Radbound University

Diffie-Hellman key exchange and ElGamal encryption

Cryptography, Spring 2020

L. Batina, J. Daemen

April 22, 2020

Institute for Computing and Information Sciences
Radboud University

Outline

Merkle-Diffie-Hellman key exchange

ElGamal encryption

Discrete log crypto security notions

Conclusions

Merkle-Diffie-Hellman key exchange

Ralph Merkle, Martin Hellman, Whitfield Diffie

Invented public key cryptography in 1976!

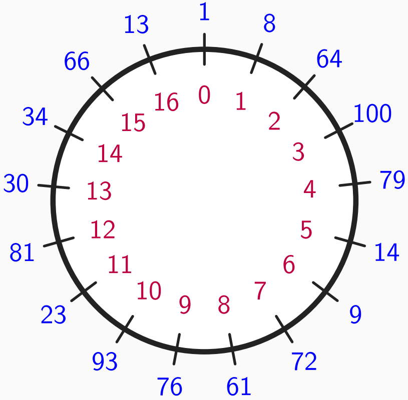
Merkle-Diffie-Hellman: cryptographic key pairs

- Domain parameters: specification of cyclic group we work in
  - Non-secret information that is common to all users
  - In this case, it consists of
    - $p \in \mathbb{N}$: prime modulus
    - $g \in (\mathbb{Z}/p\mathbb{Z})^*$: generator
  - one always takes g with large prime order $\mathrm{ord}(g) = q$
    Note: q divides $p - 1$ (due to Lagrange) so $\langle g \rangle \neq (\mathbb{Z}/p\mathbb{Z})^*$
- Every participating user has a key pair:
  - private key *PrK* that she keeps for herself: $a \in \mathbb{Z}/q\mathbb{Z}$
  - public key *PK* that she makes public: $A = g^a \in \langle g \rangle$

Key pair generation in discrete-log based crypto

$a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$
$A \leftarrow g^a$

Toy example with prime order q: $p = 103, g = 8$

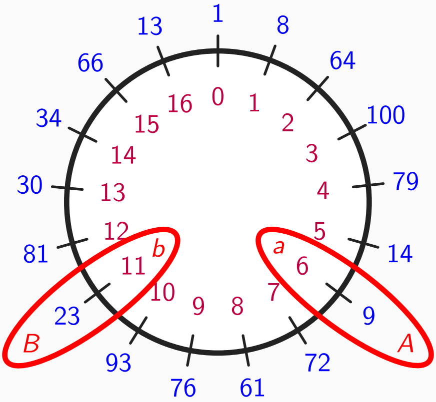


q, the order of 8, is 17

NIST: for 128 bits of security, p shall be 3072 bits long and q 256 bits

Due to discrete log solving algorithms that we will discuss later

Key pairs in our toy example



(Merkle-)Diffie-Hellman key exchange

It does **public-key based establishment of a shared secret**

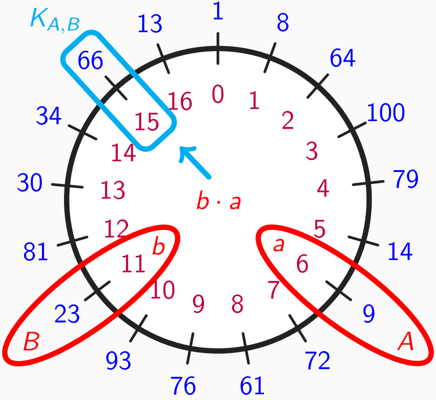
	Alice		Bob
have in advance:	$p, g, (q), a, A$		$p, g, (q), b, B$
		$\xrightarrow{\text{Alice}, A}$	
		$\xleftarrow{\text{Bob}, B}$	
	$K_{A,B} \leftarrow B^a$		$K_{B,A} \leftarrow A^b$

Alice and Bob arrive at the same shared secret $K_{A,B} = K_{B,A}$

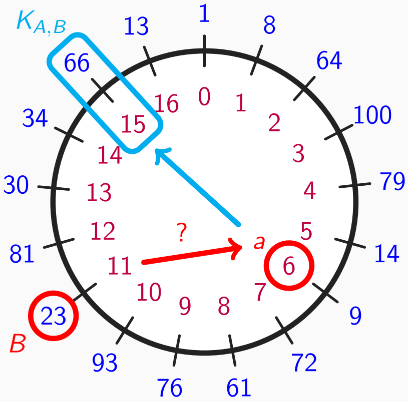
$$K_{A,B} = (B^a) = (g^b)^a = g^{b \cdot a} = g^{a \cdot b} = (g^a)^b = A^b = K_{B,A}$$

- Alice and Bob derive key(s) from secret: $K \leftarrow \mathrm{H}(\text{“KDF”}; K_{A,B})$
  - using key derivation function (KDF), in this example built from a cryptographic hash function
- This requires specifying how to encode elements of $\langle g \rangle$ as bitstrings
- They use K to encipher and/or MAC their communication

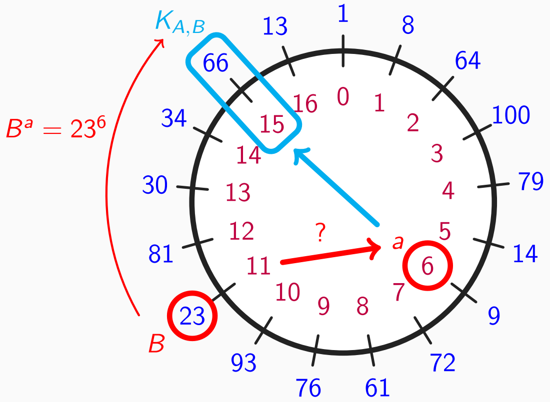
(Merkle-)Diffie-Hellman in our toy example



Alice's computation illustrated



Alice's computation illustrated



Man-in-the-middle attack

Alice's client		Eve		Bob's server
p, g, a, A		p, g, e, E		p, g, b, B
	$\xrightarrow{\text{Alice}, A}$		$\xrightarrow{\text{Alice}, E}$	
			$\xleftarrow{\text{Bob}, B}$	
	$\xleftarrow{\text{Bob}, E}$			
$K \leftarrow E^a$		$K \leftarrow A^e$		
		$K' \leftarrow B^e$		$K' \leftarrow E^b$

- Alice and Bob both unknowingly share a secret with Eve
- In subsequent exchange protected with shared secrets
  - Eve decrypts, can read plaintext, and re-encrypts
  - Eve may modify/delete messages and compute tags
- Solution: Alice must verify B belongs to Bob and vice versa

Public key authentication is essential ! !! !!!

Diffie-Hellman key exchange: attention points

- Assume Alice authenticated Bob's public key and vice versa
- Security against eavesdropping Eve
  - Eve needs either a or b to compute $K_{A,B}$
  - given g, A and B, predicting $K_{A,B}$ should be hard
  - This is called (computational) Diffie-Hellman hardness assumption (CDH)
  - CDH seems as hard as discrete log problem but no proof of this
- Domain parameters: both need to work in same cyclic group $\langle g \rangle$
- Entity authentication?
  - can be done with challenge-response using key derived from shared secret
  - along with encryption, message authentication

DH: mutual and unilateral public key authentication

- Mutual PK authentication: both parties authenticate public keys
  - If Alice validated Bob's public key, she knows only Bob has $K_{A,B}$
  - If Bob validated Alice's public key, he knows only Alice has $K_{A,B}$

- Unilateral authentication of the public key
  - Alice authenticates Bob's public key but not vice versa
  - Alice still has guarantee that only Bob knows $K_{A,B}$
  - only Bob can decipher what she enciphers with K
  - only Bob can generate tags with K

- TLS (https) almost always uses unilateral authentication
  - website does not authenticate public key of browser
  - browser generates key pair (a, A) *on the spot*

Note: even if there is public key authentication, DH does not achieve entity or message authentication as such

DH key exchange: forward secrecy

Static Diffie-Hellman: Alice and Bob have long-term key public key pairs

- limitation: $K_{A,B}$ is always the same
- leakage of $K_{A,B}$, a or b allows decryption of all past messages
- this is called: lack of *forward secrecy*

Forward secrecy

is the property that the compromise of keys in a device does not compromise encrypted communication of the past

Consider unilateral case where Bob does not validate Alice's key

- Alice can generate fresh keypair (a, A) for each session/message
- this is called an ephemeral key pair
- leaking $K_{A,B}$ or a only affects single session/message
- leaking b still affects all past cryptograms of Bob

Diffie-Hellman key exchange with forward secrecy

Diffie-Hellman variant with fresh random key pairs for each session

- Alice generates ephemeral key pair (a, A) *on the spot*
- Bob generates ephemeral key pair (b, B) *on the spot*
- They do a Diffie-Hellman with these keys
- Each destroys her/his private key and shared secret after establishment of K
- At the end of the session both destroy K
- This gives forward secrecy across session
- Public key authentication can be done as follows
  - both Alice and Bob have long-term signing keys they authenticate from each other
  - Can be done manually or via a PKI
  - They use these to provide certificate over the ephemeral public keys

ElGamal encryption

ElGamal encryption

- Warning: encryption with public key crypto is risky business
- One of the earliest public key encryption schemes is ElGamal, invented by Taher ElGamal in 1985
- Interesting because often used as building block in cryptographic protocols
- Alice encrypt a message M to cryptogram (C, A) for Bob like this:

Alice		Bob
$p, g, (q), B$		$p, g, (q), b, B(= g^b)$
$a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$A \leftarrow g^a$		
$C \leftarrow M \times B^a$	$\xrightarrow{\text{Alice},(C,A)}$	$M' \leftarrow C \times A^{q-b}$

$$M' = C \times A^{q-b} = M \times B^a \times A^{-b} = M \times (g^b)^a \times (g^a)^{-b} = M \times g^{ba} \times g^{-ab} = M$$

ElGamal encryption: attention points

Alice		Bob
$p, g, (q), B$		$p, g, (q), b, B(= g^b)$
$a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$A \leftarrow g^a$		
$C \leftarrow M \times B^a$	$\xrightarrow{\text{Alice},(C,A)}$	$M \leftarrow C \times A^{q-b}$

- Message M must be an element of $\langle g \rangle$
  - requires encoding function mapping m to $M \in \langle g \rangle$
  - note: must be efficiently decodable for Bob to decrypt
  - existence of such a function depends on the group $\langle g \rangle$
- As first step Alice generates an ephemeral key pair (a, A)
  - for security, a must be randomly generated for each encryption
  - re-use leads to leakage like in one-time pad
- Encryption costs 2 exponentiations, decryption a single one

Security of ElGamal encryption and DDH

Alice		Bob
$p, g, (q), B$		$p, g, (q), b, B (= g^b)$
$a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$A \leftarrow g^a$		
$C \leftarrow M \times B^a$	$\xrightarrow{\text{Alice},(C,A)}$	$M \leftarrow C \times A^{q-b}$

- Encryption works by multiplication with a one-time key B^a
- Secure if this key is indistinguishable from a random element in $\langle g \rangle$
- Leads to *Decisional Diffie Hellman* security notion for a group $\langle g \rangle$
  - with what Eve knows, she cannot distinguish B^a from an element randomly chosen from $\langle g \rangle$
- Don't forget: before you encrypt, verify that B is indeed Bob's public key!

Discrete log crypto security notions

Security notions

Discrete log (DL) hardness assumption

Let $a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$ and $A \leftarrow g^a$.
Given $\langle g \rangle$ and A, the success probability to determine a is negligible.

Computational Diffie-Hellman (CDH) hardness assumption

Let $a, b \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$, $A \leftarrow g^a$ and $B \leftarrow g^b$.
Given $\langle g \rangle$ and A, B, the succes probability to determine g^{ab} is negligible.

Decisional Diffie-Hellman (DDH) hardness assumption

Let $a, b, c \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$, $A \leftarrow g^a$ and $B \leftarrow g^b$.
Given $\langle g \rangle$ and A, B, the advantage of distinguishing g^{ab} and g^c is negligible.

Note: negligible means that the probability/advantage is almost 0 even for an adversary with significant computational resources N and data M

Relations between security notions

$$\text{DDH} \Rightarrow \text{CDH} \Rightarrow \text{DL}$$

- If in $\langle g \rangle$ DDH hardness assumption holds, CDH hardness holds too
  - determining the shared secret allows distinguish it from random
- If in $\langle g \rangle$ CDH holds, DL holds too
  - solving discrete log allows determining the shared secret
- Implications for cryptographic schemes
  - ElGamal encryption is secure if DDH is satisfied
  - Diffie-Hellman is secure if CDH is satisfied
  - Any discrete-log based crypto requires DL to be satisfied
- Security strength: $\log_2(N/\Pr(\text{success}))$ with N attack workload
- Achieved security strength depends on $\langle g \rangle$
  - for s bits of security $\text{ord}(g)$ must be at least 2^{2s}
  - if $\langle g \rangle \subset (\mathbb{Z}/p\mathbb{Z})^*$, it is required that $p \ggg 2^{2s}$
  - groups exist where CDH holds and DDH not, e.g. if $\text{ord}(g)$ is not prime

Conclusions

Conclusions (some more)

- Two very simple discrete-log based cryptosystems:
  - (Merkle)-Diffie-Hellman allows establishing a shared secret
  - ElGamal allows encrypting a message $M \in \langle g \rangle$
- We specified them for $\langle g \rangle \subset (\mathbb{Z}/p\mathbb{Z})^*$ but there are other choices for $\langle g \rangle$
  - "Diffie-Hellman is secure" said formally: CDH holds
  - "ElGamal is secure" said formally: DDH holds
  - Both require that DL assumption holds for $\langle g \rangle$
- Both require parties to authenticate public keys of the other party