CS 578 – Cryptography

Prof. Michael Backes

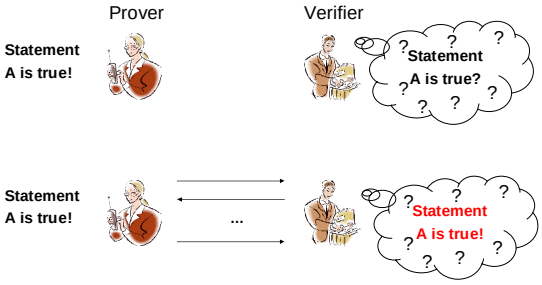
Zero-Knowledge Proofs

July 11, 2006

Administrative Announcements

- Q&A Session this Friday
- Last lecture: Next Tuesday (not exam-relevant, covers current research topics)
- Last quiz tomorrow (need 50% overall score)

Idea of Zero-Knowledge Proofs



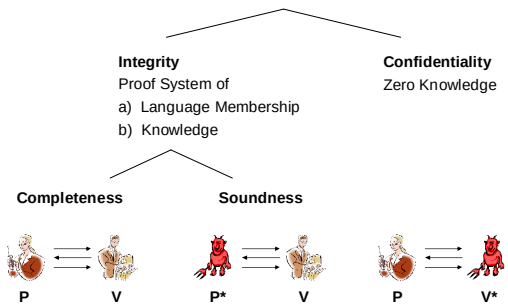
(Zero-Knowledge) Proof Systems

- Proof system: Two-party protocol between prover and verifier.
- Properties of a proof system:
  - Correctness: Prover can convince verifier of correct statements
  - Soundness: Not even a cheating prover can convince an honest verifier of wrong statements.
- Zero-Knowledge (property of the prover):
  - No (potentially dishonest) verifier learns “any new knowledge” from an honest prover.

Classes of Proof Systems

- Two main proof system classes:
  - Proofs of Language Membership:
    Given a language L and an element x, prove that $x \in L$.
  - Proofs of Knowledge:
    Given a binary relation R and an element x, prover “knows” an element w such that $(w,x) \in R$. w is called witness for x.
- Examples:
  - Language membership: Prove that a number is a QR.
  - Knowledge: Prove that one knows a square root.

Classes of Proof Systems



Ideas on Proofs of Language Membership

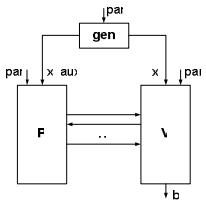
- First, why interactive proof at all?
- First reason: interactive proofs as extension of NP:
  - Every language L ∈ NP can be seen as interactive proof system:
    - Prover is computationally unrestricted and only sends one message, which is a witness of an NP-statement.
    - Verifier uses witness to decide correctness of the statement in poly-time.
- Second reason: cryptographic issues:
  - Prover typically has extra knowledge, e.g., knows a DLog or a root. Auxiliary information must be reflected in the definition.
  - Example: For proving that a number is a QR modulo N, one typically exploits that prover knows prime factors of N.

Definition Language Membership

- Definition (Interactive Proof System for Language Membership):
- Parameters: language L, security parameter(s) par.
- Two subprotocols:
  - Generation algorithm gen: On input security parameters par, outputs (x,aux) with x ∈ L.
  - Two-party protocol prove:
    - P (Prover) needs input (par, x, aux), does not make final output.
    - V (Verifier) needs input (par,x) and outputs b ∈ Bool. (b = 0 means verifier is convinced)
- Let $Exp_{P,V}((par,x,aux), (par,x)) = b$ denote the event that V outputs b.

Definition Language Membership

- Prover P and verifier V, and the subprotocols gen and prove:



Requirements

- Correct generation: For all valid parameters par, and (x,aux) ∈ [gen(par)], we have x ∈ L.
- Completeness: Correct prover P can always convince correct verifier V:
  For all valid par:
  $\Pr[Exp_{P,V}((par,x,aux), (par,x)) = 0; (x,aux) \leftarrow gen(par)] = 1$
- (Information-theoretic) Soundness: If x ∉ L, no cheating prover P* can convince a correct verifier V:
  For all probabilistic interactive algorithms P*, all valid par, and all x ∉ L and all aux:
  $$\Pr[Exp_{P^*,V}((par,x,aux), (par,x)) = 0] = 2^{-\sigma}$$
  where σ is a security parameter of par.

Computational Soundness

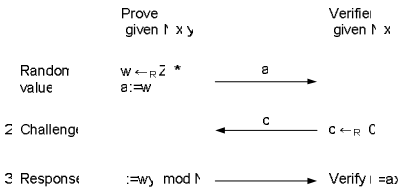
- Careful with computationally bounded cheating provers: how is x and aux chosen?
- Suitable option: Let them be chosen by the cheating prover, i.e., in polynomial-time, but (maybe) not with the correct generation algorithm.
- Definition (Computational Soundness): Given a proof system, except for the soundness, and just one security parameter n = par.
  Then the system has computational soundness if for all PPT algorithms gen*, P*, we have that
  $\Pr[x \notin L \wedge Exp_{P^*,V}((n,x,aux), (n,x)) = 0; (x,aux) \leftarrow gen^*(n)]$
  is negligible (in n).

Proving Quadratic Residuosity

- Proof of language membership: $L = \{(N,x) \mid x \in QR_N\}$
- Auxiliary information: aux = y such that $y^2 = x \bmod N$.
- Generation: Any algorithm gen that outputs pairs ((N,x),y) such that (N,x) ∈ L and $y^2 = x \bmod N$ is ok.
- Prove (iterated σ times):
  - Prover gets (N,x,y), chooses $w \leftarrow_R Z_N^*$, sets $a := w^2 \bmod N$, and outputs a.
  - Verifier receives a, chooses $c \leftarrow_R \{0,1\}$, and outputs c.
  - Prover receives c, computes $r := w \cdot y^c \bmod N$, and outputs r
  - Verifier receives r and outputs true iff $r^2 = a \cdot x^c$.

Proving Quadratic Residuosity

- Prove protocol again visually (one round):



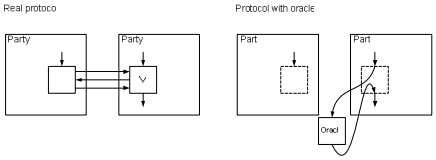
Proving Quadratic Residuosity

- Theorem: The quadratic residuosity scheme (of the last slide) is an interactive proof of language membership system.

  [proof on the board]

Defining Zero-Knowledge

- Intended to capture that verifier learn no new knowledge at all.
- NOT intended to only capture keeping a specific secret hidden!
- Why such strong goal? Modular proofs…



Defining Zero-Knowledge

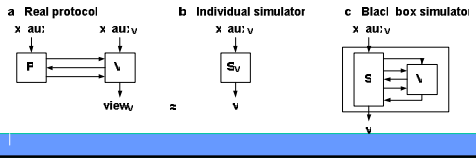
- Idea to define zero-knowledge:

  Something is no new knowledge if one could easily have computed it alone.

- Example (roots of quadratic residues):
  - Honest verifier gives random residues and expects square roots modulo N.
  - Then only someone knowing the factors of N can compute square roots.
  - But the view, as seen afterwards, only consists of pairs (x,y) with $y^2 = x \bmod N$, which one can compute oneself.
  - Not yet zero-knowledge proof since verifier assumed honest …

Defining Zero-Knowledge

- Idea more formally: For every verifier V*, there exists a simulator S_{V^*} that, without given secrets, computes views that are indistinguishable from real views of V*.
- Often black-box construction: One S that uses V* as a blackbox, including resetting V* to a previous state.
- Allow additionally arbitrary aux_{V^*} (previous knowledge, parallel sessions, etc.), strengthens the definition



Indistinguishability

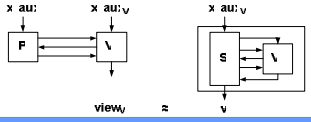
- Indistinguishability defined on ensembles of probability distributions $(Prob_i)_{i \in I}$ and $(Prob^*_i)_{i \in I}$.
- Definition (Indistinguishability): Two ensembles of probability distributions $(Prob_i)_{i \in I}$ and $(Prob^*_i)_{i \in I}$ are
  - perfectly indistinguishable if they are equal.
  - computationally indistinguishable if for all PPT algorithms D (the distinguisher), we have that

    $$|\Pr(D(i,v)=1; v \leftarrow_R Prob_i) - \Pr(D(i,v)=1; v \leftarrow_R Prob^*_i)|$$

    is negligible.

Defining Zero-Knowledge

- Definition (Zero-Knowledge). Let an interactive proof system with generation algorithm be given. It is called
  - perfectly zero-knowledge if for all PPT interactive algorithms V* (cheating verifier), there exists an (expected) PPT algorithm S_{V^*} (the simulator), such that the following two ensembles are perfectly indistinguishable:
    - The index set I is the set of tuples (par,x,aux,aux_{V^*}) with (x,aux) ∈ [gen(par)].
    - The two ensembles for (par,x,aux,aux_{V^*}) ∈ I are V*'s view in Exp_{P,V^*}((par,x,aux),(par,x,aux_{V^*})) and S_{V^*}(par,x,aux_{V^*}).

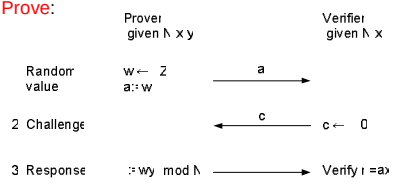


Defining Zero-Knowledge

- Definition (Zero-Knowledge). Let an interactive proof system with generation algorithm be given. It is called
  - perfectly zero-knowledge if for all PPT interactive algorithms V* (cheating verifier), there exists a PPT algorithm S_{V^*} (the simulator), such that the following two ensembles are perfectly indistinguishable:
    - The index set I is the set of tuples (par,x,aux,aux_{V^*}) with (x,aux) ∈ [gen(par)].
    - The two ensembles for (par,x,aux,aux_{V^*}) ∈ I are V*'s view in Exp_{P,V^*}((par,x,aux),(par,x,aux_{V^*})) and S_{V^*}(par,x,aux_{V^*}).
  - computationally zero-knowledge if the same holds for "suitably generated" aux_{V^*}: For all PPT algorithms G, define (x,aux) ←gen(par), aux_{V^*}← G(x,aux), and require that the two ensembles defined above are computationally indistinguishable.

Proving Quadratic Residuosity

- Proof of language membership: L = {(N,x) | x ∈ QR_N}
- Auxiliary information: aux = y such that y^2 = x mod N.
- Generation: Any algorithm gen that outputs pairs ((N,x),y) such that (N,x) ∈ L and y^2 = x mod N is ok.
- Prove:



Proving Quadratic Residuosity

- Theorem: The quadratic residuosity scheme is a perfectly zero-knowledge proof (of language membership) system.
- Need to construct a simulator that produces correctly distributed views (a,c,r,b) of (potentially dishonest) verifier V*.

Proving Quadratic Residuosity

- Construction of a simulator for an honest verifier V:
  - Choose $r \leftarrow_R Z_N^*$.
  - Choose $c \leftarrow_R \{0,1\}$.
  - Solve verification equation for a, yielding $a := r^2 / x^c$.
- Construction of a simulator for a dishonest verifier V*:
  - Choose $r \leftarrow_R Z_N^*$.
  - Choose $c \leftarrow_R \{0,1\}$.
  - Solve verification equation for a, yielding $a := r^2 / x^c$.
  - Start V*, send a to it, wait for challenge c*.
  - If $c = c^*$, send r to V*, get its final output b, and output (a,c,r,b). If $c \neq c^*$, rewind V* and start with a new value of r.
- Still to show: simulation for multiple rounds by induction, correct distribution for a,c,r,b …

On Proofs of Knowledge

- First, how to define that a machine “knows” a value?
  - Written on its tape?
    ⇒ Might be stored in reverse order …
  - In an arbitrary encoding?
    ⇒ Might be stored as N = pq …
  - Successfully computable in polynomial-time?
    ⇒ Not defined for individual input/output pairs, only for global function …

On Proofs of Knowledge

- Definition idea using knowledge extractor K:
  - K can extract witness w by using P* as a block-box.

  - Intuition: Extractor should be successful for all provers with not-negligible success probability.
  - Problem: The smaller this success probability, the more difficult extraction gets
    → grant K running time inversely proportional to the success probability of P*.

Definition Proof of Knowledge

- Definition (Interactive Proof-of-Knowledge System):
  - Parameters: binary relation R, security parameter(s) par. Let $L_R := \{x \mid \exists w: (w,x) \in R\}$.
  - Roles P and V, and subprotocols gen and prove as before
  - Let again $Exp_{P,V}((par,x,aux), (par,x)) = b$ denote the event that V outputs b.

Requirements

- Correct generation: For all valid parameters par, and (x,aux) $\in$ [gen(par)], we have (x, aux) $\in$ R.
- Completeness: Correct prover P with input a witness w (=aux) for input x can always convince correct verifier V:
  For all valid par:
  $P[Exp_{P,V}((par,x,aux), (par,x)) = 0; (x,aux) \leftarrow gen(par)] = 1$
- Weak Soundness: There exists a PPT machine K (knowledge extractor), that can interact with and reset a machine P*, and a polynomial pol such that:
  For all PPT P*, all aux, and all $x \in L_R$, we have that on input (par, x), and interacting with P*, K should output a witness w for x in expected time

$$\frac{pol(n)}{\Pr[Exp_{P^*,V}((par,x,aux),(par,x)) = 0]}$$

Weak and Strong Soundness

- Only weak soundness so far: soundness only for x satisfying $x \in L_R$.
- Did not care if prover could prove knowledge of the prime factors of a prime, a root of a QNR, etc.
- Strong soundness = weak soundness + proof of language membership
- We'll stop here for now…

Application Areas

- Identification schemes: Prove that one owns a secret key corresponding to a public key
- Signatures: Fiat-Shamir paradigm: mingle message into challenge (no details here)
- Many more applications in higher level protocols: commitment schemes, credentials used in access control etc.