

Computational problems: in the following, let  $G$  be a finite cyclic group generated by  $g$  with order  $q$

- Discrete log problem: sample  $x \xleftarrow{r} \mathbb{Z}_q$

given  $h = g^x$ , compute  $x$

- Computational Diffie-Hellman (CDH): sample  $x, y \xleftarrow{r} \mathbb{Z}_q$

given  $g^x, g^y$ , compute  $g^{xy}$

- Decisional Diffie-Hellman (DDH): sample  $x, y, r \xleftarrow{r} \mathbb{Z}_q$

distinguish between  $(g, g^x, g^y, g^{xy})$  vs.  $(g, g^x, g^y, g^r)$

Each of these problems translates to a corresponding computational assumption:

Definition. Let  $G = \langle g \rangle$  be a finite cyclic group of order  $q$  (where  $q$  is a function of the security parameter  $\lambda$ )

The DDH assumption holds in  $G$  if for all efficient adversaries  $A$ :

$$\Pr[x, y \xleftarrow{r} \mathbb{Z}_q : A(g, g^x, g^y, g^{xy}) = 1] - \Pr[x, y, r \xleftarrow{r} \mathbb{Z}_q : A(g, g^x, g^y, g^r) = 1] = \text{negl}(\lambda)$$

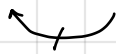
The CDH assumption holds in  $G$  if for all efficient adversaries  $A$ :

$$\Pr[x, y \xleftarrow{r} \mathbb{Z}_q : A(g, g^x, g^y) = g^{xy}] = \text{negl}(\lambda)$$

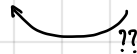
The discrete log assumption holds in  $G$  if for all efficient adversaries  $A$ :

$$\Pr[x \xleftarrow{r} \mathbb{Z}_q : A(g, g^x) = x] = \text{negl}(\lambda)$$

Certainly: if DDH holds in  $G \Rightarrow$  CDH holds in  $G \Rightarrow$  discrete log holds in  $G$



there are groups where CDH  
believed to be hard, but DDH is  
easy



Major open problem: does this hold?

Can we find a group where discrete log is hard  
but CDH is easy?

Instantiations: Discrete log in  $\mathbb{Z}_p^*$  when  $p$  is 2048-bits provides approximately 128-bits of security

$\rightarrow$  Best attack is General Number Field Sieve (GNFS) - runs in time  $2^{\tilde{O}(\sqrt[3]{\log p})}$  time

Much better than brute force -  $2^{\log p}$

$\rightarrow$  Need to choose  $p$  carefully

(e.g., avoid cases where  $p-1$  is smooth)

for DDH applications, we usually set  $p = 2q + 1$  where  
 $q$  is also a prime ( $p$  is a "safe prime") and work in the  
subgroup of order  $q$  in  $\mathbb{Z}_p^*$  ( $\mathbb{Z}_p^*$  has order  $p-1 = 2q$ )

having small prime factors

group operations all  
scale linearly (or worse) in  
bitlength of the modulus

$\uparrow$  cube root in exponent not ideal!  
if we want to double security,  
need to increase modulus by  $8x$ !  
(e.g., 16384-bit modulus for 256 bits  
of security)

Elliptic curve groups: only require 256-bit modulus for 128 bits of security

$\rightarrow$  Best attack is generic attack and runs in time  $2^{\log p/2}$

$\rightarrow$  Much faster than using  $\mathbb{Z}_p^*$ : several standards

- NIST P256, P384, P512

- Dan Bernstein's curves: Curve 25519

$\rightarrow$  Widely used for key-exchange + signatures on the web

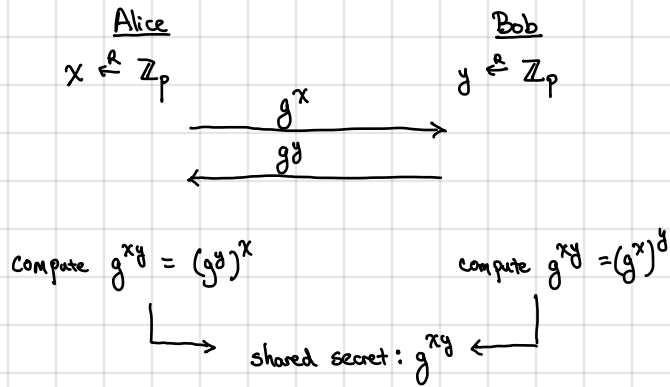
[ $p$ -algorithm - can discuss at end of semester]

} can discuss more at end of semester  
(or in advanced crypto class)

When describing cryptographic constructions, we will work with an abstract group (easier to work with, less details to worry about)

## Diffie-Hellman key exchange

- Let  $G$  be a group of prime order  $p$  (and generator  $g$ ) - choice of group, generator, and order fixed by standard



But usually, we want a random bit-string as the key, not random group element

↳ Element  $g^x$  has  $\log p$  bits of entropy, so should be able to obtain a random bit-string with  $l < \log p$  bits

↳ Solution is to use a "randomness extractor"

↳ Information-theoretic constructions based on universal hashing / pairwise-independent hashing (loses some bits of entropy)

↳ Use a "random oracle" or an "ideal hash function" [Heuristic:  $\text{SHA-256}(g, g^x, g^y, g^{xy})$ ] [binds the key to the entire transcript]

↳ Arguing security: 1. Rely on HashDH assumption  $(g, g^x, g^y, H(g, g^x, g^y, g^{xy})) \approx (g, g^x, g^y, r)$  where  $H: G^4 \rightarrow \{0,1\}^n$  and  $r \xleftarrow{R} \{0,1\}^n$

2. Model  $H$  as ideal hash function  $H: G^4 \rightarrow \{0,1\}^n$  (i.e., random oracle) and rely on CDH in  $G$  [inability to evaluate  $H$  on  $g^{xy} \Rightarrow$  output is random string]

Public-key encryption: Encryption scheme where encryption is public (does not require shared secrets)

- Setup  $(1^\lambda) \rightarrow (pk, sk)$  [generates a public/private key-pair - also called KeyGen]

- Encrypt  $(pk, m) \rightarrow c$

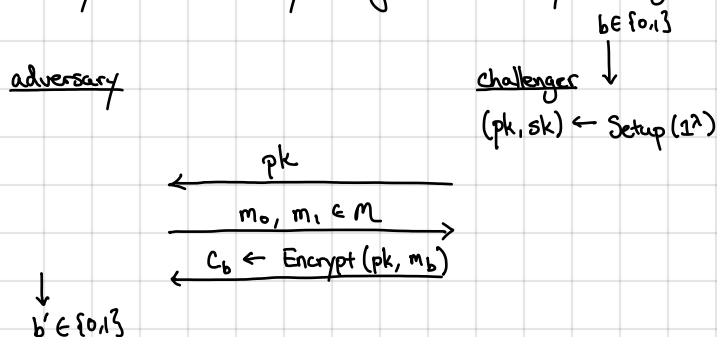
- Decrypt  $(sk, c) \rightarrow m$

Everyone can publish a public key (in a directory)

↳ Can encrypt to anyone without exchanging keys (recipient can be offline)

Correctness:  $\forall m \in \mathcal{M}: \Pr[(pk, sk) \leftarrow \text{Setup}(1^\lambda) : \text{Decrypt}(sk, \text{Encrypt}(pk, m)) = m] = 1$

Security: semantic security from secret-key setting, but adversary also gets public key



$$\text{SSAdv}[A, \Pi_{\text{PKE}}] = \left| \Pr[A \text{ outputs } 1 \mid b=0] - \Pr[A \text{ outputs } 1 \mid b=1] \right|$$