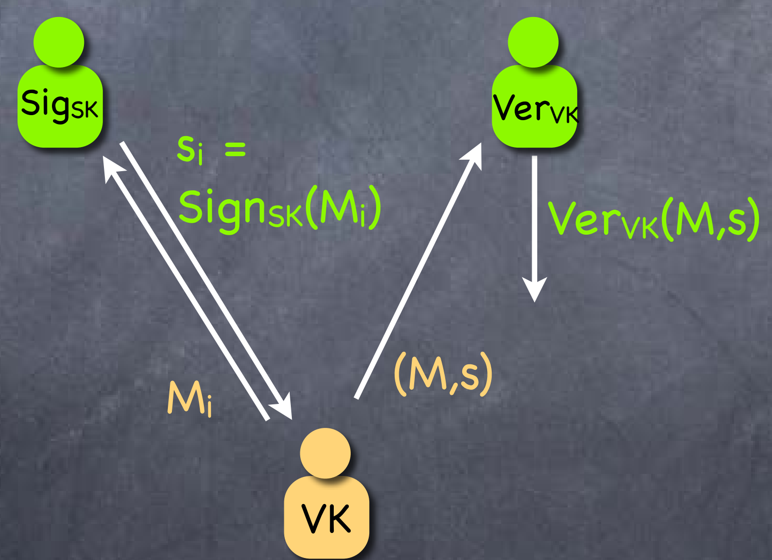
Digital Signatures (ctd.)

Lecture 17

RECALL

Digital Signatures

- Syntax: KeyGen, Sign_{SK} and Verify_{VK}.
  Security: Same experiment as MAC's, but adversary given VK



Advantage = $\Pr[\ \mathrm{Ver}_{VK}(M,s)=1$ and $(M,s) \notin \{(M_i,s_i)\}\ ]$

Weaker variant: Advantage = $\Pr[\ \mathrm{Ver}_{VK}(M,s)=1$ and $M \notin \{M_i\}\ ]$

RECALL

Signatures from OWF

Summary

- One-time, fixed-length message signatures (Lamport)
  $\xrightarrow{\text{Domain-Extension}}$ arbitrary length messages (using UOWHF)
  $\xrightarrow{\text{"Certificate Tree"}}$ many-time signatures (using PRF)
- So, in principle, full-fledged digital signatures can be entirely based on OWF
- Not very efficient: Say hashes are O(k) bits long. Then, a signature contains O(k) VKs of Lamport signature, each of which, to allow signing O(k) bit messages, is $O(k^2)$ bits long
- Today: More efficient schemes

Hash and Invert

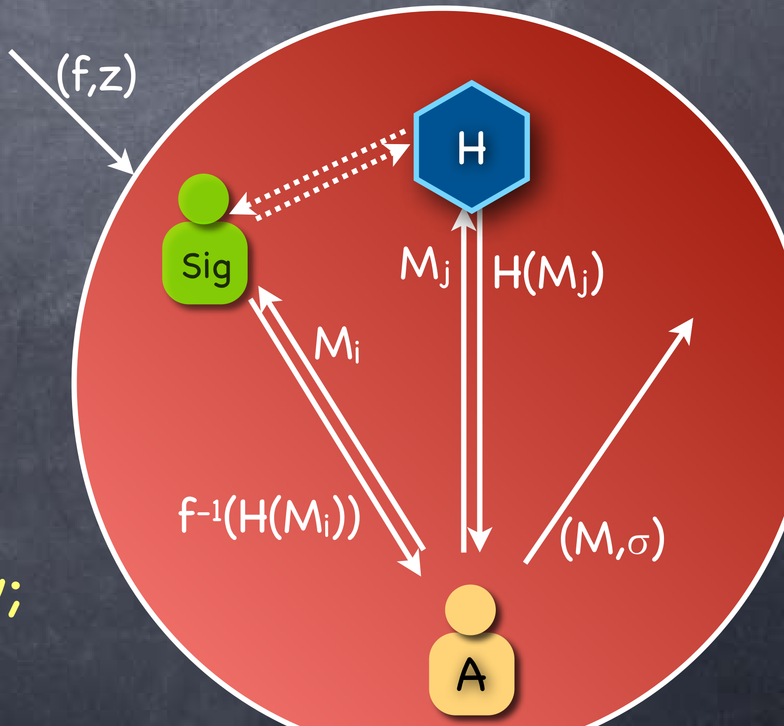
- Diffie-Hellman suggestion (heuristic): Sign(M) = f^{-1}(M) where (SK,VK) = (f^{-1},f), a Trapdoor OWP pair. Verify(M,σ) = 1 iff f(σ)=M.
  - Attack: pick σ, let M=f(σ) (Existential forgery)
- Fix, using a "hash": **Sign(M) = f^{-1}(Hash(M))**
  - Secure in the **random oracle** model
  - Hash can handle variable length inputs
  - RSA-PSS in RSA Standard PKCS#1 is based on this

Proving Security in the RO Model

- To prove: If Trapdoor OWP secure, then Sign(M) = f^{-1}(Hash(M)) is a secure digital signature, when Hash is modelled as a random oracle
  - Hope: Since adversary can't invert Hash, needs to compute f^{-1}
  - Problem: Signing oracle gives adversary access to the f^{-1} oracle. But then, trapdoor OWP gives no guarantees!
  - But adversary only sees $(x, f^{-1}(x))$ where x = Hash(M) is random. This can be arranged by picking $f^{-1}(x)$ first and fixing Hash(M) afterwards!
- Modeling as an RO: RO randomly initialized to a random function H from $\{0,1\}^*$ to $\{0,1\}^k$
  - Signer and verifier (and forger) get oracle access to H(.)
  - All probabilities also over the initialization of the RO

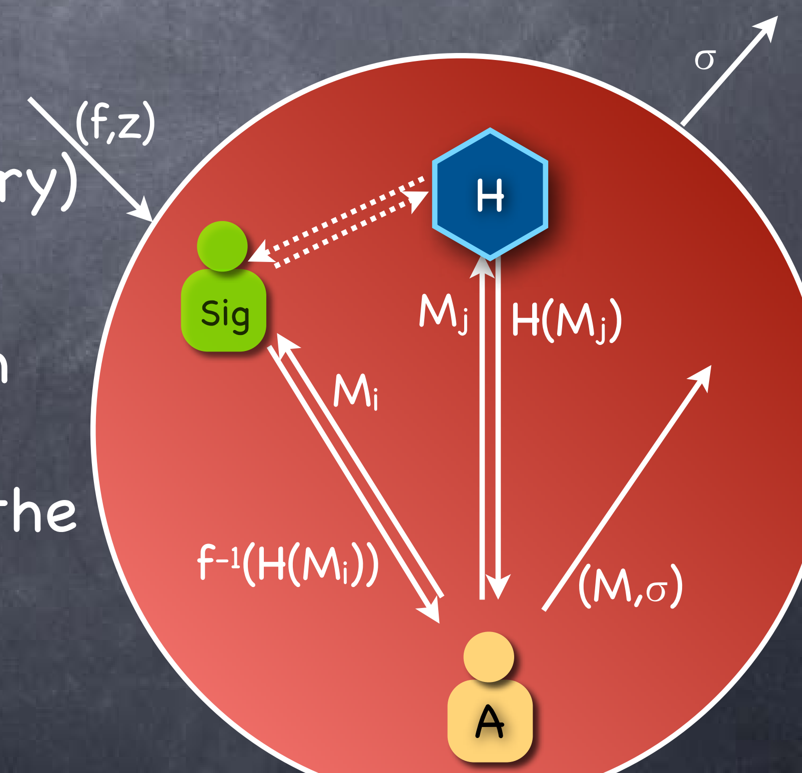
Proving Security in ROM

- Reduction: If A forges signature (where Sign(M) = $f^{-1}(H(M))$ with (f,f^{-1}) from Trapdoor OWP and H an RO), then A* that can break Trapdoor OWP (i.e., given just f, and a random challenge z, can find $f^{-1}(z)$ w.n.n.p). A*(f,z) runs A internally.
  - A expects f, access to the RO and a signing oracle f^{-1}(Hash(.)) and outputs (M,σ) as forgery
  - A* can implement RO: a random response to each new query!
  - A* gets f, but doesn't have f^{-1} to sign
    - But x = H(M) is a random value that A* can pick!
    - A* picks H(M) as x=f(y) for random y; then Sign(M) = f^{-1}(x) = y



Proving Security in ROM

- A* s.t. if A forges signature, then A* can break Trapdoor OWP
  - A* implements H and Sign: For each new M queried to H (including by Sign), A* sets H(M)=f(y) for random y; Sign(M) = y
  - But A* should force A to invert z
    - For a random (new) query M (say t^{th}) A* sets H(M)=z
      - Here queries include the "last query" to H, i.e., the one for verifying the forgery (which may or may not be a new query)
  - Given a bound q on the number of queries that A makes to Sign/H, with probability 1/q, A* would have set H(M)=z, where M is the message in the forgery
    - In that case forgery ⇒ $\sigma = f^{-1}(z)$



Schnorr Signature

- Public parameters: (G,g) where G is a prime-order group and g a generator, for which DLA holds, and a random oracle H
  - Or (G,g) can be picked as part of key generation
- Signing Key: $y \in Z_q$ where G is of order q. Verification Key: $Y = g^y$
- $\mathrm{Sign}_y(M) = (x,s)$ where $x = H(M||g^r)$ and $s = r-xy$, for a random r
- $\mathrm{Verify}_Y(M,(x,s))$: Compute $R = g^s \cdot Y^x$ and check $x = H(M||R)$
- Secure in the Random Oracle Model under the Discrete Log Assumption for a group
  - Alternately, under a heuristic model for the group (called the Generic Group Model), but under standard-model assumptions on the hash function

Cramer-Shoup Signature

- Based on "Strong RSA assumption." Here, a variant by Damgård-Koprowski based on "Strong Root Assumption."
  - For all PPT adversaries A, following probability is negligible:
  - Root Assumption: $\Pr_{G,X,e}[A(e,X) = T,\ T^e = X]$ (G,X,e) appropriately distributed
  - Strong Root Assumption: $\Pr_{G,X}[A(X) = (X,e),\ e>1,\ T^e = X]$
- Important that the order of G is unpredictable
  - In fact, $|G|$ yields $d = 1/e \bmod |G|$ s.t. with $T=X^d$, we have $T^e = X$. Will use large prime e, to guarantee $\gcd(e,|G|) = 1$.
- KeyGen: $VK = (H,G,g,X,e)$ and $SK = (VK,|G|)$ where $H \leftarrow$ CRHF, $(G,|G|) \leftarrow$ GroupGen, $g \leftarrow G$, $X = g^x$, e prime.

  Sign: (R,s,T) s.t. $R \leftarrow G$, $s \neq e$ large random prime, $Z = R^s g^{-H(message)}$, and $T = (Xg^{H(Z)})^{1/e}$ (where $1/e \bmod |G|$ is computed using $|G|$)

  Verify: Compute $Z = R^s/g^{H(message)}$. Check $s \neq e$ large, $T = (Xg^{H(Z)})^{1/e}$

Summary

- Digital signatures can be based on OWF + UWOHF + PRF
  - In turn based on OWF (or more efficiently on OWP)
- More efficiently, can be based on number-theoretic/algebraic assumptions (e.g., Cramer-Shoup signatures based on Strong RSA and CRHF)
- In practice, based on number-theoretic/algebraic assumptions in the random oracle model
  - RSA-PSS, of the form f^{-1}(Hash(M)), where f a Trapdoor OWP
  - DSA and variants, based on Schnorr signature

VK as ID: An Example
Identity-Based Encryption

- In PKE, KeyGen produces a random (PK,SK) pair
- Can I have a "fancy public-key" (e.g., my name)?
  - No! Not secure if one can pick any PK and find an SK for it!
- But suppose a trusted authority for key generation
  - Identity-Based Encryption: a key-server (with a master secret-key) that can generate a valid (PK,SK) pair for any PK
    - Encryption will use the master public-key, and the receiver's "identity" (i.e., fancy public-key)
    - In PKE, sender has to retrieve PK for every party it wants to talk to (from a trusted public directory)
    - In IBE, receiver has to obtain its SK from the authority

VK as ID: An Example
Identity-Based Encryption

- Security requirement for IBE (will skip formal statement):
  - Environment/adversary decides the ID of the honest parties
  - Adversary can adaptively request SK for any number of IDs (which are not used for honest parties)
  - "CPA security" for encryption with the ID of honest parties
- IBE (only CPA-secure) can easily give CCA-secure PKE!
  - Idea: Can't malleate an IBE ciphertext to change ID
- $\text{PKEnc}_{\text{MPK}}(m) = (\text{id}, C=\text{IBEnc}_{\text{MPK}}(\text{id}; m), \text{sign}_{\text{id}}(C))$
  - Security: can't create a different encryption with same id (signature's security); can't malleate using a different id (IBE's security)

Digital Signature with its public-key used as the ID in IBE