Special Topics in Cryptography

Mohammad Mahmoody
Last time

• RSA public key encryption
• Digital signatures

Today

• Finishing digital signatures
• Zero Knowledge Proofs
• Secure Computation.
Public Key Authentication: Digital Signatures

• Secure authentication without shared secret keys!
Defining Digital Signatures

• Alice has a signing key $sk$ and a verification key $vk$

• Using $sk$ Alice can sign $m$ with $\sigma = \text{Sign}_{sk}(m)$

• If Bob verifies $\text{Verif}_{vk}(m, \sigma) = 1$ he can be sure Alice signed $m$

• Security: For any poly-time adversary $A$ who has access to a signing oracle $\text{Sign}_{sk}(\cdot)$, the probability of $A$ finding $(m, \sigma)$ for $m$ not asked by $A$ that also passes the test $\text{Verif}_{vk}(m, \sigma) = 1$ is negligible.
One possible idea based on TDPs (e.g. RSA)

- Signing key: “private key” (or the trapdoor)
- Verification key: “public key” (or the description of the permutation)

To sign $m$ publish $\sigma(m) = t = \pi^{-1}(m)$

To verify $(m, t)$ accept if and only if: $\pi(t) = m$

Is it secure signature? No, because we can choose $t$ first and then find $m$ for it easily!
"Hash and sign" using ideal hash function

- Directly works for any message (arbitrary length):
- Suppose $h : \{0,1\}^* \rightarrow \{0,1\}^n$ for security parameter $n$
- And we have trapdoor permutation $\pi, \pi^{-1}$ on domain $\{0,1\}^n$

- Signing key $\pi^{-1}$
- Verification key $\pi$
- To sign $m$, first get $s = h(m)$ and then output $\sigma = \pi^{-1}(s)$

$\text{Verif} (m, t) : \frac{h(m) = \pi(t)}{s} \quad \frac{h(m) \neq \pi(t)}{\text{output 0}}$
Why is it a good signing method?

**Theorem:** If \((T, \tilde{n})\) is a secure TDP

\[
\Pr\left[ A \text{ wins the security game against the } \tilde{G} \text{ has a sign scheme} \right] \leq \text{neg}(\cdot)
\]

**Proof:** Reduction: assuming such \(A\)

\[
\rightarrow \text{ turn it into } B \text{ breaks } TDP \left( T, \tilde{n} \right)
\]
Zero Knowledge Proofs

• Proving the truth of statements, while revealing nothing about the proof!
Can we ever prove we know something without revealing the details of the secret?

- Alice knows a magic word to open the door inside the cave:

- How can she prove to Bob that she knows that word?

1. Alice opens the door
2. Bob enters and asks A/B
3. Alice exits from A/C
What is an “efficiently provable” property?

- Examples:

1. No known poly time alg for finding HC in a graph or even test if it exists.

2. If somebody knows cycle C in G, it is easy to convince others.

3-Coloring Problem \[ G \]

- Yes: if \( \exists \) way to color "nodes" with \( 1, 2, 3 \) such that \( i \mapsto j \quad \text{and} \quad C(i) \neq C(j) \).
What is an “efficiently provable” property?

• Complexity Class \( \text{NP} \): set of all languages \( L \) where there is an \textbf{efficient} verifier \( V \) for proving membership in \( L \). Name for all \( x \in L \) there is a “witness” (or proof) \( w \) that \( V(x, w) = 1 \) and if \( x \notin L \) then \( V(x, w) = 0 \).

\[
L : \{ x \in \{0, 1\}^* \mid \exists y \text{ s.t. } h(x, y) \text{ is valid} \}
\]

\( L \) \( \subseteq \) \( \text{NP} \)

\( \forall y \in \{0, 1\}^* \left( T(y) : \text{graph } G \text{ is 3-colorable} \iff \right. \}

\( \left. \exists \text{polynomial } T(y) \text{ s.t. } T(y)_G \right) \)}
NP complete problems

$L$ is NP complete if

Solving $L$ in poly-time can be used "in a very simple way" to solve any other $L' \in \text{NP}$. 
GMW: membership in any NP language can be proved Zero Knowledge!

• Enough to do it for one NP complete problem only.
• Idea: using *interaction*
• Suppose we have digital lockable envelopes

\[ A \xrightarrow{\text{Comm}[x]} B \]

1. Alice can "open" \( y \) into \( x \) and only \( x \)
2. Bob has no idea what \( x \) is by looking at \( y \)

\[ \exists C : \{1,2,\ldots,n\} \rightarrow \{1,2,3\} \]

Such that: \( \forall i,j \) \( M(i,j) = 1 \rightarrow c(i) \neq c(j) \)
Proving a graph is 3 colorable

Alice permutes the colors randomly.

Bob: nodes \( \{1, 2, \ldots, n\} \),
edges \( e(i,j) = 1 \) \( \neq i \rightarrow j \)

Pick \( C(i,j) \): connected edge at random amongst \( e_1, e_2, \ldots, e_m \)

if \( C \) is not a 3-coloring \( \rightarrow \) Bob catches the Alice by prob. \( \frac{1}{m} \)

\[ P_i(\text{catching Alice}) \leq \left(1 - \frac{1}{m}\right) \]

Bob opens \( E_i, E_j \) such that \( C(i) \neq C(j) \)

Bob cannot catch Alice if 3-coloring of \( C(i) \) is valid.

\( E_i \) contains \( C(i) \)

Know \( C(i) \) for all \( i \).
Why is this a convincing (sound) interactive proof?

because if (i) not 3-colorable Bob
    catch Alice \( \geq \frac{1}{m} \).

If we repeat protocol (from the begining) \( k \) time

\[
P_i(\text{not catch} : j) \leq \left( 1 - \frac{1}{m} \right)^k
\]

\[
\leq \left( \left( 1 - \frac{1}{m} \right)^m \right)^{100}
\]

\[
\leq e^{-100} \leq 2^{-100}
\]
Why is this proof carrying “zero knowledge”?

In one interaction, \( \mathcal{E} \) simulates what \( \mathcal{B} \) observes, that is, efficiently generates what \( \mathcal{B} \) observes.

\( \text{Sim:} \quad \text{Gen} \quad \text{pick} \quad (i, j) \quad \text{at random} \)
\[ \text{choose random} \quad c(i) \neq c(j) \]
\[ \text{choose} \quad c(u) \quad \text{so for all other nodes,} \quad u \neq i, u \neq j \]
\[ \text{put} \quad c(i) = (u) \quad \text{all in envelopes} \]
\[ \text{only open} \quad E_i, \quad E_j \]
Formal Definition of Zero Knowledge Proofs

Sound: if \( G \in L \rightarrow P_i \{ V \text{ accept} \} \leq \text{neg}(n) \).

Zero-knowledge: \( \exists \text{ poly-time Sim.} \forall G \in L \)

Sim (G) supr \( T \xrightarrow{\approx} T \) actual transcript

false Transcript.
How to get a lockable digital envelope?

Wrong: \[ \text{Wrong} \quad \text{Env}(a \oplus b) \approx \text{Env}_a(b) \]

open \[ \text{open by sending key } k \]

2 prop \{ hiding \}

bindig: \( \exists \) at most one \( b \) that we can open to.

Good: \[ h(b, \text{random}) = E \]

hidig because \( h \) is PRF
bindig because \( h \) is collision resistant.