Lecture 28

Lecturer: Jonathan Katz *Scribe(s): Nagaraj Anthapadmanabhan, Alvaro Cardenas*

1 Introduction

In a previous class (Lecture 25), we showed how to construct an identification scheme which is secure against a *passive adversary* using an *Honest-Verifier Zero-Knowledge Proof of Knowledge* (HVZK-PoK). We also showed that it is possible to construct an Identification Scheme secure against an *active adversary* using a *Witness Indistinguishable Proof of Knowledge* (WI-PoK). In this lecture, we will construct efficient proof systems with these properties, and thus efficient identification schemes, based on the discrete logarithm assumption. We will also see how the resulting identification schemes can be converted into signature schemes.

2 Security Against Passive Adversaries

We refer to Lecture 25 for the definitions of identification schemes and their security against passive adversaries. We also refer there for a proof that an identification scheme can be constructed using any one-way function f and an HVZK-PoK of a value x such that $f(x) = y$ (where y is included in the prover's public key). Here, we merely show a specific (efficient) HVZK-PoK for the particular case when the one-way function f is the discrete exponentiation function (and the hardness of inverting f is therefore equivalent to the discrete logarithm assumption that we have seen previously). To establish some notation, let $\mathbb{G}$ be a cyclic group of prime order q and let g be a fixed generator of $\mathbb{G}$. We will assume that $\mathbb{G}, g$, and q are publicly known. If $y = g^x$ then we let $x \stackrel{\text{def}}{=} \log_g y$.

The setup is as follows (cf. Lecture 25): the verifier knows the prover's public key y, while the prover has a secret key x such that $y = g^x$. The following protocol due to Schnorr [4] allows the prover to prove to the verifier that he indeed knows x:

$$\begin{array}{ccc}
 & \mathbb{G}, q, g, y & \\
\underline{\mathcal{P}(x)} & & \underline{\mathcal{V}} \\
r \leftarrow \mathbb{Z}_q & \xrightarrow{\quad A = g^r \quad} & \\
 & \xleftarrow{\quad c \quad} & c \leftarrow \mathbb{Z}_q \\
 & \xrightarrow{\quad b = (cx + r) \bmod q \quad} & y^c A \stackrel{?}{=} g^b
\end{array}$$

Let us first show that the above scheme is correct. If both players act honestly and $y = g^x$, then we have:

$$g^b = g^{cx+r} = g^{cx} g^r = y^c A,$$

where we use the fact that the order of the group is q, and so exponents are reduced modulo q (i.e., $g^{cx+r} = g^{cx+r \bmod q}$). We now prove that the above protocol satisfies the desired properties.

Theorem 1 *The protocol above is both honest-verifier zero-knowledge as well as a proof of knowledge.*

Proof We prove that the protocol is honest-verifier zero-knowledge by showing a simulator. The simulator proceeds as follows: it first chooses random $c, b \in \mathbb{Z}_q$ and then sets $A = g^b y^{-c}$. It outputs the transcript (A, c, b). We claim that the distribution of transcripts as output by this simulator is *identical* to the distribution of transcripts of real executions of the protocol (with an honest verifier). To see this, note that c is uniform in $\mathbb{Z}_q$ in both cases. Furthermore, the distributions of both real and simulated transcripts have A uniform in the group, independent of c. (This is clear for real transcripts since g is a generator and r is chosen uniformly at random from $\mathbb{Z}_q$, independent of c. For simulated transcripts, this is true since the value g^b is uniform in $\mathbb{G}$ and hence — for any c — the value $g^b y^{-c}$ is uniform in $\mathbb{G}$.) Finally, in both cases b is completely determined by A and c as the unique value satisfying $b = (c \cdot \log_g y + r) \bmod q$ (note that $\log_g y$ is well-defined, even if we cannot compute it efficiently). This completes this part of the proof.

To show that the protocol is a proof of knowledge, one needs to show a knowledge extractor satisfying the technical conditions hinted at (but not defined formally) in earlier lectures. Assume an adversarial prover who has some probability λ of successfully executing the protocol for a particular value of y. Note that this probability is only over the random challenge c sent in the second round. Before describing the extractor, we define some terminology: say the adversary *succeeds* on challenge c (assuming A is already fixed) if the adversary responds to this challenge with a b such that $y^c A = g^b$. Otherwise, say the adversary *fails*. We construct the extractor as follows:

> Receive some A from the adversary
> choose $c_1 \leftarrow \mathbb{Z}_q$ and send c_1 to the adversary
> if the adversary fails on c_1, halt
> otherwise, for $i = 1$ to q do:
>   choose $c_2 \leftarrow \mathbb{Z}_q \setminus \{c_1\}$
>   if the adversary succeeds on c_2, compute $\log_g y$ as described below and halt
>   if $g^i = y$, output i and halt

To complete the description, we describe how to compute $\log_g y$ when the extractor finds c_1, c_2 such that the adversary succeeds for both. In this case, we have A, c_1, c_2, b_1, b_2 such that $y^{c_1} A = g^{b_1}$ and $y^{c_2} A = g^{b_2}$. Dividing, we obtain:

$$y^{c_1 - c_2} = g^{b_1 - b_2},$$

and hence $\log_g y = (b_1 - b_2)(c_1 - c_2)^{-1} \bmod q$. Note that we can efficiently compute the latter since we have b_1, b_2, c_1, c_2, q, and $c_1 \neq c_2$ so $(c_1 - c_2)^{-1}$ exists.

Analyzing the extractor in its totality, we see that it computes the correct value for $\log_g y$ whenever the adversary succeeds on c_1. By assumption, this occurs with probability exactly λ. The only thing left to argue is that the extractor runs in expected polynomial time. To see this, we consider two possibilities: $\lambda > 1/q$ and $\lambda = 1/q$ (if $\lambda < 1/q$ then $\lambda = 0$ and the proof is easy). In the first case, say $\lambda = t/q$ for some integer $t > 1$. Now, the probability that the extractor enters the loop (i.e., the last four lines) is $\lambda = t/q$. The expected number of iterations of the loop, once reached, is $(\frac{t-1}{q})^{-1} = q/(t-1)$. So the expected total running time of the extractor is

$$\mathsf{poly} + \mathsf{poly} \cdot \frac{t}{q} \cdot \frac{q}{t-1} = \mathsf{poly} + \mathsf{poly}.$$

(Where poly in the above refers to some arbitrary polynomial in some implicit security parameter which determined the size of q.) In case $\lambda = 1/q$ the expected number of iterations of the loop is (at worst) q; however, the probability of entering the loop in the first place is $1/q$ and so the expected total running time is:

$$\mathsf{poly} + \mathsf{poly} \cdot \frac{1}{q} \cdot q = \mathsf{poly} + \mathsf{poly}.$$

In either case, then, the extractor runs in expected polynomial time. ■

3 Security Against Active Adversaries

As discussed in Lecture 25, to obtain security against active adversaries we can use a *witness-indistinguishable* proof of knowledge. Recall also that this only helps if there is more than one witness to speak of; for this reason we will now let the prover's secret be a *representation* of some value y with respect to *two* generators g, h. In more detail, if $g, h \in \mathbb{G}$ are generators we say that (x_1, x_2) is a representation of y if $g^{x_1} h^{x_2} = y$. Note that for any $y \in \mathbb{G}$ and any $x_1 \in \mathbb{Z}_q$ there is a unique $x_2 \in \mathbb{Z}_q$ such that (x_1, x_2) is a representation of y. In other words, there are q possible different "witnesses" or representations.

Whereas Schnorr's protocol is a proof of knowledge of the discrete logarithm of y (with respect to g), the following protocol due to Okamoto [3] is a proof of knowledge of a representation of y (with respect to g, h).

$$\begin{array}{lcl}
 & \mathbb{G}, q, g, h, y & \\
\underline{\mathcal{P}(x_1, x_2)} & & \underline{\mathcal{V}} \\
r_1, r_2 \leftarrow \mathbb{Z}_q & \xrightarrow{\quad A = g^{r_1} h^{r_2} \quad} & \\
 & \xleftarrow{\qquad c \qquad} & c \leftarrow \mathbb{Z}_q \\
b_1 = (cx_1 + r_1) \bmod q & & \\
b_2 = (cx_2 + r_2) \bmod q & \xrightarrow{\qquad b_1,\ b_2 \qquad} & Ay^c \stackrel{?}{=} g^{b_1} h^{b_2}
\end{array}$$

Let us first verify correctness. If $g^{x_1} h^{x_2} = y$ then we have:

$$Ay^c = g^{r_1} h^{r_2} (g^{x_1} h^{x_2})^c = g^{r_1 + cx_1} h^{r_2 + cx_2} = g^{b_1} h^{b_2}.$$

We now prove the desired properties of this protocol

Theorem 2 *The protocol above is a witness indistinguishable proof of knowledge.*

Proof The proof of knowledge property is easy to show, along the lines of the proof for the case of Schnorr's protocol. Without going through the details again, we simply show how to extract a representation from two accepting transcripts sharing the same value of A. Thus, assume we have values $A, c, c', b_1, b_2, b_1', b_2'$ such that $Ay^c = g^{b_1}h^{b_2}$ and $Ay^{c'} = g^{b_1'}h^{b_2'}$ but $c \neq c'$. Dividing, we obtain $y^{c-c'} = g^{b_1-b_1'}h^{b_2-b_2'}$ and so $(\frac{b_1-b_1'}{c-c'}, \frac{b_2-b_2'}{c-c})$ is a representation of y (again, we use the fact that $c - c' \neq 0$ so that the necessary inverse exists).

It remains to argue that the protocol is witness indistinguishable. To prove this, we show that for any adversarial verifier $\mathcal{V}^*$ and any two representations (x_1, x_2), (x_1', x_2'), the distribution on the transcripts of an execution of the protocol when the prover uses the first representation is *identical* to the distribution on the transcripts of an execution of the protocol when the prover uses the second representation. First note that the distribution over the first message A sent by the prover is independent of the representation being used. Next, the challenge c may be viewed as a deterministic function of A (since we can imagine fixing the random coins of the dishonest verifier — note that c may be chosen using some arbitrary (poly-time) function, since the adversary may not be following the honest verification procedure), and so the distribution on c is each case will be identical.

It only remains to argue about the final message b_1, b_2. Conditioned on some fixed values of A, c, when the prover is using the first representation this message is distributed according to:

$$\begin{aligned} b_1 &= cx_1 + r_1 \\ b_2 &= cx_2 + r_2, \end{aligned}$$

where r_1, r_2 are uniformly distributed *over representations of* A. (Another way to view this distribution is one in which r_1 is chosen uniformly from $\mathbb{Z}_q$ and then r_2 is the unique value such that (r_1, r_2) is a representation of A.) It is not hard to see that this is equivalent to saying that b_1, b_2 are uniformly distributed over representations of Ay^c. (Namely, b_1 is uniform in $\mathbb{Z}_q$ and then b_2 is the unique value such that (b_1, b_2) is a representation of Ay^c.) Conditioned on the same values of A, c, when the prover is using the *second* representation the final message is distributed according to:

$$\begin{aligned} b_1 &= cx_1' + r_1 \\ b_2 &= cx_2' + r_2, \end{aligned}$$

where r_1, r_2 are uniformly distributed *over representations of* A. But then b_1, b_2 are again distributed uniformly over representations of Ay^c.

The above discussion shows that the protocol is perfectly witness indistinguishable. ■

The above theorem does not quite allow us to directly apply the results from Lecture 25 and claim that Okamoto's scheme is an identification scheme secure against active adversaries. (If we were directly following the paradigm of Lecture 25, then we would have the prover's public key be y_1, y_2 and the prover would give a witness-indistinguishable proof of knowledge of either $\log_g y_1$ or $\log_g y_2$.) However, the same ideas behind the proof of Theorem 2, Lecture 25 can be used to prove this result. We assume the reader is familiar with that proof, and just sketch an outline of the proof here.

Theorem 3 *The protocol above gives an identification scheme secure against active adversaries.*

Proof (Sketch) Given an active adversary $\mathcal{A}$ attacking the scheme, we will use this adversary to compute $\log_g h$ (*not* $\log_g y$ or something similar as in the case of Schnorr's protocol!). We do this as follows:

1. Given input g, h we choose random x_1, x_2 and set $y = g^{x_1} h^{x_2}$. The public key y is given to $\mathcal{A}$ and the secret key is (x_1, x_2). Note that we know a perfectly valid secret key for y!

2. We then interact with $\mathcal{A}$, who will be acting as a dishonest verifier. Note that we can easily simulate the actions of a prover (without any rewinding or any difficulty) since we know a valid representation of y.

3. Once $\mathcal{A}$ is done with the previous stage, it then tries to impersonate the prover. If it succeeds, we run the knowledge extractor for the proof of knowledge to extract a representation (x_1', x_2') for y.

4. Assuming we have extracted some (x_1', x_2') as above, the key claim is that with all but negligible probability we have $(x_1', x_2') \neq (x_1, x_2)$. Why should this be the case? Well, since the protocol is perfectly witness indistinguishable, $\mathcal{A}$ has no idea (in an information-theoretic sense) which representation of y we know, and all valid representations are equally likely from $\mathcal{A}$'s point of view. Since there are q possible representations, $(x_1', x_2') = (x_1, x_2)$ with probability $1/q$, which is negligible.

5. Assuming we have extracted a representation (x_1', x_2') different from (x_1, x_2), we can compute $\log_g h$ by noting that $g^{x_1} h^{x_2} = g^{x_1'} h^{x_2'}$ and so
$$g^{x_1 - x_1'} = h^{x_2' - x_2}.$$
Thus, $\log_g h = (x_1 - x_1')(x_2' - x_2)^{-1} \bmod q$. (Note that since the representations are different we must have $x_2' \neq x_2 \bmod q$.)

6. Putting everything together, if $\mathcal{A}$ succeeds with non-negligible probability, then we compute $\log_g h$ with non-negligible probability.

■

4 From Identification Schemes to Signature Schemes

In this section, we show how any identification scheme of a certain form can be transformed into a signature scheme in the random oracle model. Assume a 3-round identification scheme in which the challenge sent by an honest verifier in the second round is generated by picking an element uniformly at random from some space. (The technique extends for multi-round protocols but we will deal with the 3-round case here since this is most common and leads to the most efficient signature schemes.) Let A, c, b denote the messages sent in the first, second, and third rounds, respectively. We transform this protocol into a signature scheme in the following way: the signer's public key is the public key of the identification scheme and the secret key is the secret key of the identification scheme. To sign message m, the signer begins by generating an initial message A just as in the identification scheme. The signer then computes $c = H(A, m)$ where H is a cryptographic hash function modeled as a random oracle. Finally, the signer computes the correct response b to this "challenge" c (using the secret key and its knowledge of how A was generated) and outputs the signature (A, b). Anyone can verify this signature on message m by computing $c = H(A, m)$ and then checking whether (A, c, b) is a valid transcript of the identification protocol with respect to the given public key. Applied to the Schnorr identification protocol, we obtain:

$$
\begin{array}{lcl}
 & PK = (\mathbb{G}, q, g, y) & \\
\underline{\mathsf{Sign}_{SK}(m) \text{ (where } SK = x = \log_g y)} & & \underline{\mathcal{V}(PK, m)} \\
r \leftarrow \mathbb{Z}_q; A = g^r & & \\
c = H(A, m); b = cx + r & \xrightarrow{\quad A,\ b \quad} & \\
 & & c = H(A, m) \\
 & & y^c A \stackrel{?}{=} g^b
\end{array}
$$

The general transformation described above (i.e., for an arbitrary identification scheme) was first proposed by Fiat and Shamir [2] and is known as the *Fiat-Shamir transformation*. It has since been analyzed rigorously in numerous works. We state the following without proof, and refer the interested reader to [1] for more details and a full proof.

Theorem 4 *When the Fiat-Shamir transformation is applied to any identification scheme (of the appropriate form) which is secure against passive attacks, the resulting signature scheme is existentially unforgeable under adaptive chosen-message attacks in the random oracle model.*

The above theorem is quite nice, in that it shows that an identification protocol secure against a very *weak* form of attack (i.e., a passive attack) suffices to give a signature scheme which is secure in (essentially) the *strongest* sense.

References

[1] M. Abdalla, J. H. An, M. Bellare, C. Namprempre. From Identification to Signatures via the Fiat-Shamir Transform: Minimizing Assumptions for Security and Forward-Security. Eurocrypt 2002.

[2] A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. Crypto '86.

[3] T. Okamoto. Provably-Secure and Practical Identification Schemes and Corresponding Signature Schemes. Crypto '92.

[4] C.-P. Schnorr. Efficient Identification and Signatures for Smart Cards. Crypto '89.