g^z) if b = 1.
- The adversary outputs b* and wins if b* = b.

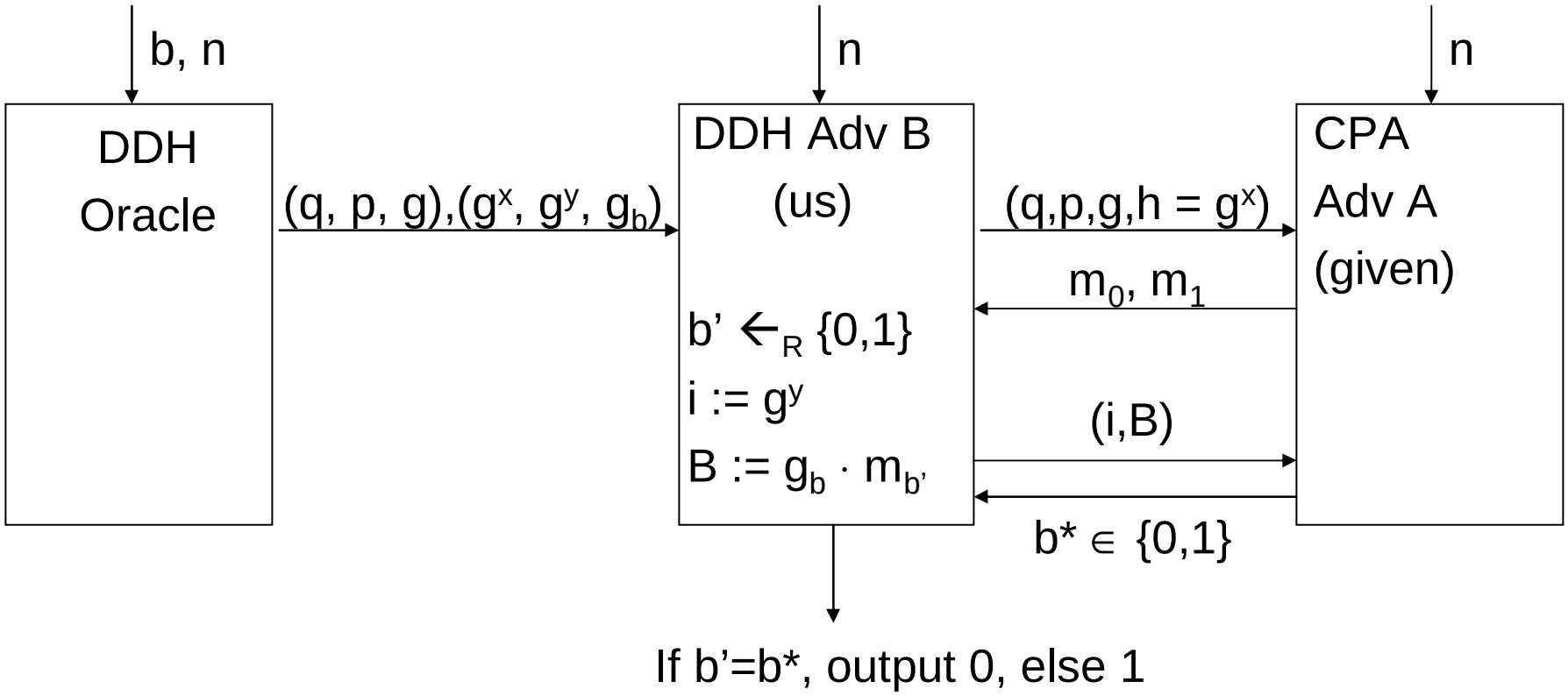
Decisional Diffie-Hellman (DDH) Game

- For b = 0,1, define $EXP_A^{DDH}(b)$ as:



- The advantage of adversary A in breaking DDH is $Adv^{DDH}[A] = |Pr[EXP_A^{DDH}(0)=1] – Pr[EXP_A^{DDH}(1)=1]|$
- DDH Assumption: $Adv^{DDH}[A]$ negligible in n for all efficient A.

CPA of ElGamal: Proof Overview



DDH in Z^*_p easy (!)

- DDH in Z^*_p easy!
- Definition would be
  - Challenger randomly chooses n-bit prime p and a generator g of Z^*_p and outputs (p,g)
  - Challenger chooses a bit b
  - Challenger chooses x, y, z $\in$ {1,…,p-1} random and outputs
    - (g^x, g^y, g^{xy}) if b = 0 and
    - (g^x, g^y, g^z) if b = 1.
- The adversary outputs b* and wins if b* = b.

DDH in Z^*_p easy (cont'd)

- Adversary receives (p,g) and (g^x, g^y, g_b) with $g_b = g^{xy}$ if b = 0 and $g_b = g^z$ if b = 1.
- Adversary computes c := lsb($DLog_g(g_b)$) (via the Legendre symbol)
- Adversary outputs 0 if c = 0, and 1 if c = 1
- Claim: Advantage is 1/4 and thus not negligible

[proof on the board]