ICS0026 Cryptography

Commitments & identification

Taaniel Kraavi

14 November 2024

IT College
Tallinn University of Technology

Remote coin flip ver. 1

How can Alice & Bob remotely decide an outcome via a coin flip?

1. Alice ‘calls’ the result
2. Bob flips the coin
3. Alice wins if she guessed correctly, else Bob wins

How can Alice fix a value without immediately revealing it?

- If Alice cheats, Bob must be able to catch it
- Bob must not learn the value before Alice reveals it

Remote coin flip ver. 2

How can Alice and Bob jointly flip a coin remotely?

- Alice flips her coin and sends her result to Bob
- Bob flips his coin and sends his result to Alice
- Alice & Bob XOR their results to obtain the same value

What additional challenges do you foresee?

Commitment schemes

A *commitment* fixes a value but allows it to be revealed later.

- A committer and a receiver
- The committer *commits* their value
- The committer *opens* (reveals/decommits) the value later (not always)

Two essential properties of commitment schemes:

- *Hiding*: the commitment alone does not reveal the committed value
- *Binding*: the committer cannot open the commitment to a different value

Quantifying the properties

Perfect security

- There is a zero chance that an unbounded adversary can break the property

Statistical security

- There is a negligible chance that an unbounded adversary can break the property

Computational security

- There is a negligible chance that a computationally bounded adversary can break the property

Commitments cannot be both perfectly hiding and perfectly binding.

Cryptosystems as commitment schemes

Any functional IND-CPA cryptosystem is hiding and perfectly binding.

- Intuitively, why is it perfectly binding?
- Why is it hiding?
- Why is it not perfectly hiding?

E.g. ElGamal encryption can be used as a commitment scheme.

- ElGamal commitments may refer to both classic and lifted ElGamal
- We can create public ElGamal parameters without knowing the secret key
- If the secret key is known, the scheme becomes *extractable*

Two additional properties

Extractability

- Knowing a secret value, the message can be extracted from the commitment
- Public parameters are indistinguishable in both settings

Equivocability

- ‘Fake’ commitments are indistinguishable from real commitments
- The fake commitments can be opened to arbitrary values

Some commitments therefore require a trusted setup.

Pedersen commitments

Let q be prime and let $\mathbb{G} = \langle g \rangle$ be a q-element DL-group.
Choose h uniformly from $\mathbb{G} \setminus \{1\}$ and set pk $\leftarrow (g, h)$.

To commit $m \in \mathbb{Z}_q$, choose $r \xleftarrow{u} \mathbb{Z}_q$ and output

$$\begin{cases} c \leftarrow g^m h^r \ , \\ d \leftarrow (m, r) \ . \end{cases}$$

A tuple (c, m, r) is a valid decommitment for m if $c = g^m h^r$.

Pedersen commitments

Pedersen commitments are

- Perfectly hiding
  - e.g. in $\mathbb{Z}_{23}$: $2^2 3^7 \equiv 8$, $2^3 3^0 \equiv 8$, $2^4 3^4 \equiv 8$, ...
- Computationally binding
  - Let $g^x \equiv h$ with x unknown.
  - It is then hard to find distinct pairs $(m_1, r_1), (m_2, r_2)$ s.t. $g^{m_1} h^{r_1} \equiv g^{m_2} h^{r_2}$.
- Equivocable
  - If x is known, breaking the binding property is trivial.
  - Generate $s \xleftarrow{u} \mathbb{Z}_q$, set $\hat{c} \leftarrow g^s$.
  - To open $\hat{c}$, compute $r \leftarrow (s - m) \cdot x^{-1}$.

Homomorphic properties

Both Pedersen and ElGamal commitments are homomorphic.

- Only lifted ElGamal commitments are ‘interoperable’ with Pedersen’s.
- Desirable property for their use in many protocols.
- Not always desirable!

Why do we need commitments?

Commitments are an essential primitive in cryptography.

- Typically they are not used on their own (although they can).
- It is possible to prove relations over committed values.
- Commitments are essential to many zero-knowledge protocols.
- More generally, many secure protocols rely on commitments.

How would you use commitments in the remote coin flipping protocol?

Identification protocols

Identification protocols are generally based on one or more of:

- What you are (biometrics)
- What you have
- What you know

Cryptography works with the latter two.

- What you know: some secret
- What you have: cryptographic hardware device (last week's topics)

Identification requires *freshness*, but does not require *non-repudiation*.

- Security requirement: preventing *impersonation* attacks

Security assumptions

We must assume a trusted registration protocol

- Digital identification is not possible without it
- No legal identity without physical measures

We must prevent man-in-the-middle attacks

- e.g. chessmaster attack

We only consider cryptographic attacks

- Cryptography cannot defend against ‘non-math’ attacks
- Physical attacks against smart cards, credential exfiltration, ...

Challenge-response paradigm

A prover must answer a verifier's challenge for identification.

Familiar approaches:

- Symmetric encryption
  - e: KPA-SEC, v: IND-CPA (adaptive)
- Symmetric authentication (HMAC)
  - e, v: ROM
- Asymmetric encryption
  - e: IND-CPA, v: IND-CCA2
- Asymmetric authentication
  - e: KMA-SEC, v: CMA-SEC

What about password-based authentication?

- Does not work over a cleartext channel.

Zero-knowledge identification

The asymmetric approaches presented previously can be quite ‘costly’ in practice.

How can we prevent cheating verifier attacks with an ‘efficient’ scheme?

- A cheating verifier must not get useful information from the prover.
- The protocol must be *zero-knowledge*.
- The protocol must convince an honest verifier that an (honest) prover knows the secret.

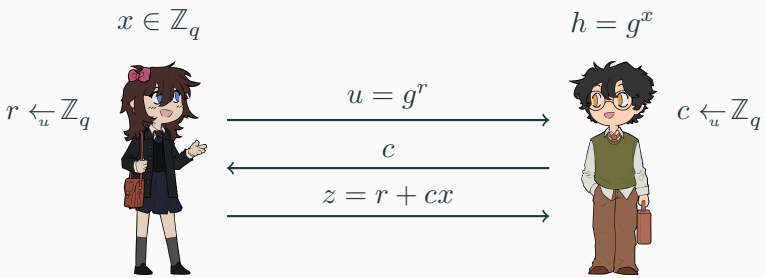
Simulation

Zero-knowledge is based on the principle of *simulation*.

- Messages sent by the prover can be efficiently simulated.
- These messages convince the verifier that the prover knows the secret.
- I.e. an accepting transcript can be generated without involving the prover.

Schnorr's protocol

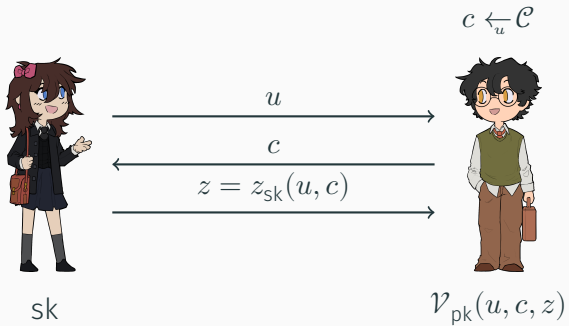
Schnorr's protocol allows proving knowledge of a discrete logarithm without leaking it.



The group $\mathbb{G} = \langle g \rangle$ must be a DL group with a prime cardinality q.

Sigma protocols

Σ-protocols are efficiently computable three-move protocols that must be (special) honest verifier zero-knowledge.



In the *transcript* (u, c, v), u is a commitment, c is a (varying) challenge, and z is a (consistent) response.

Danger zone!

Sigma protocols & zero-knowledge schemes are advanced cryptography.

- You should know that they exist
- You should not use them directly
- The complexity snowballs
- Not comparable to popular and standardised schemes
  - encryption & signature schemes, KEMs, MACs, hash functions

Danger zone!

These topics are a prime example of unknown unknowns.

- Reading a few blog posts to get an intuition to start using them is not enough
- Extensive amounts of study required for even the ‘entry level’
- Not gatekeeping: you are more than welcome to spiral down the rabbit hole