Lecture 9: Sigma Protocols

MIT - 6.5610

Spring 2023

Henry Corrigan-Gibbs

Plan

- Recap: PKE
  - * CCA Security
  - * CCA-Secure ElGamal enc.
- Defn: ID Protocols
- Break
- Defn: Sigma protocols
- Schnorr's ID protocol

Logistics

* Meet w/ us re: projects
* Post re: Proj on Piazza

Due Friday 5pm

- Team membership
- Pset 2

Recap: PKE

Over msg space $\mathcal{M}$

$Gen(1^\lambda) \to (sk, pk)$

$Enc(pk, m) \to ct$

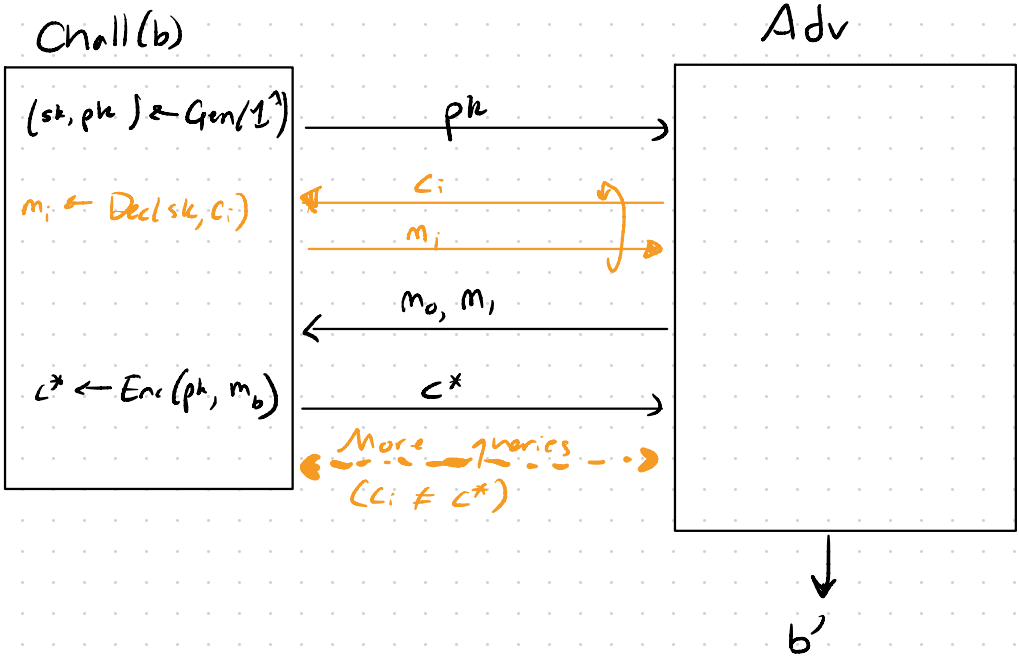
$Dec(sk, ct) \to m$

Correctness: For all $(sk, pk) \leftarrow Gen(1^\lambda)$, $\forall m \in \mathcal{M}$

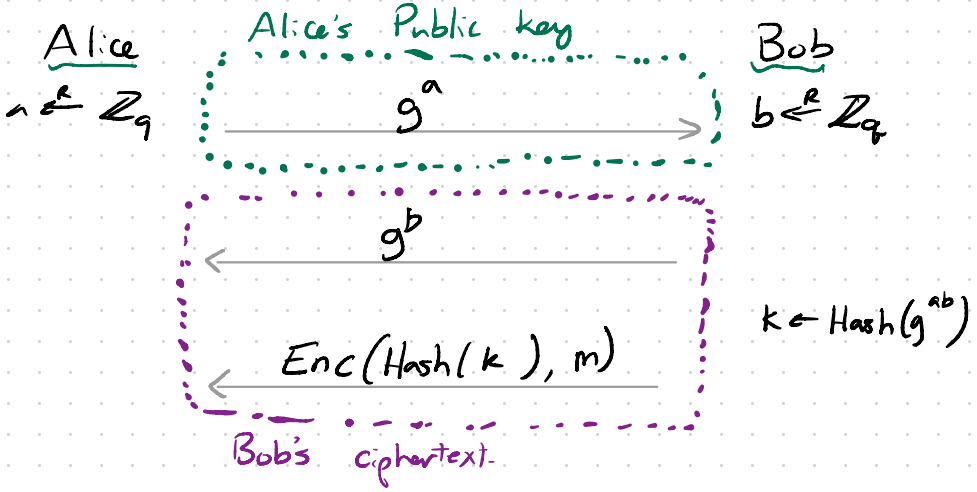
$Dec(sk, Enc(pk, m)) = m$.

CPA Security $\forall$ eff advs $\exists$ negl fn st. advs advantage in dist $b=0$ vs. $b=1$ is negl(n)...

[CCA security allows decryption queries]



DH as Pub-key Enc



Params: AE Scheme (Enc_{AE}, Dec_{AE}) Keyspace $\mathcal{K}$

Hash fn: $H: \mathbb{G} \to \mathcal{K}$

Group $\mathbb{G}$ of order q, generator $g \in \mathbb{G}$

$(\mathbb{G} = \{g, g^2, g^3, g^4, \ldots, g^q\})$

Gen() $\to$ $x \xleftarrow{R} \mathbb{Z}_q$

return $sk = x$, $pk = g^x$

Enc(pk, m) $\to$ $r \xleftarrow{R} \mathbb{Z}_q$

$k \leftarrow Hash(pk^r) = Hash(g^{xr})$

return $(g^r, Enc_{AE}(k, m))$

Dec(sk, (c_0, c_1)) $\to$ $k \leftarrow Hash(c_0^{sk}) = Hash(g^{xr})$

return $Dec_{AE}(k, c_1)$

(See Boneh-Shoup §12.4 for details)

If we model H as a random oracle,* and "interactive CDH assumption" holds, this scheme is CCA secure PKE

*more later

[Given (g, g^x, g^y) and oracle $\mathcal{O}(u, v) \mapsto \{u^x \stackrel{?}{=} v\}$ compute g^{xy}]

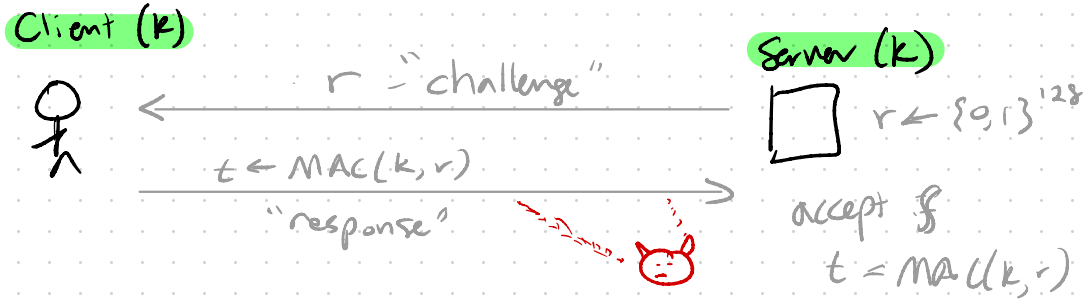
Plan for the rest of the week...

- Identification protocol
  - * Defn
  - * Schnorr
- Fiat-Shamir heuristic
- Digital signatures
  - * Defn
  - * Construction

} Next time

Identification Protocol

Way back when we talked about MACs...



We didn't define security formally...

Weak goal: "security against eavesdropping attacks"

* Attacker watches many interactions b/w C & S
* Attacker tries to authenticate

(Why weak? What if attacker is server?)

→ ID Protocols used in some cases (chip & PIN, password, etc)
↳ As we'll see, digital sigs even more common.

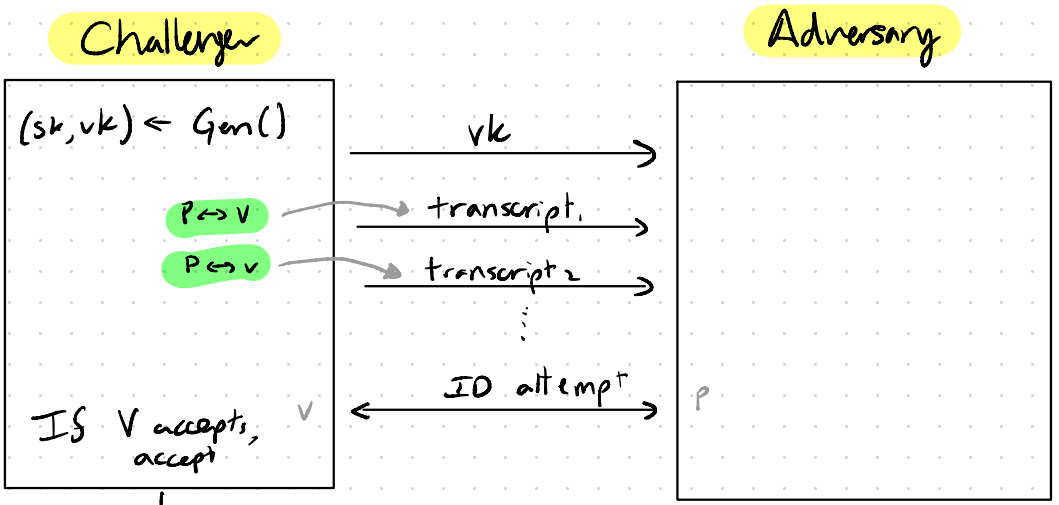
ID Protocol: Security against Eavesdropping Attacks

Consists of keygen alg $Gen() \to (sk, vk)$,
client alg P, prover
server alg V, verifier

See Boneh-Shoup §18

P & V can be stateful, interactive.

Attacker's power: See server state
Attacker's goal: Authenticate as client (impersonate)



Secure if adv causes chal to accept w/ negl prob

[N.B. Security here only holds when server is honest. Doesn't protect against active attacks: server that deviates from protocol.]

PROBLEM:

MAC and password-based ID schemes

NOT secure against eavesdropping attack! v_k is secret

If attacker gets v_k (compromises server) $\Rightarrow$ can authenticate as client!

Credential breaches are common! RSA, LastPass, ...

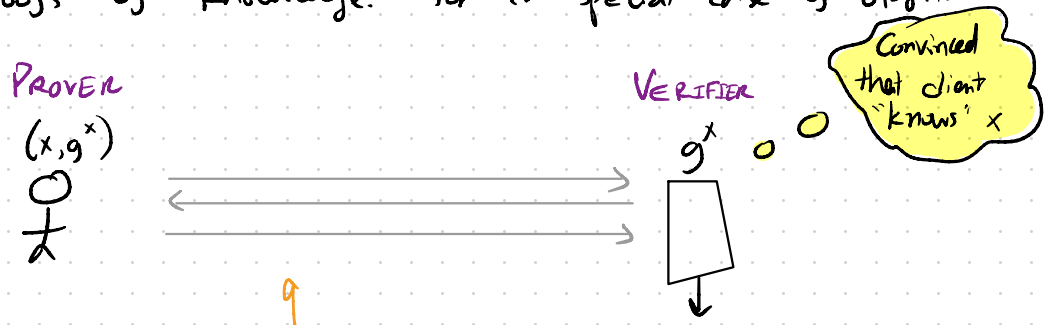
Goal: A "public-key" ID protocol.

↳ No secrets stored on server.

Useful as stepping stone to construct digital sig schemes (next time).

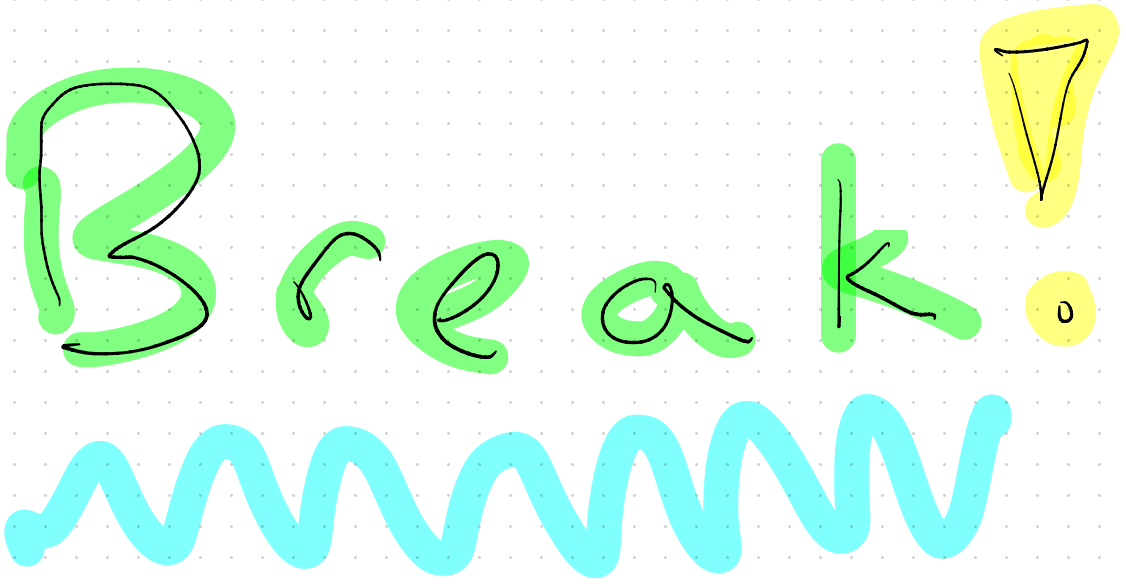
↳ Used everywhere!

To build pk ID protocols, we will construct Zk "proofs of knowledge" for the special case of dlog...



↳ [flow of msgs looks like Σ (sigma)]

[Yael will cover Zk later on in MUCH more detail.]


Break!

Defn of Sigma Protocol

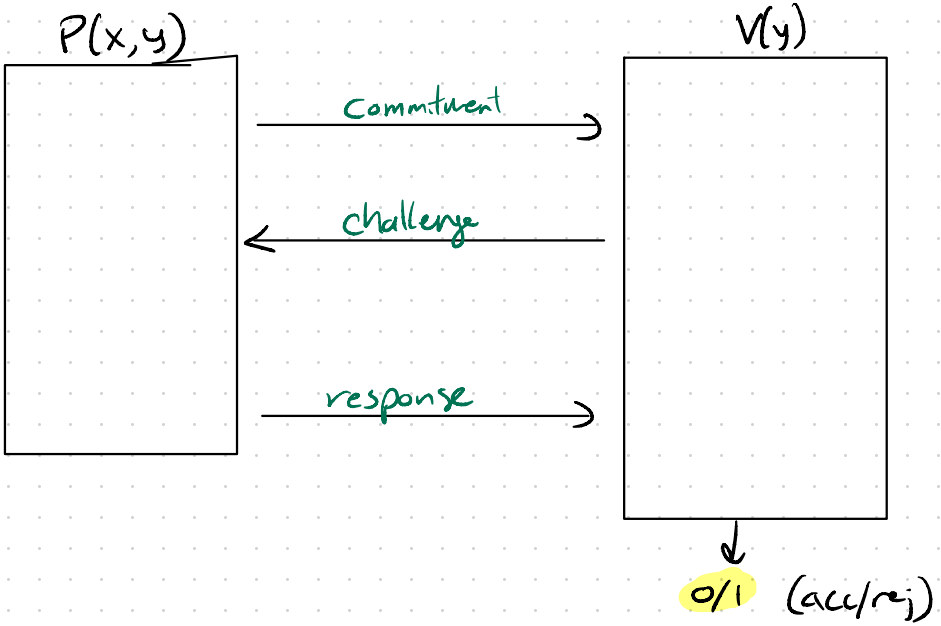
Let $R \subseteq \{0,1\}^* \times \{0,1\}^*$ be a relation. [efficiently recognizable]

Think: $R = \{(x, g^x) \mid x \in \mathbb{Z}_q\}$ for $G = \langle g \rangle$ order q

For $(x, y) \in R$

x: witness, y: statement

Sigma protocol consists of two algs (P, V):



Let $\langle P, V \rangle (x,y)$ denote output of P & V on (x,y).

Defn: A sigma protocol for relation R is eff algs (P, V) that satisfies the following props:

1. **Completeness.** Honest V accepts honest P.

$$\forall\ (x, y) \in R$$

$$\Pr\left[\langle P, V\rangle(x, y) = 1\right] = 1.$$

2. **Knowledge soundness.** Only way P^* convinces V is by "knowing" witness x.

[Philosophical Q: What does it mean for an algorithm to know x? Any ideas?]

Idea: P^* "knows" x if given access to successful P^*, can "extract" x from it.

$\exists$ eff alg E s.t. for transcripts that $V(y)$ accepts $(t, c, z), (t, c', z')$ with $c \neq c'$

$$E\left((t, c, z), (t, c', z')\right) = x.$$

[E is the "extractor."]

3. Honest-verifier zero knowledge.

Honest V "learns nothing about x" from interacting with honest P.

[Philosophical Q: What does it mean to "learn nothing" from an interaction?]

Idea: V has "learned nothing" if it can eff. simulate tx of its interaction with P.

(P,V) is HVZK if $\exists$ eff Sim s.t. $\forall (x,y) \in R$

$$\{\text{msgs between } P(x,y) \text{ and } V(y)\} \approx \{Sim(y)\}.$$

→ Input to simulator captures "what leaks" to V.

There are "simple" Sigma protocols for:

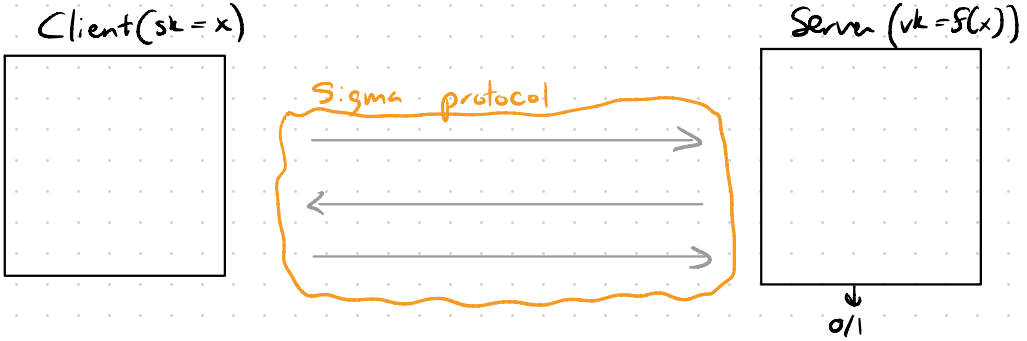
* Dlog $R = \{(x, g^x) : x \in \mathbb{Z}_p\}$
* RSA
* Factoring $R = \{((p,q), p \cdot q) : p,q \in Primes\}$
* Any NP relation .. 3SAT, 3Coloring, ...
  ↳ Not necessarily concretely efficient

Back to ID Protocols

For a OWF $f: X \to Y$ define

$$R_f := \{(x, f(x)) : x \in X\}$$

$$\text{Gen}() \to \begin{cases} x \xleftarrow{R} X, & y \leftarrow f(x) \\ sk \leftarrow x, & vk \leftarrow y \end{cases}$$



Claim: This ID protocol is secure against eavesdropping attacks.

HVZK $\Rightarrow$ Eavesdropper "learns nothing" about secret x by seeing many auth transcripts (except vk)

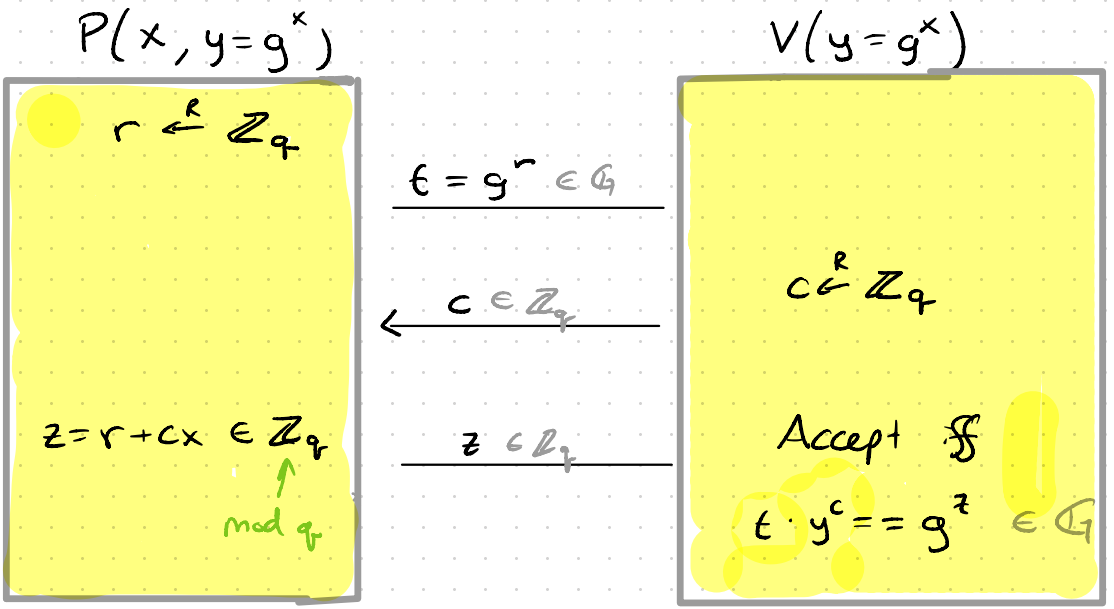
Knowledge $\Rightarrow$ Cheating client can't auth
↳ Cheats w.p. $\leq 1/|\text{challenge space}| + \sqrt{\text{OWF Adv}}$

Any client who can cheat scheme can invert OWF.

Schnorr's Protocol

Group $G = \{g, g^2, g^3, \ldots, g^q\}$ of prime order q in which dlog assumption holds.

Relation $R = \{(g, g^x) : x \in \mathbb{Z}_q\}$.



Important that challenge space is large.
But can be $\ll q$. – e.g. 128 bits instead of 256.
If P can guess challenge in advance, can cheat V.

Schnorr's Protocol: Analysis

1. Completeness. For $(x, y^x) \in R$ verifier accepts iff

$$y^c \cdot t = g^z$$
$$g^{xc} \cdot g^r = g^{r+cx} \quad \checkmark$$

2. Knowledge soundness. Consider accepting txs:

(t, c, z), (t, c', z') $\quad c \neq c'$

[Extractor outputs $x = \frac{z - z'}{c - c'} \in \mathbb{Z}_q$. Why?]

$$t \cdot y^c = g^z \qquad t \cdot y^{c'} = g^{z'}$$

$$y^{c-c'} = g^{z-z'}$$

Since $c - c' \neq 0$, $\frac{1}{c-c'}$ exists.

$$y = g^{\frac{z-z'}{c-c'}} \in G$$

$$\Rightarrow x = \frac{z - z'}{c - c'} \in \mathbb{Z}_q$$

3. HVZK.

Consider the simulator.

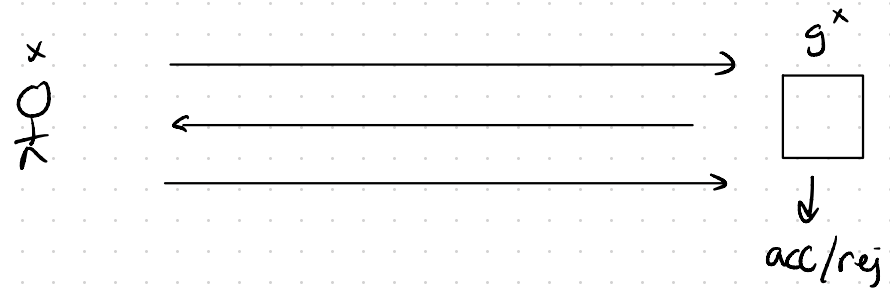
$$\text{Sim}(y) = \begin{cases} c \xleftarrow{R} \mathbb{Z}_q \\ t \leftarrow g^z \cdot y^{-c} \in G \\ \text{output } (t, c, z) \end{cases}$$

Need to show simulation is perfect.

* In true ts, r, c are independent uniform $\in \mathbb{Z}_q$.
* Value t is s.t. $t\, y^c = g^z \in G$.
* Same here!

Full ZK allows verifier to deviate from protocol.
Sim must work for all V^* — not just honest.
$\Rightarrow$ This simulation fails!
e.g. $c = AES(g^r, 0)$

So we now have an ID scheme from dlog



Beware: Typically don't want to use directly.
e.g. active attacks.

Extensions: "OR" Protocols

P can convince V that it knows 1 of n dlogs

Idea: Run n sigma protocols in parallel.
P can "cheat" on at most one of them

$P(x_i^*)$ | $V(g^{x_1}, g^{x_2}, \ldots, g^{x_n})$ where $y_1 = g^{x_1}$, $y_2 = g^{x_2}$, $y_n = g^{x_n}$

For $i = 1, \ldots, n$
$(t_i, c_i, z_i) \leftarrow Sim(y_i)$
$r_{i^*} \xleftarrow{R} \mathbb{Z}_q$
$t_{i^*} \leftarrow g^{r_{i^*}}$

$t_1, \ldots, t_n$ →

$c \xleftarrow{R} \mathbb{Z}_q$

← c

Choose
c_{i^*} s.t.
$c_{i^*} + \sum_{i=1, i \neq i^*}^{n} c_i = c \in \mathbb{Z}_q$

$c_1, \ldots, c_n$
$z_1, \ldots, z_n$ →

$z_{i^*} \leftarrow r_{i^*} + c_{i^*} \cdot x_{i^*} \in \mathbb{Z}_q$

For all $i \in \{1, \ldots, n\}$
$g^{z_i} \stackrel{?}{=} t_i \, y_i^{c_i}$

Given two accepting txs, argue that $\exists\ i^*$ s.t.
$c_{i^*} \neq c'_{i^*} \Rightarrow$ Can extract at least one dlog.