EIDMA-Stieltjesweek Graduate Course

Σ-protocols

September 23, 2003

Berry Schoenmakers
TU Eindhoven

berry@win.tue.nl

http://www.win.tue.nl/~berry/

1. Definitions

1.1. Σ-protocols

Let $R = \{(v, w)\}$ be a binary relation. (It is assumed that for some given polynomial p that $|w| \leq p(|v|)$ for all $(v, w) \in R$.) Here, v denotes the common input to prover and verifier, and w denotes a witness, which is the private input to the prover. Let $L_R = \{v | \exists w : (v, w) \in R\}$.

A Σ-protocol for relation R is of the following form:

$$
\begin{array}{cccc}
\textsf{Prover} & & \textsf{Verifier} & \\
\text{(random tape } s_P) & & \text{(random tape } s_V) & \\
((v, w) \in R) & & (v \in L_R) & \\
a := \alpha(v, w, s_P) & & & \\
 & \xrightarrow{\quad a \quad} & & \\
 & & c := \gamma(s_V) & \begin{array}{c}\text{public}\\ \text{coin}\end{array} \\
 & \xleftarrow{\quad c \quad} & & \\
r := \rho(v, w, c, s_P) & & & \\
 & \xrightarrow{\quad r \quad} & & \\
 & & \phi(v, a, c, r)? &
\end{array}
$$

1.2. Security properties for Σ-protocols

Completeness: if P and V follow the protocol, the verifier always accepts.

Special soundness: for any v and any pair of accepting conversations (a, c, r) and (a, c', r') with $c \neq c'$ one can efficiently compute witness w such that $(v, w) \in R$.

Special honest-verifier zero-knowledge: there exists a p.p.t. machine S (simulator) which for any v and c produces conversations (a, c, r) with the same probability distribution as conversations between the honest P and V with common input v and challenge c.

Note that a cheating prover succeeds with probability at most $1/q$, where q denotes the cardinality of the challenge space $\gamma(\cdot)$.

2. Schnorr-based examples

2.1. Schnorr's protocol

$$
\begin{array}{ccc}
\textsf{Prover} & & \textsf{Verifier} \\
(x = \log_g h) & & \\
 & & \\
u \in_R \mathbb{Z}_q & & \\
a := g^u & & \\
 & \xrightarrow{\qquad a \qquad} & \\
 & & c \in_R \mathbb{Z}_q \\
 & \xleftarrow{\qquad c \qquad} & \\
r := u + cx & & \\
 & \xrightarrow{\qquad r \qquad} & \\
 & & g^r \stackrel{?}{=} a h^c
\end{array}
$$

2.2. Parallel composition

Running two instances of Schnorr's protocol in parallel, for the *same* public key h, results in a Σ-protocol with a larger challenge length.

$$
\begin{array}{ccc}
\textsf{Prover} & & \textsf{Verifier} \\
(x = \log_g h) & & \\
& & \\
u_1, u_2 \in_R \mathbb{Z}_q & & \\
a_1 := g^{u_1} & & \\
a_2 := g^{u_2} & & \\
& \xrightarrow{\quad a_1, a_2 \quad} & \\
& & c_1, c_2 \in_R \mathbb{Z}_q \\
& \xleftarrow{\quad c_1, c_2 \quad} & \\
r_1 := u_1 + c_1 x & & \\
r_2 := u_2 + c_2 x & & \\
& \xrightarrow{\quad r_1, r_2 \quad} & \\
& & g^{r_1} \stackrel{?}{=} a_1 h^{c_1} \\
& & g^{r_2} \stackrel{?}{=} a_2 h^{c_2}
\end{array}
$$

2.3. AND composition

Given two public keys h_1, h_2, one proves knowledge of $\log_g h_1$ and $\log_g h_2$, by running two instances of the Schnorr proof in parallel, using a *common* challenge.

$$
\begin{array}{ccc}
\textsf{Prover} & & \textsf{Verifier} \\
((x_1, x_2) = (\log_g h_1, \log_g h_2)) & & \\
& & \\
u_1, u_2 \in_R \mathbb{Z}_q & & \\
a_1 := g^{u_1} & & \\
a_2 := g^{u_2} & & \\
& \xrightarrow{\quad a_1, a_2 \quad} & \\
& & c \in_R \mathbb{Z}_q \\
& \xleftarrow{\quad c \quad} & \\
r_1 := u_1 + cx_1 & & \\
r_2 := u_2 + cx_2 & & \\
& \xrightarrow{\quad r_1, r_2 \quad} & \\
& & g^{r_1} \stackrel{?}{=} a_1 h_1^c \\
& & g^{r_2} \stackrel{?}{=} a_2 h_2^c
\end{array}
$$

2.4. OR composition

It turns out that there is a proof of knowledge of (at least) one of $x_1 = \log_g h_1$ and $x_2 = \log_g h_2$ of the *same* complexity as an AND proof.

We let the prover do a proof of knowledge for both $\log_g h_1$ and $\log_g h_2$ in parallel but giving the prover one degree of freedom in choosing the two challenges for these proofs. This allows the prover to *cheat* in one of the two proofs.

Suppose the prover knows x_1 but does not know x_2. The prover will then do a real proof of knowledge for $\log_g h_1$, and use the honest-verifier zero-knowledge property of the Schnorr protocol to create a simulated proof for $\log_g h_2$.

$$
\begin{array}{l|lcl}
\multicolumn{2}{c}{\textsf{Prover}} & & \textsf{Verifier} \\
(\text{using } x_2 = \log_g h_2) & (\text{using } x_1 = \log_g h_1) & & \\
r_1, c_1, u_2 \in_R \mathbb{Z}_q & r_2, c_2, u_1 \in_R \mathbb{Z}_q & & \\
a_1 := g^{r_1} h_1^{-c_1} & a_1 := g^{u_1} & & \\
a_2 := g^{u_2} & a_2 := g^{r_2} h_2^{-c_2} & & \\
 & & \xrightarrow{\quad a_1, a_2 \quad} & \\
 & & & c \in_R \mathbb{Z}_q \\
 & & \xleftarrow{\quad c \quad} & \\
c_2 := c - c_1 & c_1 := c - c_2 & & \\
r_2 := u_2 + c_2 x_2 & r_1 := u_1 + c_1 x_1 & & \\
 & & \xrightarrow{\quad c_1, c_2, r_1, r_2 \quad} & \\
 & & & c_1 + c_2 \stackrel{?}{=} c \\
 & & & g^{r_1} \stackrel{?}{=} a_1 h_1^{c_1} \\
 & & & g^{r_2} \stackrel{?}{=} a_2 h_2^{c_2}
\end{array}
$$

2.5. Equality of Discrete Logs

Given two public keys $h_1 = g_1^x, h_2 = g_2^x$, one proves knowledge of $x = \log_{g_1} h_1 = \log_{g_2} h_2$, by running two instances of the Schnorr proof in parallel, using a *common* random choice, a *common* challenge and a *common* response.

$$
\begin{array}{ccc}
\textsf{Prover} & & \textsf{Verifier} \\
(x = \log_{g_1} h_1 = \log_{g_2} h_2) & & \\
 & & \\
u \in_R \mathbb{Z}_q & & \\
a_1 := g_1^u & & \\
a_2 := g_2^u & & \\
 & \xrightarrow{\quad a_1, a_2 \quad} & \\
 & & c \in_R \mathbb{Z}_q \\
 & \xleftarrow{\quad c \quad} & \\
r := u + cx & & \\
 & \xrightarrow{\quad r \quad} & \\
 & & g_1^r \stackrel{?}{=} a_1 h_1^c \\
 & & g_2^r \stackrel{?}{=} a_2 h_2^c
\end{array}
$$

2.6. Schnorr signatures

Schnorr signatures are obtained by applying the Fiat-Shamir heuristic to Schnorr's protocol: compute the challenge as a hash $\mathcal{H}(\cdot)$ of the message m and the value a.

Signer		Receiver
$(x = \log_g h)$		
$u \in_R \mathbb{Z}_q$		
$a := g^u$		
$c := \mathcal{H}(m, a)$		
$r := u + xc$	$\xrightarrow{\quad a, r \quad}$	$c := \mathcal{H}(m, a)$
		$g^r \stackrel{?}{=} ah^c$

(As an optimization, one may send c instead of a, as the bit-length of c may be much smaller than the bit-length of a. The receiver computes $a := g^r h^{-c}$ and accepts if $c = \mathcal{H}(m, a)$.)
The Fiat-Shamir technique for converting Σ-protocols into signature schemes is provably secure in the so-called random oracle model.

3. Exercises

Exercise 1 *Prove the special soundness of the OR composition for the Schnorr protocol.*

Exercise 2 *Let g, h denote generators of a group G of large prime order q such that $\log_g h$ is unknown to anyone. Let $B = g^x h^y$ denote the common input to prover and verifier, where $x, y \in \mathbb{Z}_q$ is private input to the prover. For each of the following predicates $\psi(x, y)$, design a Σ-protocol that proves knowledge of $x, y \in \mathbb{Z}_q$ such that $B = g^x h^y$ and $\psi(x, y)$ holds:*

a. $\psi(x, y) \equiv \mathrm{true}$;

b. $\psi(x, y) \equiv x = y$;

c. $\psi(x, y) \equiv \alpha x + \beta y = \gamma$ for given $\alpha, \beta, \gamma \in \mathbb{Z}_q$;

d. $\psi(x, y) \equiv x \in \{0, 1\}$;

e. $\psi(x, y) \equiv x \in \{0, \ldots, 2^k - 1\}$, where k is a fixed integer, $1 \leq k \leq \lfloor \log_2 q \rfloor$;

f. $\psi(x, y) \equiv x \neq 0$;

g. $\psi(x, y) \equiv \exists a \in \mathbb{Z}_q : x = a^2$;