Algorithms of Information Security: Cryptographic Protocols I

Faculty of Information Technology
Czech Technical University in Prague

October 20, 2021

Identification objectives and applications

- The general setting for an identification protocol involves a *prover* or *claimant A* and *verifer B.*
- The verifer is presented with, or presumes beforehand, the purpoted identity of the claimant.
- The goal is to corroborate that the identity of the claimant is indeed A, i.e., to provide entity authentication.

Definition

Entity authentication is the process whereby one party is assured (through acquisition of corroborative evidence) of the identity of a second party involved in a protocol, and that the second has actually participated (i.e., is active at, or immediately prior to, the time the evidence is acquired).

Objectives of identification protocols

- From the point of view of the verifier, the outcome of an entity authentication protocol is either *acceptance* of the claimant's identity as authentic (completion with acceptance), or *termination without acceptance* (rejection).
- More specifically, the objectives of an identification protocol include the following.
  - In the case of honest parties A and B, A is able to successfully authenticate itself to B, i.e., B will complete the protocol having accepted A's identity.
  - *(transferability)* B cannot reuse an identification exchange with A so as to successfully impersonate A to a third party C.

Objectives of identification protocols

- *(impersonation)* The probability is negligible that any party C distinct from A, carrying out the protocol and playing the role of A, can cause B to complete and accept A's identity. Here *negligible* typically means "is so small that it is not of practical significance"; the precise definition depends on the application.
- The previous points remain true even if:
  - (polynomially) large number of previous authentications between A and B have been observed;
  - the adversary C has participated in previous protocol executions with either or both A and B;
  - and multiple instances of the protocol, possibly initiated by C, may be run simultaneously.

Basis of identification

- An identification (or entity authentication) protocol is a "real-time" process in the sense that it provides an assurance that the party being authenticated is operational at the time of protocol execution - that party is taking part, having carried out some action since the start of the protocol execution.
- Identification protocols provide assurances only at the particular instant in time of successful protocol completion.
- Entity authentication techniques may be divided into three main categories, depending on which of the following the security is based:
  - *something known.* Examples include standard passwords (sometimes used to derive a symmetric key), Personal Identification Numbers (PINs), and the secret or private keys whose knowledge is demonstrated in challenge-response protocols.

Basis of identification

- *something possessed.* This is typically a physical accessory, resembling a passport in function. Examples include magnetic-striped cards, chipcards (plastic cards the size of credit cards, containing an embedded microprocessor or integrated circuit; also called smart cards or IC cards), and hand-held customized calculators (password generators) which provide time-variant passwords.
- *something inherent (to a human individual).* This category includes methods which make use of human physical characteristics and involuntary actions (biometrics), such as handwritten signatures, fingerprints, voice, retinal patterns, hand geometries, and dynamic keyboarding characteristics. These techniques are typically non-cryptographic.

Properties of identification protocols

Identification protocols may have many properties. Properties of interest to users include:

- *reciprocity of identification.* Either one or both parties may corroborate their identities to the other, providing, respectively, *unilateral* or *mutual* identification. Some techniques, such as fixed-password schemes, may be susceptible to an entity posing as a verifier simply in order to capture a claimant's password.
- *computational efficiency.* The number of operations required to execute a protocol.
- *communication efficiency.* This includes the number of passes (message exchanges) and the bandwidth required (total number of bits transmitted).

Properties of identification protocols

More subtle properties include:

- *real-time involvement of a third party (if any).* Examples of third parties include an on-line trusted third party to distribute common symmetric keys to communicating entities for authentication purposes; and an on-line (untrusted) directory service for distributing public-key certificates, supported by an off-line certification authority
- *nature of trust required in a third party (if any).* Examples include trusting a third party to correctly authenticate and bind an entity's name to a public key; and trusting a third party with knowledge of an entity's private key.
- *nature of security guarantees.* Examples include provable security and zero-knowledge properties.
- *storage of secrets.* This includes the location and method used (e.g., software only, local disks, hardware tokens, etc.) to store critical keying material.

Model

- Alice wishes to prove to Bob her identity in order to access a resource, obtain a service etc.
- Bob may ask the following:
  - Who are you? (prove that you're Alice)
  - Who is Alice?
- Eve wishes to impersonate Alice:
  - One time impersonation
  - Full impersonation (identity theft)

Identification Scenarios

- Local identification
  - Human authenticator
  - Device
- Remote identification
  - Human authenticator
  - Corporate environment (e.g. LAN)
  - E-commerce environment
  - Cable TV/Satellite: Pay-per-view; subscription verification
  - Remote login or e-mail from an internet cafe

Initial Authentication

- The problem: how does Alice initially convince anyone that she's Alice?
- The solution must often involve a "real-world" type of authentication – id card, driver's license etc.
- Errors due to the human factor are numerous (example – the Microsoft-Verisign fiasco).
- Even in scenarios where OK for Alice to be whoever she claims she is, may want to at least make sure Alice is human (implemented, e.g. for new users in Yahoo mail).

One-time passwords based on one-way functions (Lamport's scheme)

- In Lamport's one-time password scheme, the user begins with a secret w. A one-way function (OWF) H is used to define the password sequence: $w, H(w), H(H(w)), \ldots, H^t(w)$.
- The password for the i^{th} identification session, $1 \leq i \leq t$, is defined to be $w_i = H^{t-i}(w)$.

Lamport's OWF-based one-time passwords

Algorithm 1 Lamport's OWF-based one-time passwords

SUMMARY. A identifies itself to B using one-time passwords from a sequence.

1. *One-time setup.*
   - User A begins with a secret w. Let H be a one-way function.
   - A constant t is fixed (e.g.,$t = 100$ or 1000), defining the number of identifications to be allowed. (The system is thereafter restarted with a new w.)
   - A transfers (the *initial shared secret*) $w_0 = H^t(w)$, in a manner guaranteeing its authenticity, to the system B. B initializes its counter for A to $i_A = 1$.

Lamport's OWF-based one-time passwords

Algorithm 1 Lamport's OWF-based one-time passwords

2. *Protocol messages.* The i_{th} identification, $1 \leq i \leq t$, proceeds as follows:

$$A \rightarrow B : A, i, w_i (= H^{t-i}(w)) \ (1)$$

Here $A \rightarrow B : X$ denotes A sending the message X to B.

3. *Protocol actions.* To identify itself for session i, A does the following.

- A's equipment computes $w_i = H^{t-i}(w)$ (easily done either from w itself, or from an appropriate intermediate value saved during the computation of $H^t(w)$ initially), and transmits (1) to B.
- B checks that $i = i_A$, and that the received password w_i satisfies: $H(w_i) = w_{i-1}$. If both checks succeed, B accepts the password, sets $i_A \leftarrow i_A + 1$, and saves w_i for the next session verification.

Zero-knowledge identification protocols

- A disadvantage of simple password protocols is that when a claimant A (called a *prover* in the context of zero-knowledge protocols) gives the *verifier* B her password, B can thereafter impersonate A.
- Challenge-response protocols improve on this: A responds to B's challenge to demonstrate knowledge of A's secret in a time-variant manner, providing information not directly reusable by B.
- This might nonetheless reveal some partial information about the claimant's secret; an adversarial verifier might also be able to strategically select challenges to obtain responses providing such information.

Zero-knowledge identification protocols

- Zero-knowledge (ZK) protocols are designed to address these concerns, by allowing a prover to demonstrate knowledge of a secret while revealing no information whatsoever (beyond what the verifier was able to deduce prior to the protocol run) of use to the verifier in conveying this demonstration of knowledge to others.
- The point is that only a single bit of information need be conveyed - namely, that the prover actually does know the secret.

Interactive proof systems and zero-knowledge protocols

- The ZK protocols are instances of *interactive proof systems,* wherein a prover and verifier exchange multiple messages (challenges and responses), typically dependent on random numbers (ideally: the outcomes of fair coin tosses) which they may keep secret.
- The prover's objective is to convince (prove to) the verifier the truth of an assertion, e.g.,claimed knowledge of a secret.
- The verifier either accepts or rejects the proof.
- The traditional mathematical notion of a proof, however, is altered to an interactive game wherein proofs are probabilistic rather than absolute; a proof in this context need be correct only with bounded probability, albeit possibly arbitrarily close to 1. For this reason, an interactive proof is sometimes called a *proof by protocol.*

Interactive proof systems and zero-knowledge protocols

- Interactive proofs used for identification may be formulated as proofs of knowledge. A possesses some secrets, and attempts to convince B it has knowledge of the secret s by correctly responding to queries (involving publicly known inputs and agreed upon functions) which require knowledge of s to answer.
- Note that proving knowledge of s differs from proving that such s exists – for example, proving knowledge of the prime factors of n differs from proving that n is composite.

- An interactive proof is said to be a *proof of knowledge* if it has both the properties of completeness and soundness. Completeness may be viewed as the customary requirement that a protocol functions properly given honest participants.

Definition

(completeness property) An interactive proof (protocol) is complete if, given an honest prover and an honest verifier, the protocol succeeds with overwhelming probability (i.e., the verifier accepts the prover's claim). The definition of overwhelming depends on the application, but generally implies that the probability of failure is not of practical significance.

Interactive proof systems and zero-knowledge protocols

Definition

(soundness property) An interactive proof (protocol) is sound if there exists an expected polynomial-time algorithm M with the following property: if a dishonest prover (impersonating A) can with non-negligible probability successfully execute the protocol with B, then M can be used to extract from this prover knowledge (essentially equivalent to A's secret) which with overwhelming probability allows successful subsequent protocol executions.

- An alternate explanation of the condition in last definition is as follows: the prover's secrets together with public data satisfies some polynomial-time predicate, and another solution of this predicate (possibly the same) can be extracted, allowing successful execution of subsequent protocol instances.

Interactive proof systems and zero-knowledge protocols

Definition

(zero-knowledge property) A protocol which is a proof of knowledge has the zero-knowledge property if it is simulatable in the following sense: there exists an expected polynomial-time algorithm (simulator) which can produce, upon input of the assertion(s) to be proven but without interacting with the real prover, transcripts indistinguishable from those resulting from interaction with the real prover.

- The zero-knowledge property implies that a prover executing the protocol (even when interacting with a malicious verifier) does not release any information (about its secret knowledge) not otherwise computable in polynomial time from public information alone.

Comments on zero-knowledge vs. other asymmetric protocols

The following observations may be made regarding zero-knowledge (ZK) techniques, as compared with other public-key (PK) techniques.

- *no degradation with usage:* protocols proven to have the ZK property do not suffer degradation of security with repeated use, and resist chosen-text attacks. This is perhaps the most appealing practical feature of ZK techniques.
- *encryption avoided:* many ZK techniques avoid use of explicit encryption algorithms.
- *efficiency:* while some ZK-based techniques are extremely efficient, protocols which formally have the zero-knowledge property typically have higher communications and/or computational overheads than PK protocols which do not.

Fiat-Shamir identification protocol (basic version)

- The objective for A is to identify itself by proving knowledge of a secret s (associated with A through authentic public data) to any verifier B, without revealing any information about s not known or computable by B prior to execution of the protocol.
- The security relies on the difficulty of extracting square roots modulo large composite integers n of unknown factorization, which is equivalent to that of factoring n.

Fiat-Shamir identification protocol (basic version)

Algorithm 2 Fiat-Shamir identification protocol (basic version)

SUMMARY. A proves knowledge of s to B in t executions of a 3-pass protocol.

1. *One-time setup.*
   - A trusted center T selects and publishes an RSA-like modulus $n = pq$ but keeps primes p and q secret.
   - Each claimant A selects a secret s coprime to $n, 1 \leq s \leq n-1$, computes $v = s^2 \mod n$, and registers v with T as its public key.
2. *Protocol messages.* Each of t rounds has three messages with form as follows.
   - $A \rightarrow B : x = r^2 \mod n$
   - $A \leftarrow B : e \in \{0, 1\}$
   - $A \rightarrow B : y = r \cdot s^e \mod n$

Fiat-Shamir identification protocol (basic version)

Algorithm 2 Fiat-Shamir identification protocol (basic version)

3. *Protocol actions.* The following steps are iterated t times (sequentially and independently). B accepts the proof if all t rounds succeed.
   - A chooses a random (commitment) $r, 1 \leq r \leq n-1$, and sends (the witness) $x = r^2 \bmod n$ to B.
   - B randomly selects a (challenge) bit $e = 0$ or $e = 1$, and sends e to A.
   - A computes and sends to B (the response) y, either $y = r$ (if $e = 0$) or $y = rs \bmod n$ (if $e = 1$).
   - B rejects the proof if $y = 0$, and otherwise accepts upon verifying $y^2 = x \cdot v^e \bmod n$. (Depending on $e, y^2 = x$ or $y^2 = xv \bmod n$, since $v = s^2 \bmod n$. Note that checking for $y = 0$ precludes the case $r = 0$.)

Feige-Fiat-Shamir identification protocol

- The basic version of the Fiat-Shamir protocol can be generalized, and the Feige-Fiat-Shamir (FFS) identification protocol is a minor variation of such a generalization.
- The FFS protocol involves an entity identifying itself by proving knowledge of a secret using a zero-knowledge proof; the protocol reveals no partial information whatsoever regarding the secret identification value(s) of A.
- It requires limited computation (a small fraction of that required by RSA), and is thus well-suited for applications with low-power processors (e.g., 8-bit chipcard microprocessors).