Lecture 10: Digital Signatures

MIT - 6.5610

Spring 2023

Henry Corrigan-Gibbs

Plan

- Recap: Schnorr's ID Protocol
  - ↳ Extensions (?)
- Defn: Digital Sigs
- Break
- Schnorr signatures (ECDSA, ...)
- Fiat-Shamir Heuristic
- Certificates

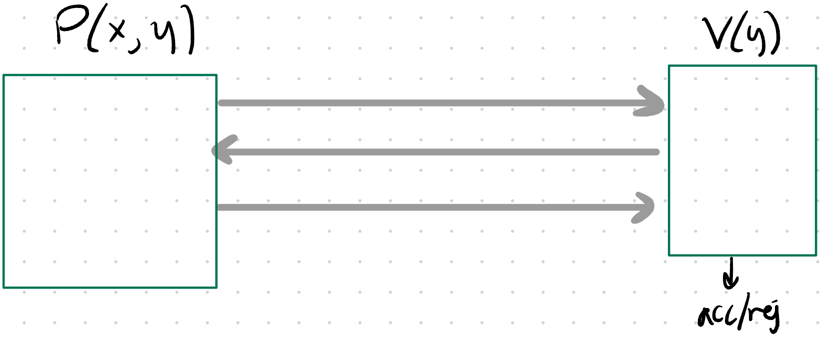
Logistics

- Ask us re: team membership (Friday!)
- Pset 2 due Friday
- Anon feedback!

Recap: ZK Proof of Knowledge

Relation $\underset{\text{witness}}{R \subseteq \{0,1\}^*} \times \underset{\text{statement}}{\{0,1\}^*}$

e.g. $R_{G,g} = \{(x, g^x) : x \in \mathbb{Z}_q\}$
for $G = \{g, g^2, \ldots, g^q\}$



Properties

1. Completeness $\forall (x,y) \in R \quad \langle P, V\rangle(x,y) = 1$.

2. Knowledge Soundness $\exists$ eff E s.t. $\forall y \; \forall P^*$

$$\underbrace{\Pr\left[(x,y) \in R : x \leftarrow E^{P^*}(y)\right]}_{\text{Can 'extract' witness from } P^*} \geq \underbrace{\Pr\left[\langle P^*, V\rangle(\cdot, y) = 1\right]}_{\text{Cheating } P^* \text{ convinces honest } V} - \underbrace{\varepsilon}_{\text{small}}$$

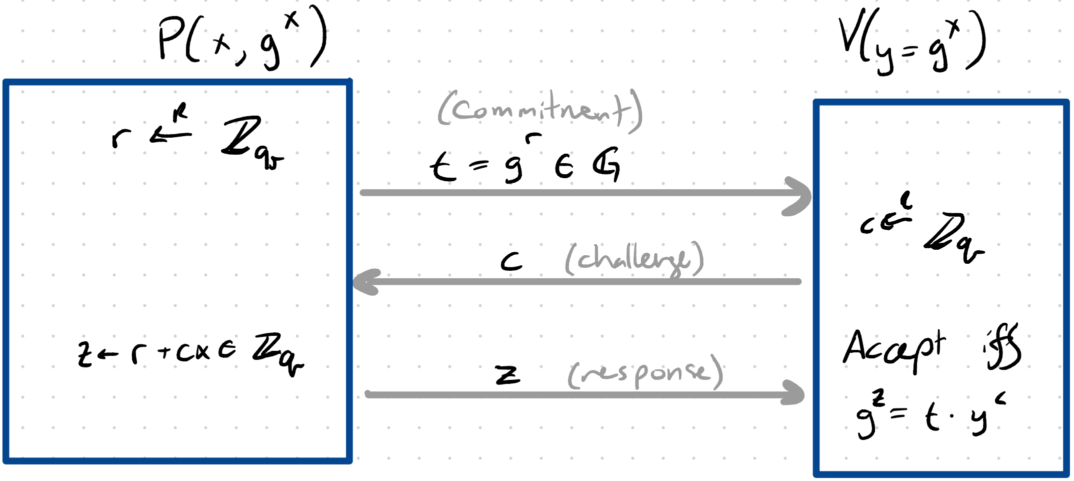
Intuition for extraction

3. HVZK $\exists$ eff Sim s.t. $\forall (x,y) \in R$

$$\left\{\begin{matrix}\text{transcript of} \\ P \leftrightarrow V \text{ on } (x,y)\end{matrix}\right\} \overset{c}{\approx} \left\{ Sim(y) \right\}$$

Can simulate interaction WITHOUT knowing witness
⇒ "learn nothing" about witness

Schnorr: ZK PoK for Dlog



$P(x, g^x)$

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

$z \leftarrow r + cx \in \mathbb{Z}_q$

(commitment)

$t = g^r \in \mathbb{G}$

c (challenge)

z (response)

$V(y = g^x)$

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

Accept iff

$g^z = t \cdot y^c$

Showed last time: Completeness, knowledge.

For simplicity assume: $\Pr[\langle P^*, V\rangle(y) = 1] = 1$.

Extractor $E^{P^*}(y) :=$

```
Run $P^* \to (t, c, z)$
Rewind $P^*$ to point before sending $c$
Run $P^* \to (t, c', z')$
Extract $\mathrm{dlog}_g(y)$ As in last lecture
    $x = \frac{z - z'}{c - c'} \in \mathbb{Z}_q$
```

Showing that E succeeds often requires a bit of work...

Schnorr: Analysis

Now: HVZK... Need to construct Sim

```
Sim$(y = g^x)$:
    $c, z \xleftarrow{\$} \mathbb{Z}_q$
    $t \leftarrow g^z \cdot y^{-c} \in \mathbb{G}$
    output $(t, c, z)$
```

Claim: $\{\text{real tx on } (x, y)\} \equiv \{\text{Sim}(y)]$

- For each (c, z) in real $\exists$ exactly one t, equiprobable

↪ Exactly the same in simulation.

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

$P(x_{i^*})$ $\qquad\qquad$ $V(\overset{y_1=}{g^{x_1}}, \overset{y_2=}{g^{x_2}}, \dots, \overset{y_n=}{g^{x_n}})$

For $i=1,\dots,n$

$(t_i, c_i, z_i) \leftarrow Sim(y_i)$

$r_{i^*} \xleftarrow{R} \mathbb{Z}_q$

$t_{i^*} \leftarrow g^{r_{i^*}}$

$$\xrightarrow{t_1,\dots,t_n}$$

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

$$\xleftarrow{c}$$

Choose

c_{i^*} s.t.

$$c_{i^*} + \sum_{\substack{i=1\\ i\neq i^*}}^{n} c_i = c \in \mathbb{Z}_q$$

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

$$\xrightarrow[z_1,\dots,z_n]{c_1,\dots,c_n}$$

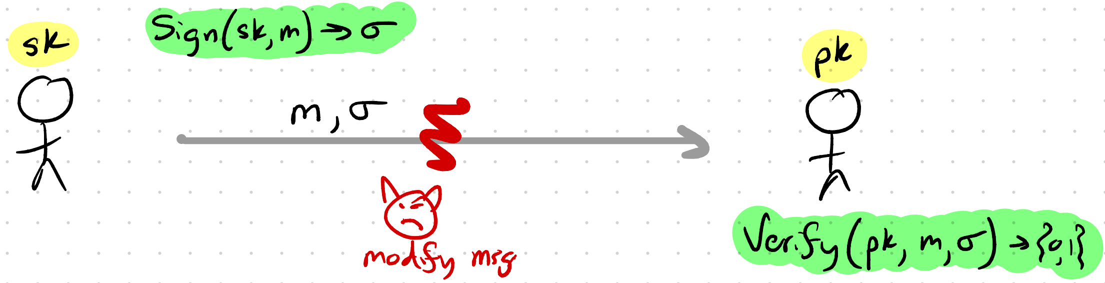
For all $i \in \{1,\dots,n\}$

$g^{z_i} \overset{?}{=} t_i \, y_i^{c_i}$

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

Digital Signatures

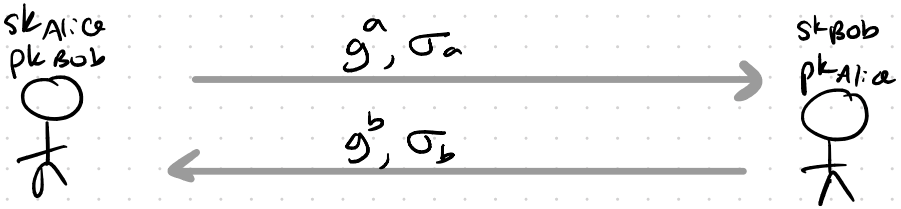
- Public-key version of a MAC.
- Used everywhere! HTTPS, s/w update, ssH, VPN, enc msg, ...


sk
$\text{Sign}(sk, m) \to \sigma$
pk
m, σ
modify msg
$\text{Verify}(pk, m, \sigma) \to \{0,1\}$

Verifier should detect tampering by adversary

App: Authenticated DH key Exchange

★ Used in practice!



Q: Where does Alice get pk_{Bob}?

↳ Did we just move the problem around

Digital Sigs: Defn

Msg space $\mathcal{M}$. Three eff algs:

$\mathrm{Gen}(1^n) \to (sk, pk)$

$\mathrm{Sign}(sk, m) \to \sigma$

$\mathrm{Verify}(pk, m, \sigma) \to \{0,1\}$

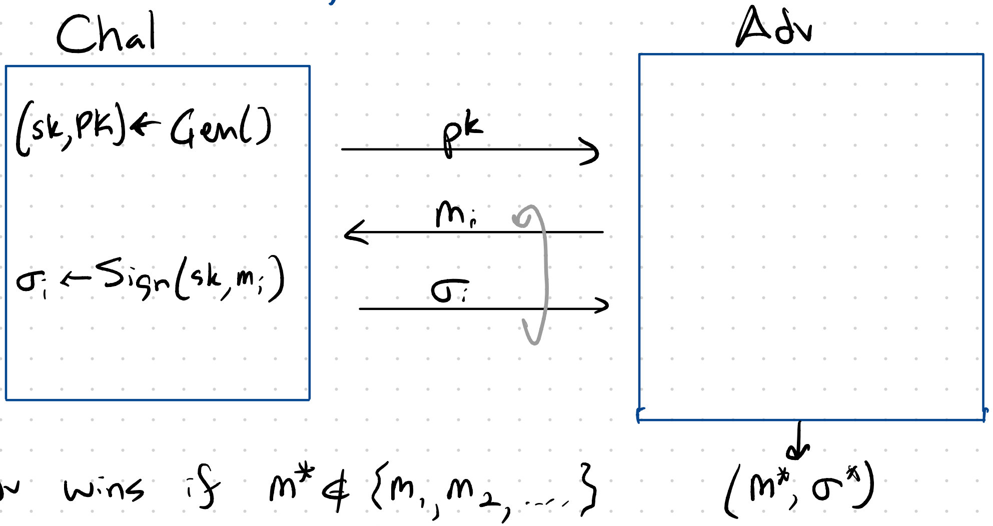
Correctness:

$\forall\ (sk, pk) \leftarrow \mathrm{Gen}(1^\lambda)\ \ \forall\ m \in \mathcal{M}$

$$\mathrm{Ver}(pk, m, \mathrm{Sign}(sk, m)) = 1$$

Security: Existential unforgeability under chosen msg attack (EUF-CMA)

$\forall$ eff adv A $\exists$ negl fn s.t. A's advantage in following game is negl:



Chal
Adv
$(sk, pk) \leftarrow \mathrm{Gen}()$
pk
m_i
σ_i
$\sigma_i \leftarrow \mathrm{Sign}(sk, m_i)$
(m^*, σ^*)

Adv wins if $m^* \notin \{m_1, m_2, \dots\}$
AND $\mathrm{Ver}(pk, m^*, \sigma^*) = 1$

Notes on Sec Def

- Strong: - Adv sees sigs on msgs of its choice
  - Can forge on any msg

- BUT: admits schemes in which given (m, σ) can generate (m, σ')

  New sig on old msg

Break

Constructing Digital Sigs

Many nice ways to do it!

- * From OWF (Lamport, ...)
- * Trapdoor OWF (RSA)
- * PoK protocol + OWF (Schnorr, ...)

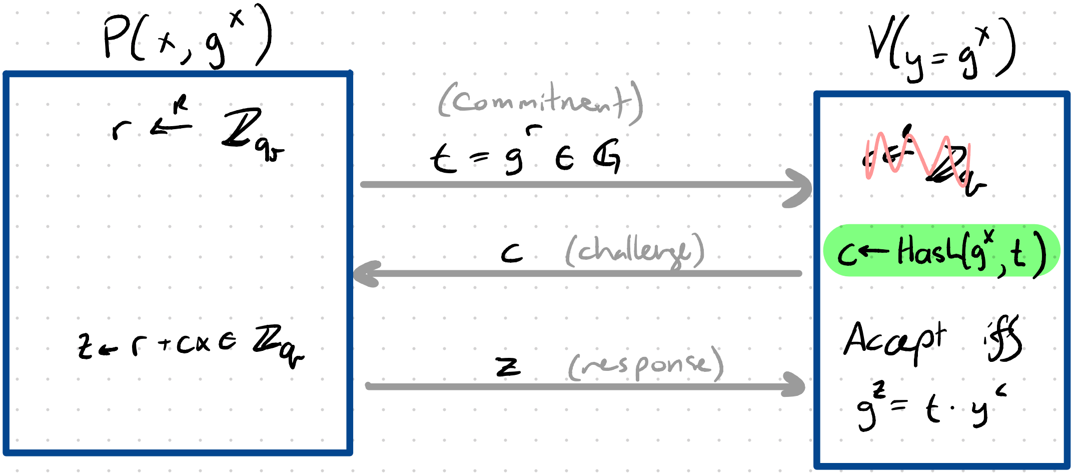
↰ We will see this one.

- * On Internet today, Schnorr-like schemes common (Why EC-DSA)
- * RSA less & less common – longer sigs & pk (256B vs 32/64 B)
- * PQ schemes coming

Basic idea of Schnorr sig

- * Take interactive Sigma protocol & make it non interactive.
- * Proof of knowledge of sk becomes sig
  ↳ Whoever generated pf must know sk
- * Bind message to be signed in there somewhere

Back to Schnorr



$P(x, g^x)$

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

$z \leftarrow r + cx \in \mathbb{Z}_q$

(commitment)

$t = g^r \in \mathbb{G}$

c (challenge)

z (response)

$V(y = g^x)$

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

$c \leftarrow Hash(g^x, t)$

Accept iff

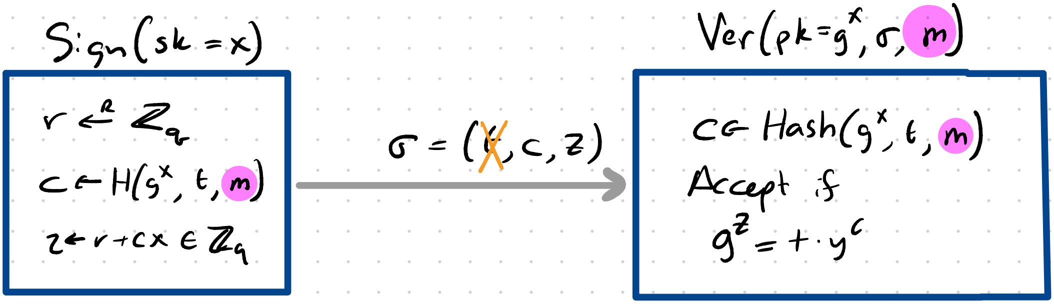
$g^z = t \cdot y^c$

Schnorr Signatures

* Ed25519, much the same

* EC-DSA same idea but tweaked to avoid patents

Signature scheme is almost the same, except w/ msg hashed in when computing challenge.



$Sign(sk = x)$

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

$c \leftarrow H(g^x, t, m)$

$z \leftarrow r + cx \in \mathbb{Z}_q$

$\sigma = (\cancel{t}, c, z)$

$Ver(pk = g^x, \sigma, m)$

$c \leftarrow Hash(g^x, t, m)$

Accept if

$g^z = t \cdot y^c$

Optimized $\left\{ \begin{array}{l} t' \leftarrow g^z \cdot y^{-c} \\ \text{Accept if } c = Hash(g^x, t') \end{array} \right.$

Keygen just generates dlog instance

$$Gen() = \left| \begin{array}{l} x \xleftarrow{R} \mathbb{Z}_q \\ \text{return } (x, g^x) \end{array} \right.$$

What about security?

→ We converted an interactive to a non-interactive one using a hash fn.

“Fiat-Shamir heuristic”

→ For which choices of hash fn H does this transformation preserve security of the underlying scheme.

↳ More later...

Two approaches: [Different views of same thing?]

1. Make new assumption

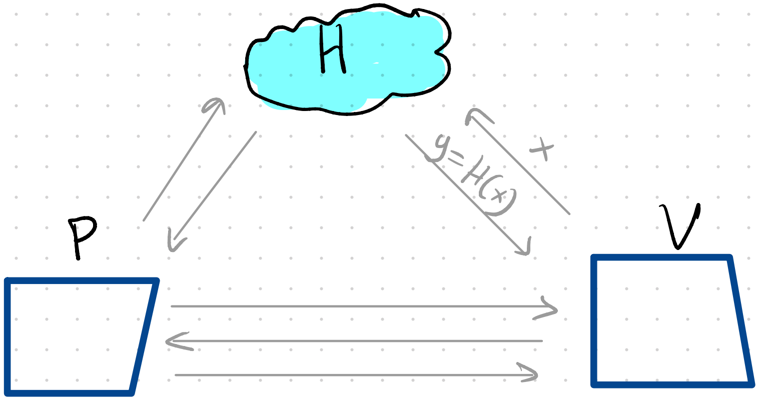
Plug in “reasonable” crypto hash fn (e.g. SHA2) and assume that the resulting sig scheme is secure

↳ Not so elegant? But pragmatic

2. Change the model of computation

“Random-oracle model” [BR93]

↳ Assume that all parties have (only) oracle access to a true random hash fn.



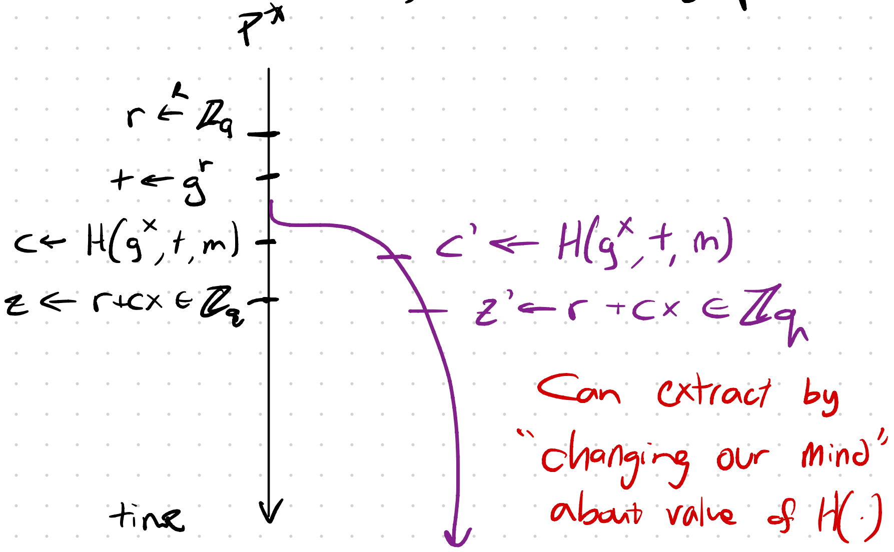
More on Random-Oracle Model (ROM?)

Inf. Thm: IF Schnorr is secure ID scheme against eavesdropping attacks, Schnorr sig scheme is secure sig scheme (EUF-CMA), provided that we model hash fn H as R.O.

Why does R.O.M. help argue security?

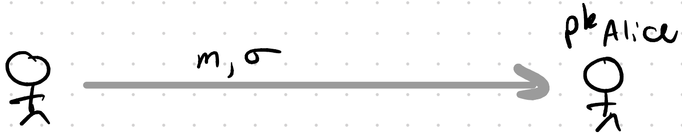
Intuition: In Schnorr ID scheme cheating P* really cannot predict what the challenge will be!

Technically: Even in non-interactive setting can extract dlog from cheating prover P*


P^*
$r \xleftarrow{R} \mathbb{Z}_q$
$t \leftarrow g^r$
$c \leftarrow H(g^x, t, m)$
$z \leftarrow r + cx \in \mathbb{Z}_q$
time
$c' \leftarrow H(g^x, t, m)$
$z' \leftarrow r + cx \in \mathbb{Z}_q$

Can extract by "changing our mind" about value of $H(\cdot)$

Certificates & PKI = Pub key infrastructure



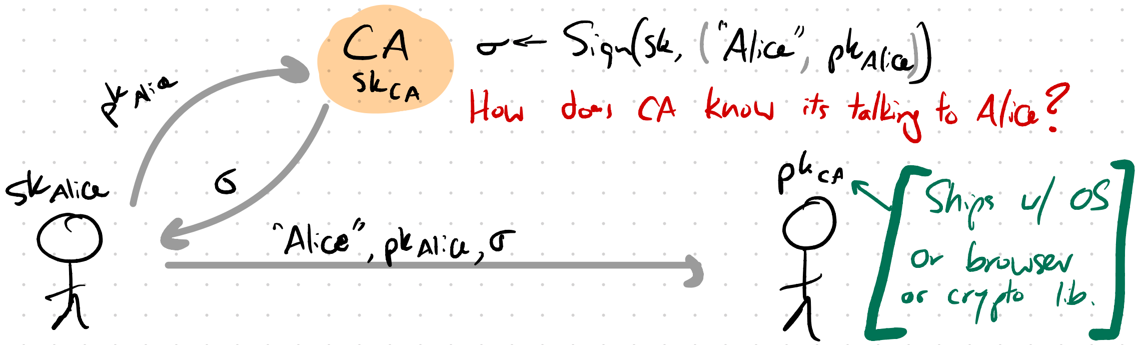
Where does Bob get Alice's pk?

Many options. All bad in their own way.

1. Name as Pk, as in Bitcoin, Tor hidden svcs
   - + Solves pk dist problem
   - - Lose key? Remember?

2. Trust on first use, as in SSH, Signal, WhatsApp
   - + Simple, intuitive, effective?
   - - No protection on 1st msg, key changes?

3. Certificates, used in TLS (HTTPS in your browser, etc.)
   - + Scales well, no online CA interaction
   - - Validation weak, lost key?, "weakest link" security (compromising one CA is enough to forge any* cert)



$\sigma \leftarrow \mathrm{Sign}(sk, (\text{"Alice"}, pk_{Alice}))$

How does CA know its talking to Alice?

[Ships w/ OS or browser or crypto lib.]