07. Zero-Knowledge Proofs

이형태

2019학년도 2학기

Proof Systems

A Proof System

An argument between a prover (denoted $\mathcal{P}$) and a verifier (denoted $\mathcal{V}$) in order that $\mathcal{P}$ convinces $\mathcal{V}$ of a language L

Classical Proofs ≈ Written Exam

- $\mathcal{P}$ writes down all that it has to say.
- $\mathcal{V}$ checks this statement.

⇒ There are no interactions between $\mathcal{P}$ and $\mathcal{V}$.

- However, there are lots of real world applications where $\mathcal{P}$ **cannot prove to** $\mathcal{V}$ via a classical proof.

  Examples:
  - How to prove that two graphs are isomorphic?
  - How to prove that I am what I am?

  ⋮

Interactive Proof Systems in Cryptography

An Interactive Poof System in Cryptography

- $\mathcal{P}$ can have **infinite** running time like a powerful wizard.
- $\mathcal{V}$ has **polynomial** running time and has the ability to generate **random numbers**–probabilistic polynomial time (PPT), like a person.
- After interacting for a polynomial time, $\mathcal{V}$ accepts or rejects the proof by $\mathcal{P}$.

- Note: In a different setting, the powers of $\mathcal{P}$ and $\mathcal{V}$ may be set differently.

Σ-Protocol for Relation R

- A binary relation for some problem $R : \{0,1\}^* \times \{0,1\}^*$ (e.g., $R = \{(v,w) | w = \text{SHA-256}(v)\}$)
- In this case, we call w a witness.

Σ-Protocol for Relation $R = \{(v,w)\} \subseteq V \times W$

		$\mathcal{P}$		$\mathcal{V}$
		$(v,w) \in R$		$v \in V$
$a = \alpha(v, w, u_{\mathcal{P}})$			$\xrightarrow{\text{announcement } a}$	
			$\xleftarrow{\text{challenge } c}$	$c \overset{\$}{\leftarrow} C$
$r = \rho(v, w, c, u_{\mathcal{P}})$			$\xrightarrow{\text{response } r}$	
				$\varphi(v, a, c, r)?$

Zero-Knowledge Proofs

A zero-knowledge proof is a protocol between $\mathcal{P}$ and $\mathcal{V}$ that satisfies the following three properties:

(Informally speaking,)

- **Completeness**: If $\mathcal{P}$ and $\mathcal{V}$ follow the protocol, then $\mathcal{V}$ always accepts.
- **Soundness**: If the statement is false, no cheating prover $\mathcal{P}$ can convince the honest verifier $\mathcal{V}$ that it is true, except with negligible probability.
- **Zero-Knowledgeness**: If the statement is true, no verifier $\mathcal{V}$ learns anything other than the fact that the statement is true.

Schnorr's Protocol for Proving Knowledge of DL

- Let $\mathbb{G} = \langle g \rangle \subset \mathbb{Z}_p^*$ be a cyclic group of order q.
- Goal: Given a public value $y = g^x$, the prover should convince the verifier that the prover knows x.

Description of Schnorr's Protocol

$\mathcal{P}$		$\mathcal{V}$
1. Choose $\alpha \xleftarrow{\$} \mathbb{Z}_q$		
2. Compute $\beta = g^\alpha \mod p$	$\xrightarrow{\quad \beta \quad}$	
	$\xleftarrow{\quad c \quad}$	1. Choose $c \xleftarrow{\$} \mathbb{Z}_q$
3. Compute $s = xc + \alpha \mod q$	$\xrightarrow{\quad s \quad}$	
		2. Check if $g^s \stackrel{?}{=} y^c \beta$

Schnorr's Protocol for Proving Knowledge of DL (Cont.)

- **Completeness**: $g^s = g^{xc+\alpha} = (g^x)^c g^\alpha = y^c \beta$
- **Soundness**: If the cheating prover can generate two valid pairs (β, c, s) and (β, c', s'), then

$$\begin{aligned} g^s = y^c \beta \text{ and } g^{s'} = y^{c'} \beta \quad &\Rightarrow \quad g^{s-s'} = y^{c'-c} \\ &\Leftrightarrow \quad y = g^{(s-s')/(c'-c)} \end{aligned}$$

  Thus, he actually knows the witness $x = \dfrac{s - s'}{c - c'}$.

- **Zero-Knowledgeness**: The distributions of the following two sets are indistinguishable.

$$\begin{aligned} &\{(\beta, c, s) | \alpha \in_R \mathbb{Z}_q, \beta = g^\alpha, s = \alpha + cx \bmod q\} \ \text{ and} \\ &\{(\beta, c, s) | r \in_R \mathbb{Z}_q, \beta = g^r y^{-c}\} \end{aligned}$$

Making a Σ-Protocol Non-Interactive

- Replace a challenge c by an output of cryptographic hash function H
- Secure under the random oracle model (Assume that H is a random oracle.)

Transformation into Non-Interactive Protocol

$\mathcal{P}$		$\mathcal{V}$
$(v, w) \in R$		$v \in V$
1. $a = \alpha(v, w, u_{\mathcal{P}})$ 2. $c = H(a, v)$ 3. $r = \rho(v, w, c, u_{\mathcal{P}})$	$\xrightarrow{(a, c, r)}$	
		1. Compute $c = H(a, v)$ 2. $\varphi(v, a, c, r)$?

Non-Interactive Version of Schnorr's Protocol

- Let $\mathbb{G} = \langle g \rangle \subset \mathbb{Z}_p^*$ be a cyclic group of order q.
- H: a cryptographic hash function
- Goal: Given a public value $y = g^x$, the prover should convince the verifier that the prover knows x.

Non-Interactive Version of Schnorr's Protocol

$\mathcal{P}$		$\mathcal{V}$
1. Choose $\alpha \xleftarrow{\$} \mathbb{Z}_q$		
2. Compute $\beta = g^\alpha \mod p$		
3. Compute $c = H(\beta, y)$		
4. Compute $s = xc + \alpha \mod q$	$\xrightarrow{(\beta, c, s)}$	
		1. Compute $\beta' = g^s y^{-c}$
		2. Check if $c = H(\beta', y)$

$$(\because \beta' = g^s y^{-c} = g^{\alpha + xc}(g^x)^{-c} = g^{\alpha + xc - xc} = g^\alpha = \beta)$$

Signature from a Non-Interactive Zero-Knowledge Protocol

- Add a message M into an input of H once the prover computes a challenge
- v: public key, w: secret key

Transformation into Non-Interactive Protocol

$\mathcal{P}$		$\mathcal{V}$
$(v, w) \in R$		$v \in V$
Sign		
1. $a = \alpha(v, w, u_{\mathcal{P}})$		
2. $c = H(M, a, v)$		
3. $r = \rho(v, w, c, u_{\mathcal{P}})$	$\xrightarrow{\text{signature } \sigma = (a, c, r)}$	**Verify**
		1. Compute $c = H(M, a, v)$
		2. $\varphi(v, a, c, r)$?

Schnorr Signature

Key Generation

1. Generate a cyclic subgroup $\mathbb{G}$ with order q of $\mathbb{Z}_p^*$.
2. Select a random generator g of $\mathbb{G}$.
3. Choose a random integer x from $\mathbb{Z}_q^*$ and compute $y = g^x$ in $\mathbb{G}$.
4. Output a public key $pk = (p, q, g, y)$ and a secret key x.

Sign

Given the secret key $sk = x$ and a message M,

1. Choose a random integer α from $\mathbb{Z}_q^*$.
2. Compute $\beta = g^\alpha \bmod p$.
3. Compute $c = H(M, \beta, y)$.
4. Compute $s = \alpha + xc \bmod q$
5. Output $\sigma = (M, (c, s))$.

Schnorr Signature (Cont.)

Verify

Given the public key $pk = (p, q, g, y)$ and a signature $\sigma = (M, (c, s))$,

1. Compute $\beta' = g^s y^{-c}$
2. Check if $c \stackrel{?}{=} H(M, \beta', y)$. If it holds, return 1. Otherwise, return 0.

Correctness

$$\because \beta' = g^s y^{-c} = g^{\alpha+xc}(g^x)^{-c} = g^{\alpha+xc-xc} = g^\alpha = \beta$$

References

Sch18 B. Schoenmakers, Lecture Notes - Cryptographic Protocols, Chapter 4 & 5, Version 1.32, Feb 2018.