CSC 591 Cryptography	11/14/18

Lecture 22 – Zero-Knowledge Proofs

Lecturer: Alessandra Scafuro *Scribe: Abida Haque*

Zero-Knowledge Proofs

Main takeaways:

1. Proofs are interactive.
2. Verifier does not learn anything but that the theorem is true.

Traditionally, proofs consist of a prover P who outputs a proof. Some verifier V reads the proof and decides if it is correct. Instead, here we allow V to *interact* with P, communicating with messages. Both P and V can flip coins (choose randomness).

For a protocol Π, let $V(x)$ represent the *view* of the conversation from the verifier's point of view on input x. $V(x)$ consists of:

- messages sent between P and V.
- The randomness chosen by V.

In order for the protocol to be zero-knowledge $V(x)$ needs to be distributed the same as one V could have generated himself. Intuitively, the verifier can come up with the proof himself.

Definition

Definition 1 (Zero-Knowledge Proof) *A pair of algorithms (interactive Turing Machines) $\langle P, V\rangle$ is a zero-knowledge proof for a language L if it satisfies the following properties:*

1. *Soundness: For all malicious prover P^*, if the theorem is false then P^* convinces V with only negligible probability. Formally, $\forall x$, $x \notin L$*

$$Pr[\langle P^*, V\rangle \text{ is accepting}] \leq negl(n).$$

2. *Zero-Knowledge: Suppose there is a PPT machine Sim which knows V and on input x outputs values of the form $V(x)$.*

   Formally we define this with a simulator, who has no witness. Let out *represent the distribution of outputs of a machine on some inputs. There exists a PPT algorithm Sim such that $\forall x \in L$*

$$\mathsf{out}(Sim(x)) \approx \mathsf{out}(\langle P(w, x), V(x)\rangle).$$

Assumption

We use the same assumptions as ElGamal, which is defined using the hardness of the DDH problem. ElGamal uses a cyclic group $\mathbb{Z}_q$, which is of order q and has a generator g.

Recall ElGamal:

$$
\begin{array}{lcl}
\textbf{Alice} & & \textbf{Bob} \\
\hline
x \leftarrow_{\$} \mathbb{Z}_q & & \\
X \leftarrow g^x & & \\
 & \xrightarrow{\quad \mathbb{G}, q, g, X \quad} & \\
 & & ENC : m \in \mathbb{G}, y \leftarrow_{\$} \mathbb{Z}_q \\
 & & c_1 = g^y, c_2 = h^y \cdot m \\
 & \xleftarrow{\quad c = (c_1, c_2) \quad} & \\
DEC : \hat{m} = \dfrac{c_2}{c_1^x} & &
\end{array}
$$

Scheme

ElGamal Secret Key / Schnorr Σ-Protocol

$$
\begin{array}{lcl}
\textbf{Prover} & & \textbf{Verifier} \\
\hline
r \leftarrow_{\$} \mathbb{Z}_q & & \\
A = g^r & & \\
 & \xrightarrow{\quad A \quad} & \\
 & & c \leftarrow_{\$} \mathbb{Z}_q \\
 & \xleftarrow{\quad c \quad} & \\
z = yc + r & \xrightarrow{\quad z \quad} & \\
 & & g^z \stackrel{?}{=} X^c A
\end{array}
$$

Security Proof

Prove knowledge of an Elgamal Secret Key. We need to prove completeness, soundness (aka proof of knowledge), and zero-knowledge.

Completeness (Correctness)

$$
\begin{align}
z &= yc + r \tag{1}\\
g^z &= Y^c A \tag{2}\\
g^z &= g^{yc} g^r \tag{3}\\
g^z &= g^{yc+r} \tag{4}
\end{align}
$$

Soundness (Proof of Knowledge) If P^* convinces V then P^* must know y.

Assume P^* convinces V. In other words, P^* produces a good transcript.

Observations:

1. P^* is an interactive TM.

2. P^* generates a good transcript: (A, c, z) is accepting.

We rewind P^* to right after he's given A. We give a new value $c' \neq c$ to P^*. Then we get two accepting transcripts:

$$g^z = g^{yc+r} \text{ and } g^{z'} = g^{yc'+r}$$

$$
\begin{align}
g^z &= g^{yc+r} \tag{5}\\
g^{z'} &= g^{yc'+r} \implies \tag{6}\\
g^{z-z'} &= g^{yc+r-yc'-r} \implies \tag{7}\\
g^{z-z'} &= g^{y(c-c')} \implies \tag{8}\\
\frac{z-z'}{c-c'} &= y \tag{9}
\end{align}
$$

If I have two transcripts, then I can compute the secret y.

Zero-Knowledge. Simulator can generate an accepting transcript, knowing only $\mathbb{Z}_q$, q, g, and X (prover's public key) without the secret y.

$$
\begin{align}
z &\leftarrow_{\$} \mathbb{Z}_q \tag{10}\\
c &\leftarrow_{\$} \mathbb{Z}_q \tag{11}\\
A &= \frac{g^z}{Y^c} = g^{z-yc} \tag{12}
\end{align}
$$