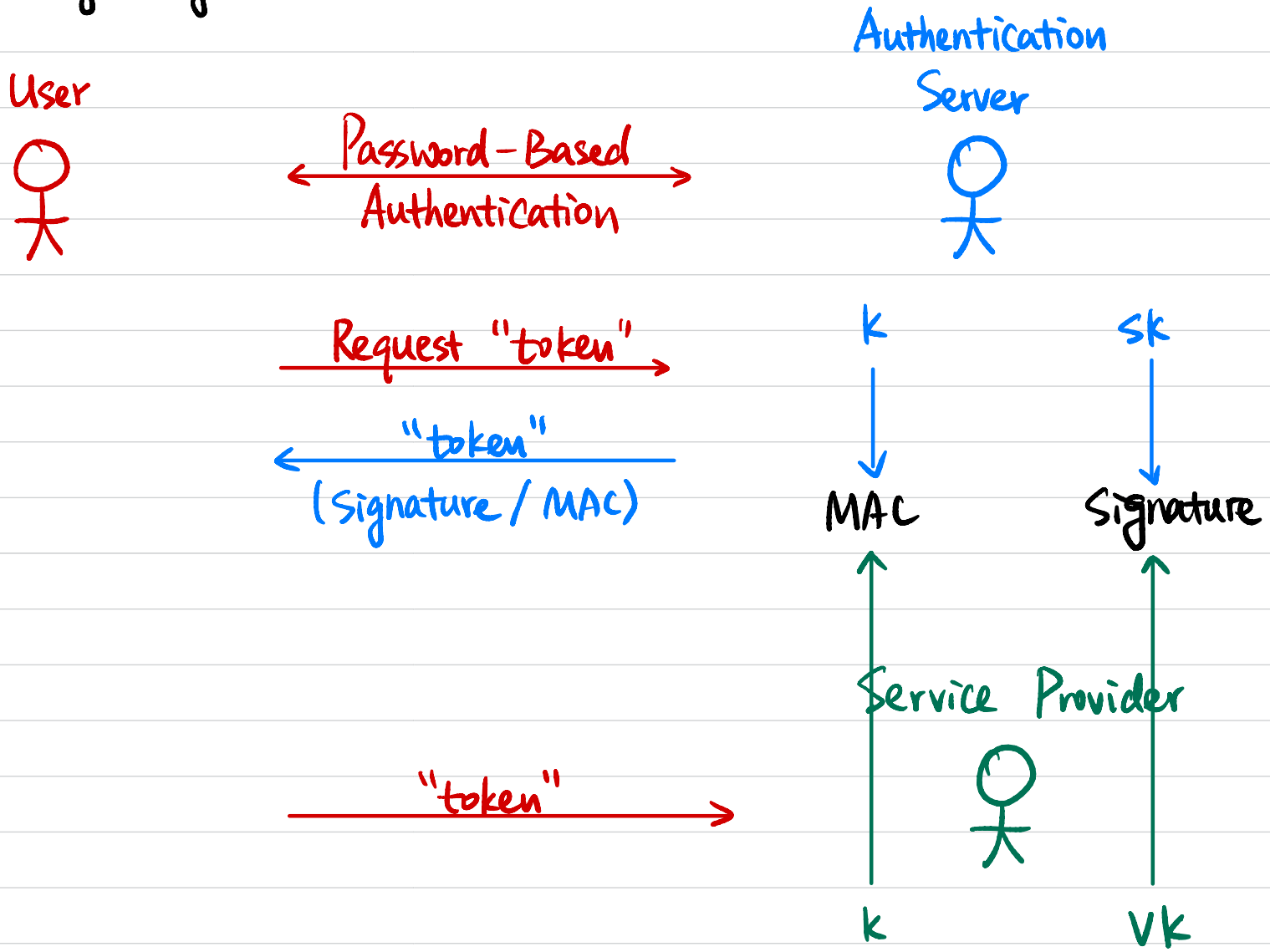
CSCI 1515 Applied Cryptography

This Lecture:

- Case Study: Single Sign-On (SSO) Authentication
- Definition of Zero-Knowledge Proofs
- Example: Schnorr's Identification Protocol
- Example: Diffie-Hellman Tuple

Case Study: Single Sign-On (SSO) Authentication



- OAuth / OpenID: Sign-in with Google / Apple / Brown / ...
- Kerberos: enterprises

Zero-Knowledge Proofs

Alice

Bob

[There is a bug in your code]

[I have the secret key for this ciphertext]

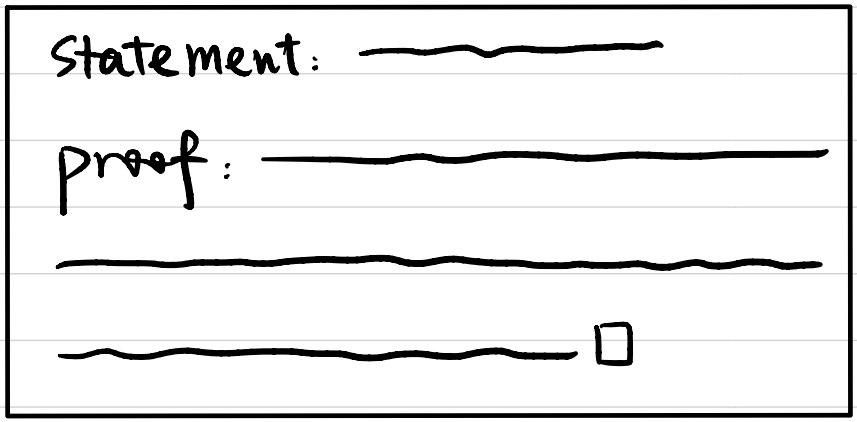
[There is enough balance in my Bitcoin account]

[ have different colors]

What is a proof?

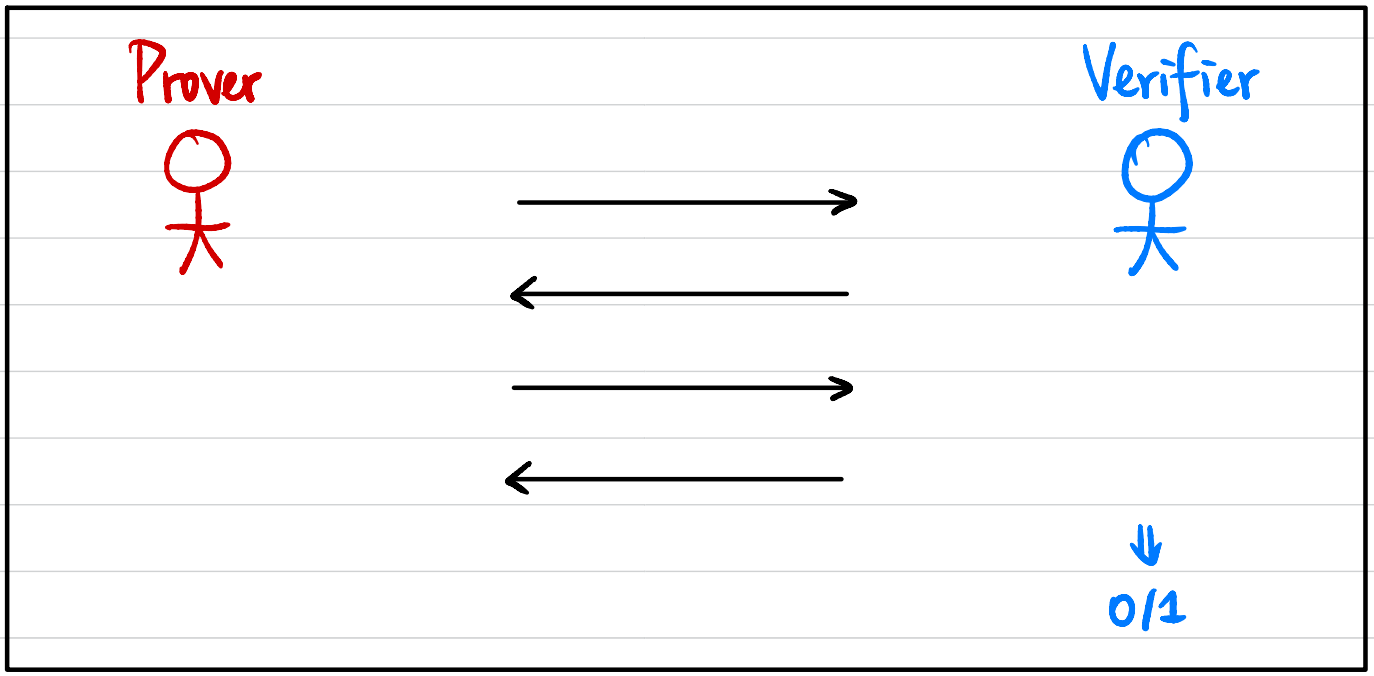
What does zero-knowledge mean?

What is a "proof system"?



- **Completeness:** If statement is true, then ∃ proof that proves it's true.
- **Soundness:** If statement is false, then ∄ proof that proves it's true.

Interactive Proof



- **Completeness:** If statement is true,

  $\exists P$ s.t. $\Pr[P \longleftrightarrow V \text{ outputs } 1] = 1$.

- **Soundness:** If statement is false,

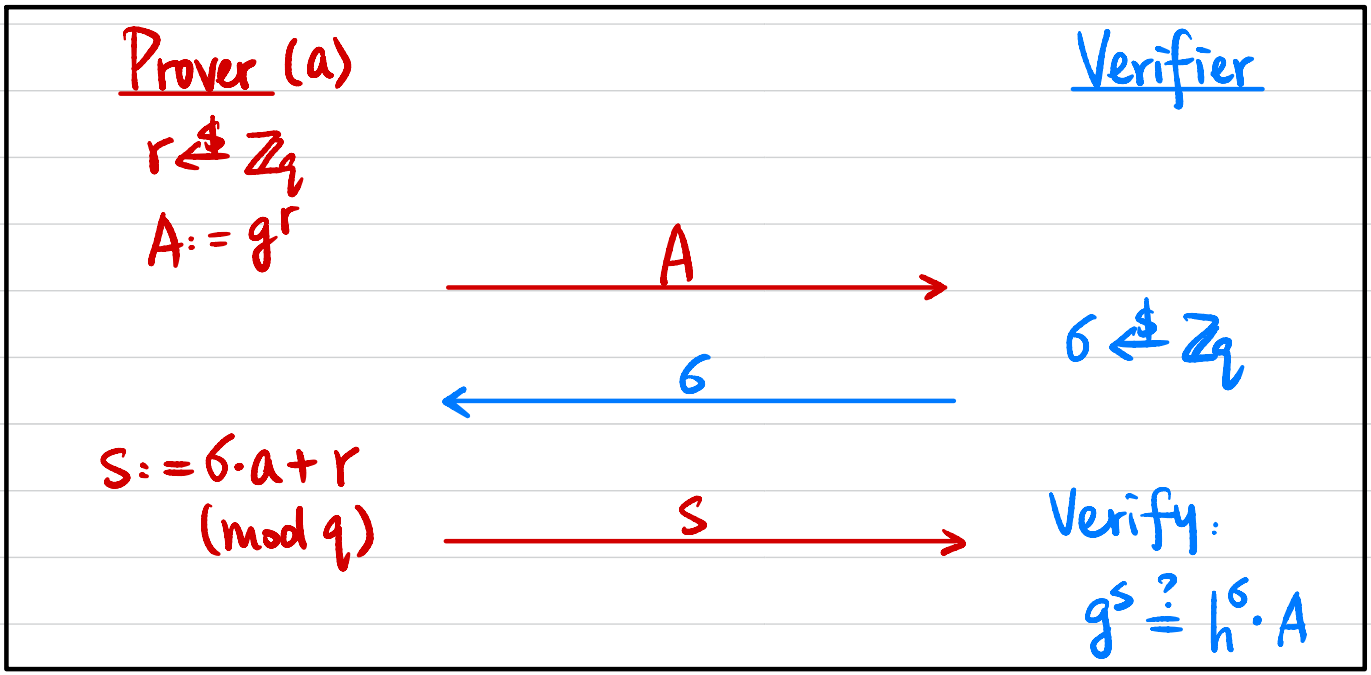
  $\forall P^*$, $\Pr[P^* \longleftrightarrow V \text{ outputs } 1] \simeq 0$.

- **Zero-Knowledge?**

Example: Schnorr's Identification Protocol

Public: Cyclic group G of order q, generator g, $h = g^a$

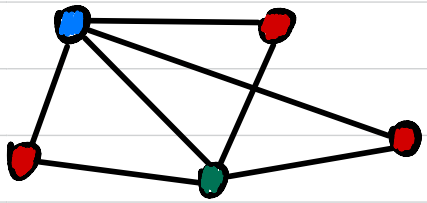
Prover's secret: a



- **Completeness?** If P knows a?
- **Soundness?** If P doesn't know a?
- **Zero-Knowledge?** What does V learn?

NP as a Proof System

Example: Gragh 3-Coloring



NP language $L = \{ G : G \text{ has 3-coloring} \}$

NP relation $R_L = \{ (G, 3COL) \}$

(public) statement

(secret) witness

Example: DLOG: Cyclic group G of order q, generator g

NP language $L = \{ h : h \in G \}$

NP relation $R_L = \{ (h, a) : h = g^a \}$

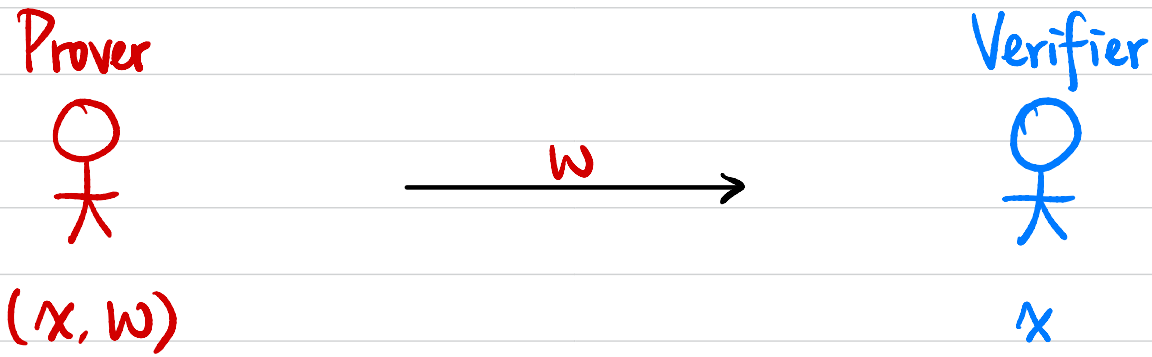
(public) statement

(secret) witness

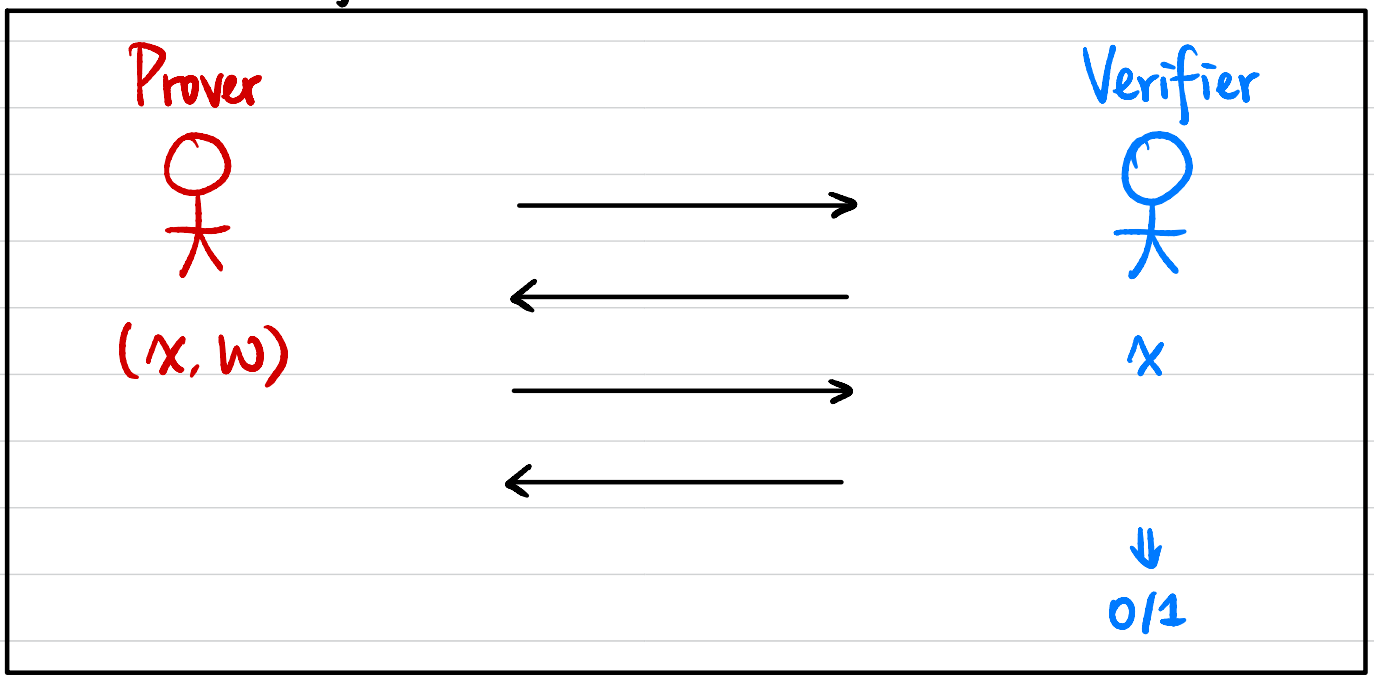
NP as a Proof System

A language L is in NP if $\exists$ poly-time V s.t.

- **Completeness:** $\forall x \in L$, $\exists w$ s.t. $V(x, w) = 1$
  (w: witness)
- **Soundness:** $\forall x \notin L$, $\forall w^*$, $V(x, w^*) = 0$



Zero-Knowledge Proof (ZKP)



Let (P, V) be a pair of probabilistic poly-time (PPT) interactive machines.

(P, V) is a **zero-knowledge proof system** for a language L with associated relation R_L if

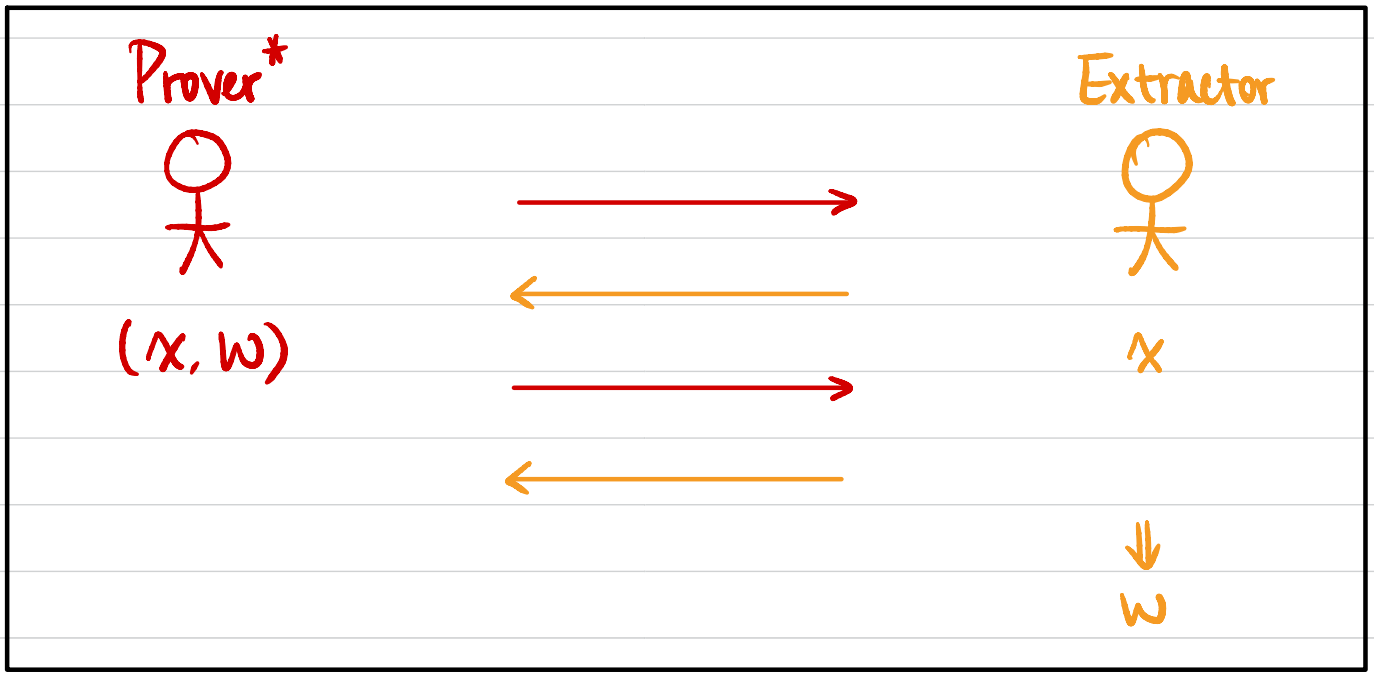
- **Completeness:** $\forall (x, w) \in R_L$, $\Pr[P(x, w) \longleftrightarrow V(x) \text{ outputs } 1] = 1$.

  $\forall (x, w) \in R_L$, P can prove it.

- **Soundness:** $\forall x \notin L$, $\forall P^*$, $\Pr[P^*(x) \longleftrightarrow V(x) \text{ outputs } 1] \simeq 0$.

  $\forall x \notin L$, any P^* cannot prove it.

Proof of Knowledge (PoK)

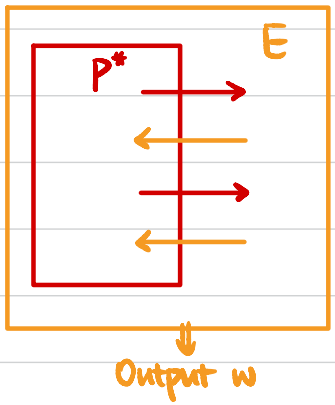


- **Proof of Knowledge:**

  $\exists$ PPT E s.t. $\forall P^*$, $\forall x$,

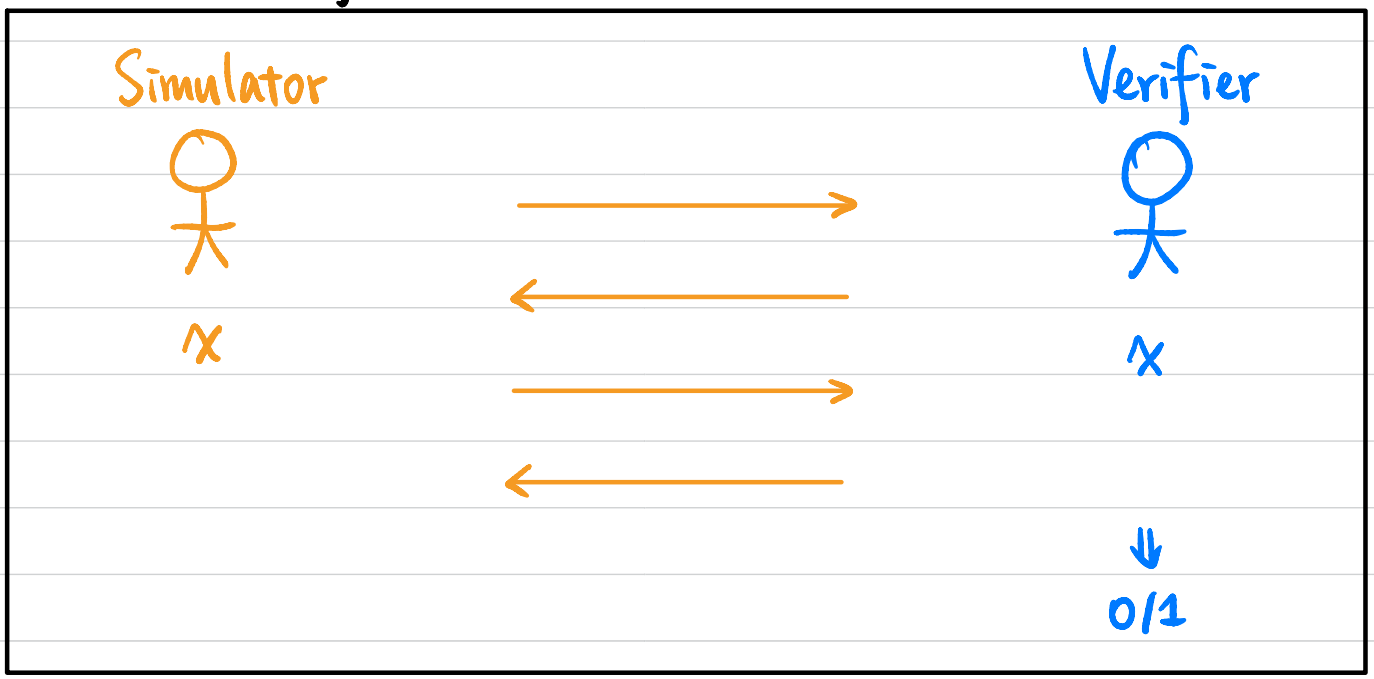
  $$\Pr\left[E^{P^*(\cdot)}(x) \text{ outputs } w \text{ s.t. } (x,w) \in R_L\right] \simeq \Pr\left[P^* \leftrightarrow V(x) \text{ outputs } 1\right].$$

  If P^* can prove it, P^* must know w.



How is it possible?

Zero-Knowledge Proof (ZKP)



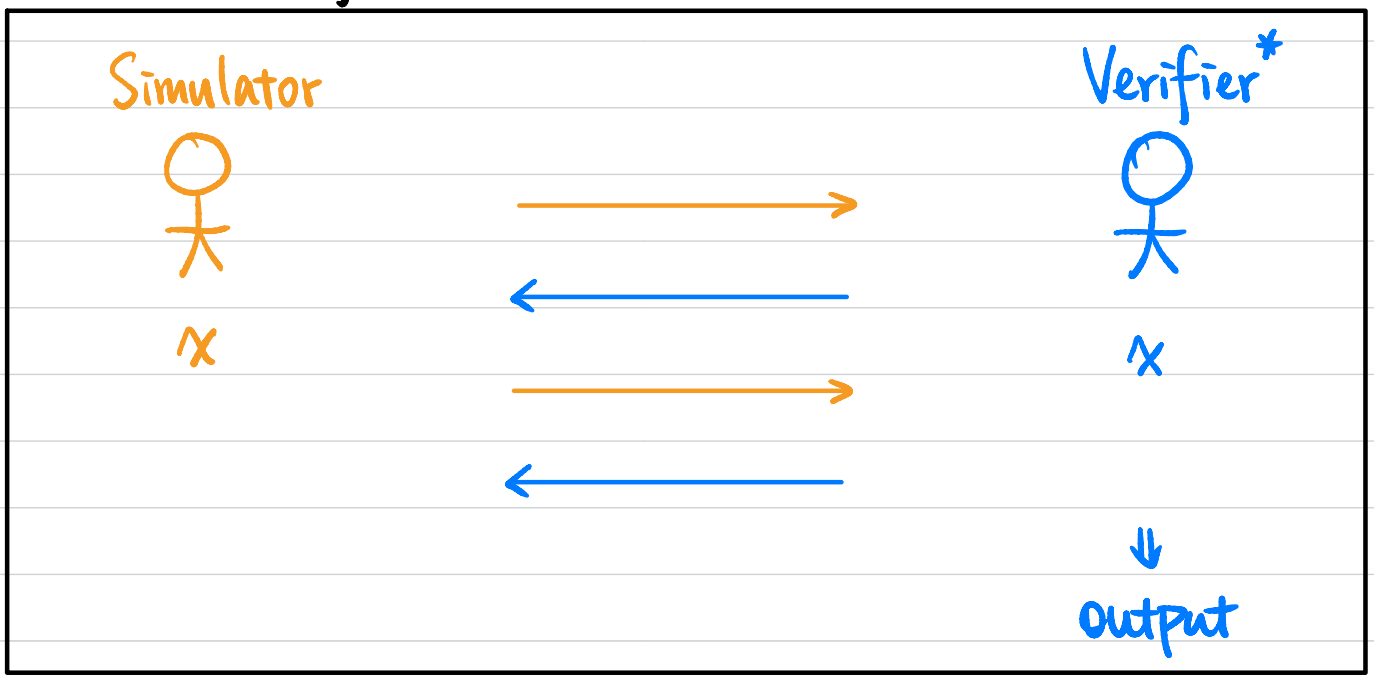
- **Honest-Verifier Zero-Knowledge (HVZK):**

$$\exists \text{PPT } S \quad \text{s.t.} \quad \forall (x, w) \in R_L,$$

$$\text{View}_V[P(x,w) \longleftrightarrow V(x)] \simeq S(x)$$

An honest V doesn't learn anything about w.

Zero-Knowledge Proof (ZKP)

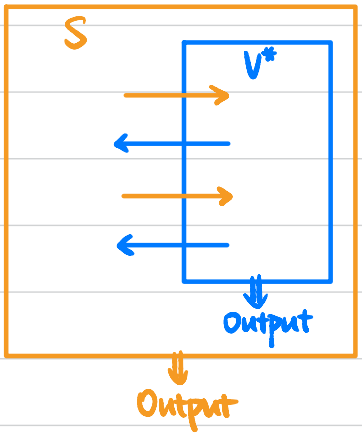


- **Zero-Knowledge** (Malicious Verifier):

$$\forall \text{PPT } V^*, \ \exists \text{PPT } S \quad \text{s.t.} \quad \forall (x, w) \in R_L,$$

$$\text{Output}_{V^*}[P(x,w) \longleftrightarrow V^*(x)] \simeq S(x)$$

A malicious V^* doesn't learn anything about w.



How is it possible?

Zero-Knowledge Proof of Knowledge

- **Completeness:** $\forall (x,w) \in R_L, \quad \Pr[P(x,w) \longleftrightarrow V(x) \text{ outputs } 1] = 1.$

  $\forall (x,w) \in R_L$, P can prove it.

- **Soundness.** $\forall x \notin L, \ \forall P^*, \quad \Pr[P^*(x) \longleftrightarrow V(x) \text{ outputs } 1] \simeq 0.$

  $\forall x \notin L$, any P^* cannot prove it.

- **Proof of Knowledge:** $\exists$ PPT E s.t. $\forall P^*, \forall x,$

  $$\Pr[E^{P^*(\cdot)}(x) \text{ outputs } w \text{ s.t. } (x,w) \in R_L] \simeq \Pr[P^* \leftrightarrow V(x) \text{ outputs } 1].$$

  If P^* can prove it, P^* must know w.

- **Honest-Verifier Zero-Knowledge (HVZK):** $\exists$ PPT S s.t. $\forall (x,w) \in R_L,$

  $$\text{View}_V[P(x,w) \longleftrightarrow V(x)] \simeq S(x)$$

  An honest V doesn't learn anything about w.

- **Zero-Knowledge:** $\forall$ PPT V^*, $\exists$ PPT S s.t. $\forall (x,w) \in R_L,$

  $$\text{Output}_{V^*}[P(x,w) \longleftrightarrow V^*(x)] \simeq S(x)$$

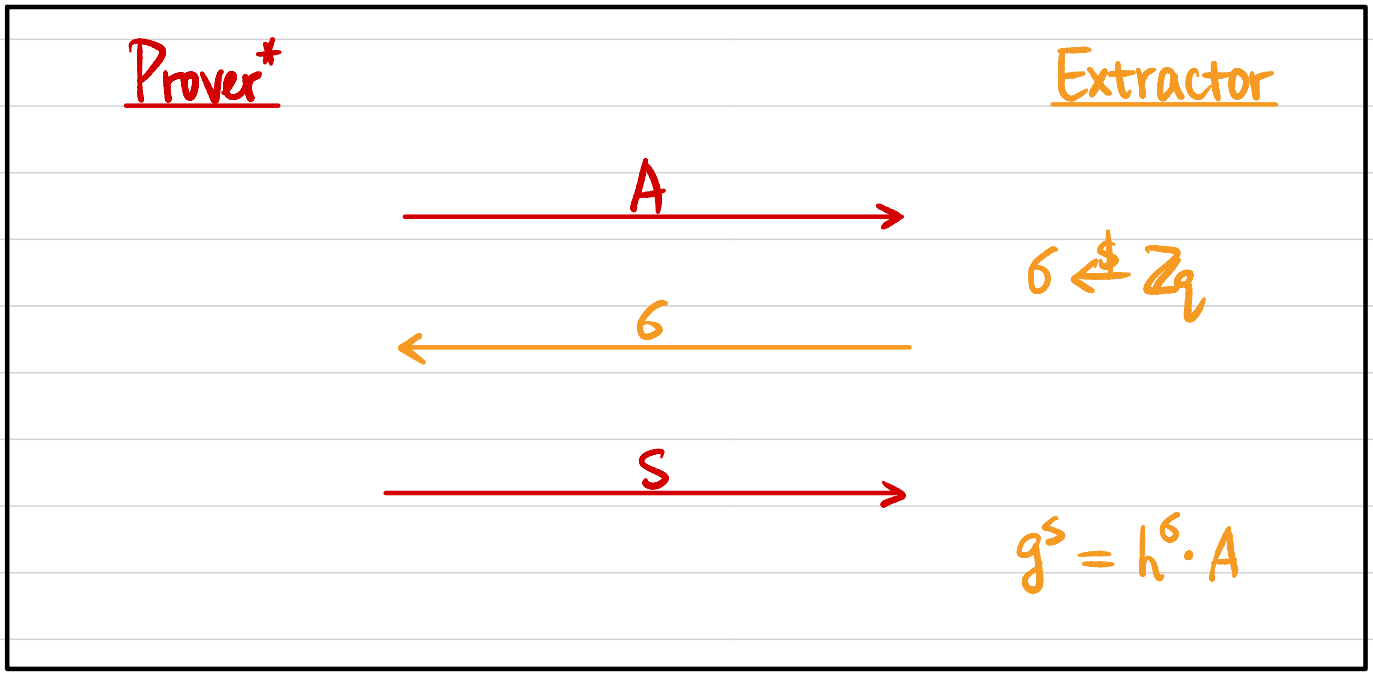
  A malicious V^* doesn't learn anything about w.

Example: Schnorr's Identification Protocol

Proof of Knowledge?

$\exists$ PPT E s.t. $\forall P^*$, $\forall x$,

$$\Pr\left[E^{P^*(\cdot)}(x) \text{ outputs } w \text{ s.t. } (x,w) \in R_L\right] \simeq \Pr\left[P^* \leftrightarrow V(x) \text{ outputs } 1\right].$$



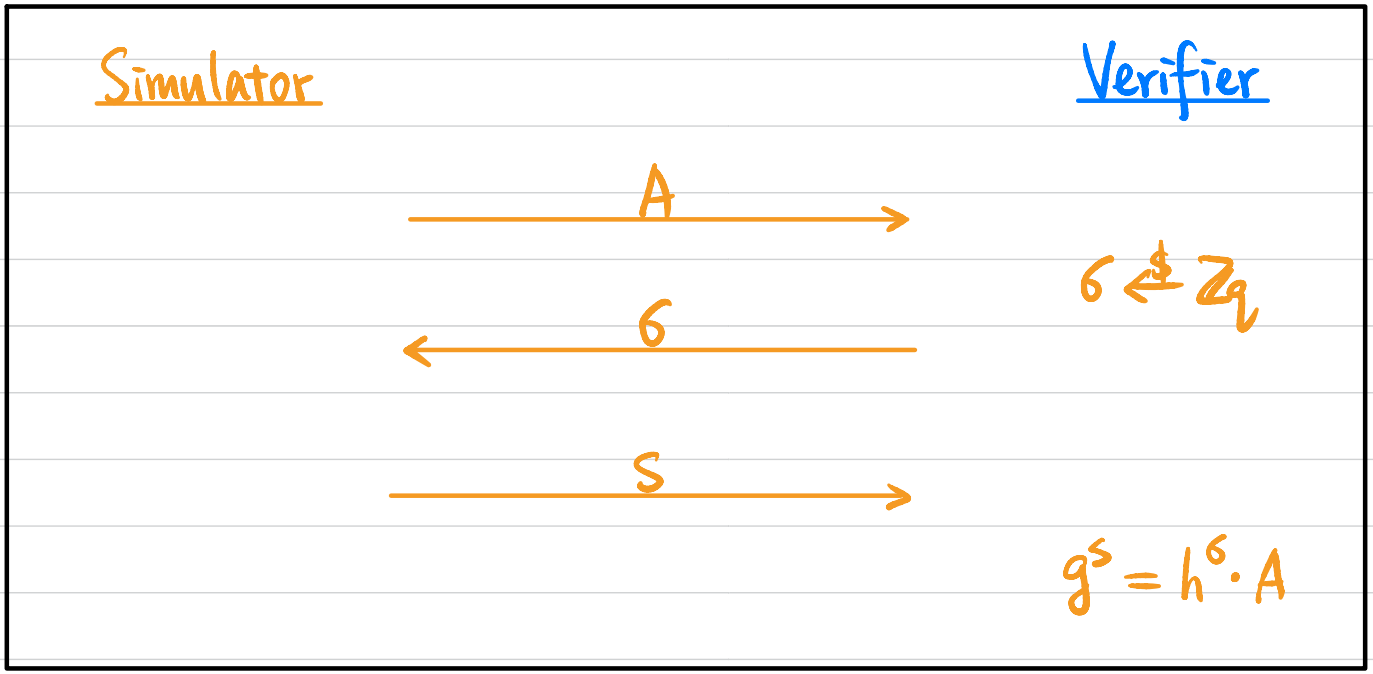
How to extract a s.t. $h = g^a$?

Example: Schnorr's Identification Protocol

Honest-Verifier Zero-Knowledge (HVZK)?

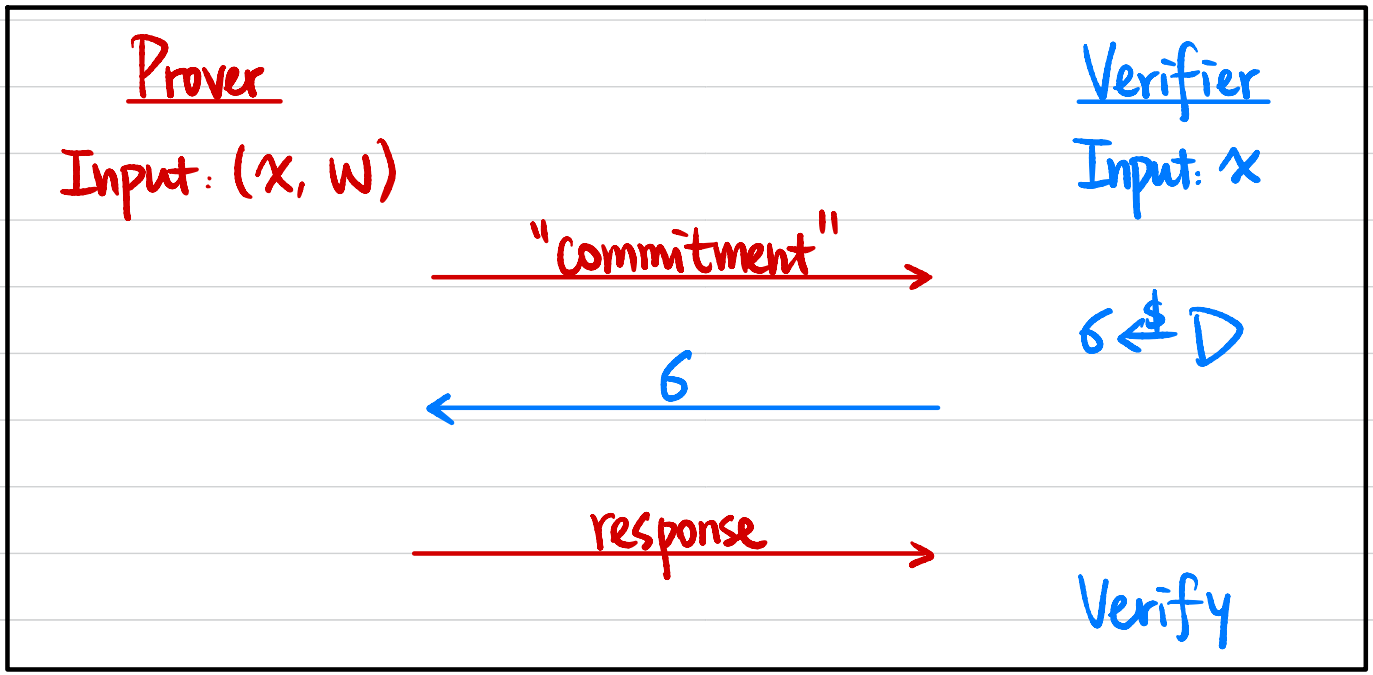
$\exists$ PPT S s.t. $\forall (x, w) \in R_L$,

$$\text{View}_V[P(x,w) \longleftrightarrow V(x)] \simeq S(x)$$



How to generate (A, s) s.t. $g^s = h^c \cdot A$?

Sigma Protocols Σ

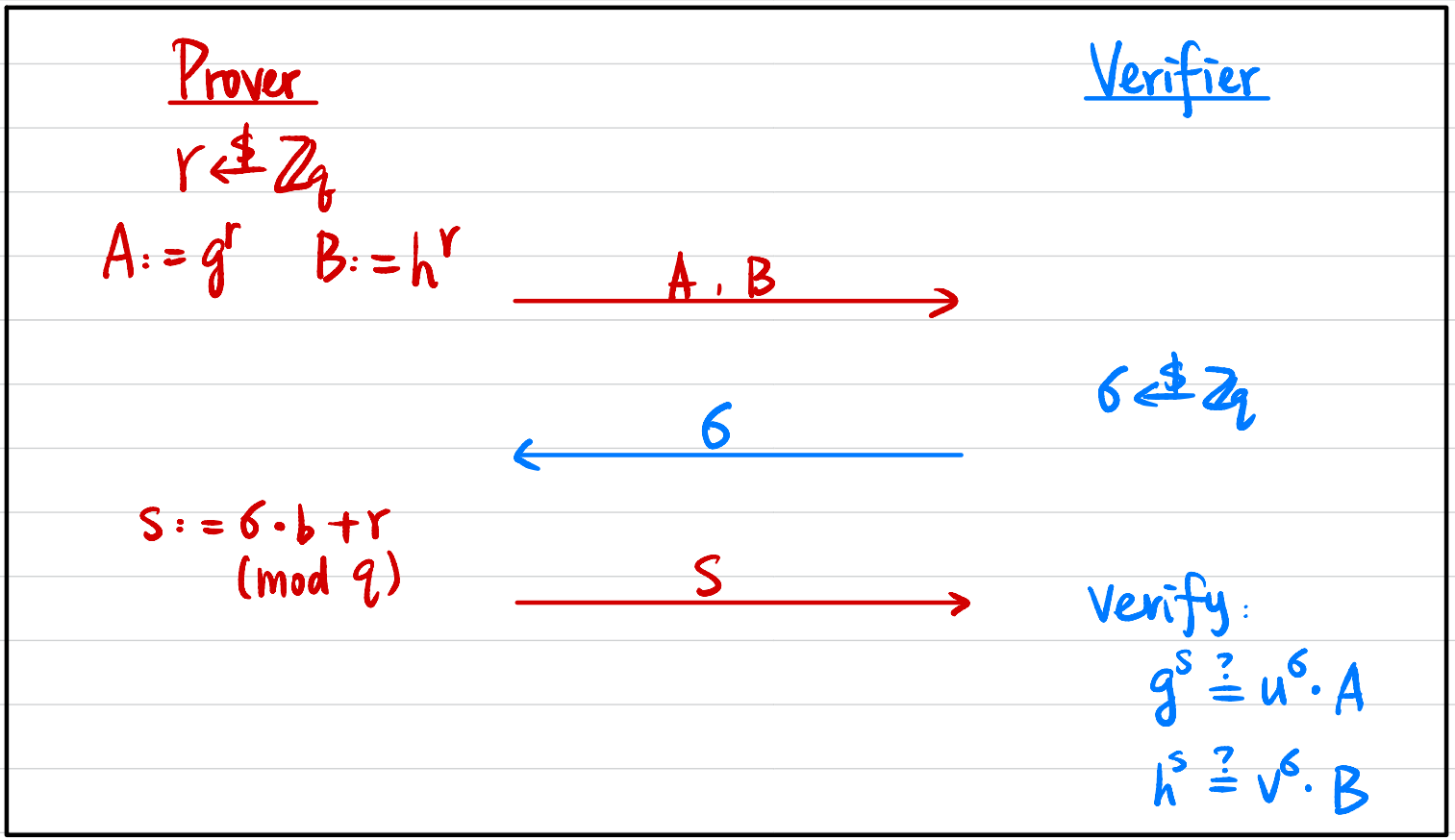


Example: Diffie-Hellman Tuple

Public: Cyclic group G of order q, generator g, $(h,u,v) = (g^a, g^b, g^{ab})$

Prover's secret witness: b s.t. $u = g^b \wedge v = h^b$

$R_L = \{ ((h,u,v), b) \}$



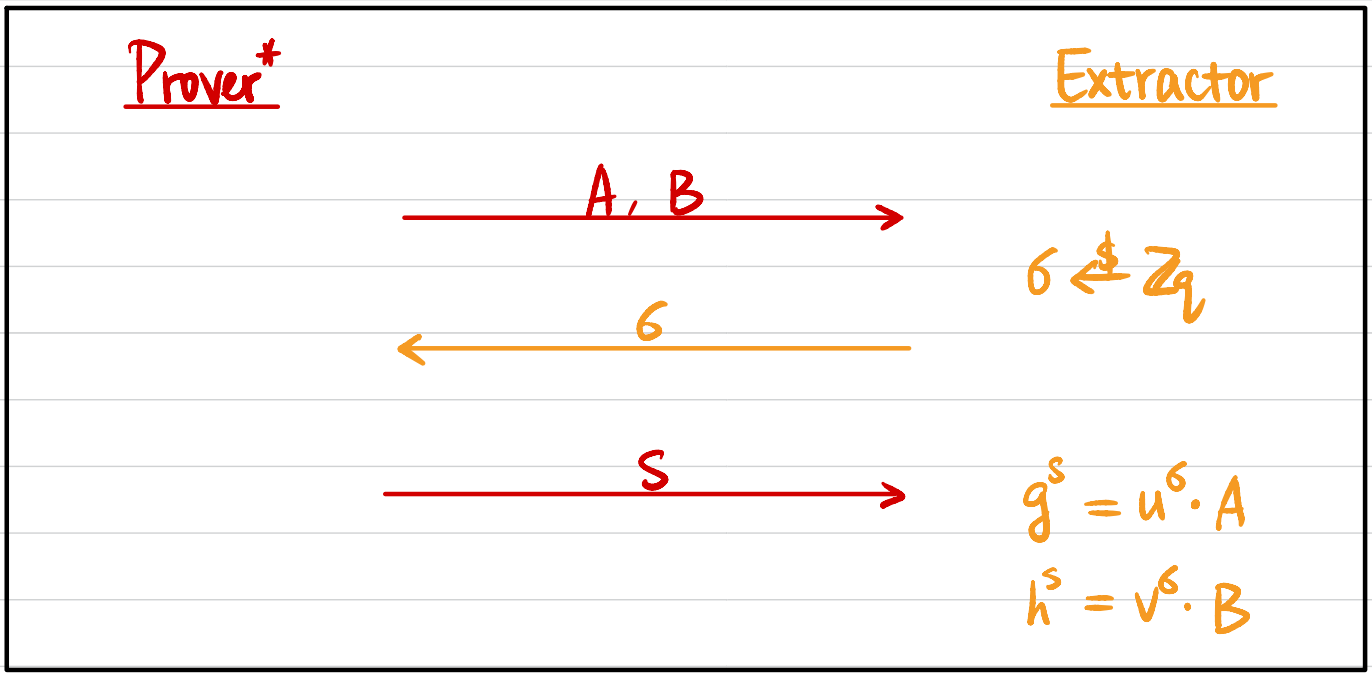
Completeness? $\forall (x,w) \in R_L \quad \Pr[P(x,w) \longleftrightarrow V(x) \text{ outputs } 1] = 1.$

Example: Diffie-Hellman Tuple

Proof of Knowledge?

$\exists$ PPT E s.t. $\forall P^*$, $\forall x$,

$$\Pr\left[E^{P^*(\cdot)}(x) \text{ outputs } w \text{ s.t. } (x,w) \in R_L\right] \simeq \Pr\left[P^* \leftrightarrow V(x) \text{ outputs } 1\right].$$



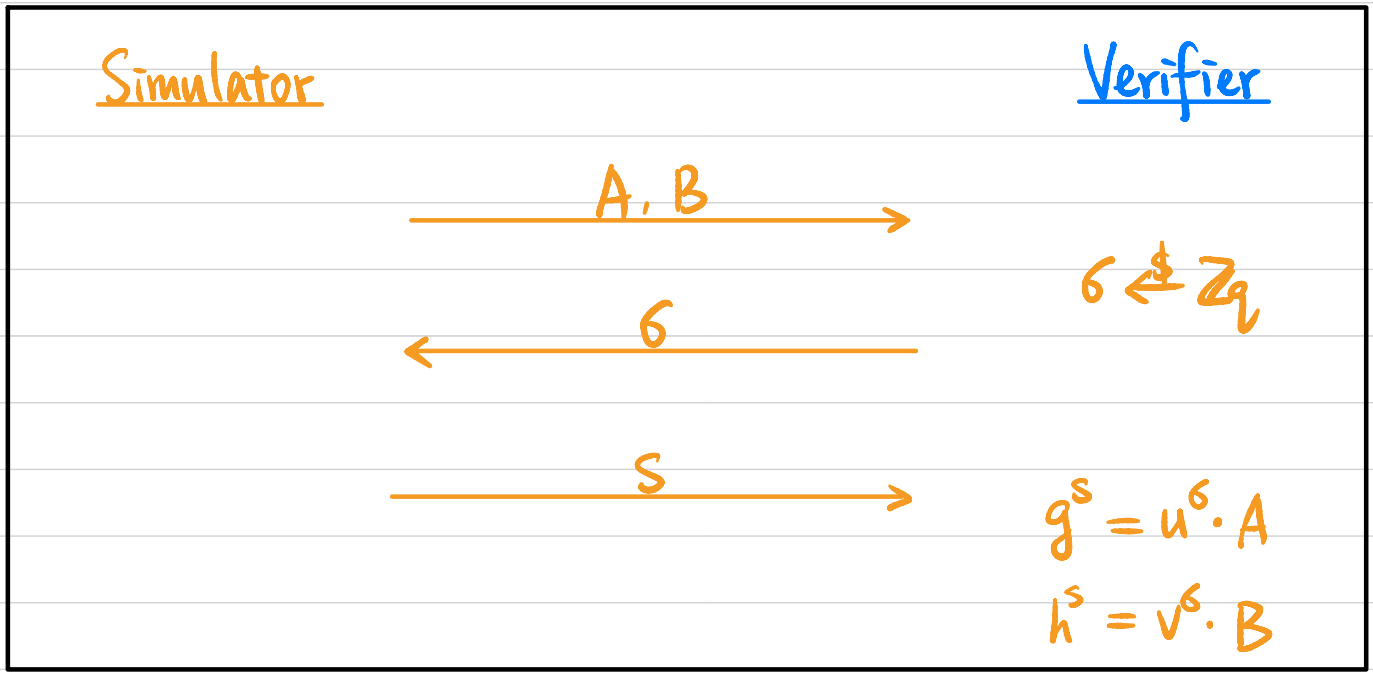
How to extract b s.t. $u = g^b \wedge v = h^b$?

Example: Diffie-Hellman Tuple

Honest-Verifier Zero-Knowledge (HVZK)?

$\exists$ PPT S s.t. $\forall (x, w) \in R_L$,

$$\text{View}_V[P(x,w) \longleftrightarrow V(x)] \simeq S(x)$$



How to generate (A, B, s) s.t. $g^s = h^\sigma \cdot A \wedge h^s = v^\sigma \cdot B$?