- Queries $H(m_i)$ with $i \neq j$ are answered with $y_i = [\sigma_i^e \bmod N]$, where σ_i is uniform in $\mathbb{Z}_N^*$. Since exponentiation to the eth power is a one-to-one function, y_i is uniformly distributed as well.

Finally, observe that whenever experiment $\mathsf{Sig\text{-}forge}''_{\mathcal{A},\Pi}(n)$ would output 1, then $\mathcal{A}'$ outputs a correct solution to its given RSA instance. This follows since $\mathsf{Sig\text{-}forge}''_{\mathcal{A},\Pi}(n) = 1$ implies that $j = i$ and $\sigma^e = H(m_i) \bmod N$. Now, when $j = i$, algorithm $\mathcal{A}'$ does not abort and in addition $H(m_i) = y$. Thus, $\sigma^e = H(m_i) = y \bmod N$, and so σ is the desired inverse. Using Equation (13.1), this means that

$$\begin{aligned}\Pr[\mathsf{RSA\text{-}inv}_{\mathcal{A}',\mathsf{GenRSA}}(n) = 1] &= \Pr[\mathsf{Sig\text{-}forge}''_{\mathcal{A},\Pi}(n) = 1] \\ &= \frac{\Pr[\mathsf{Sig\text{-}forge}_{\mathcal{A},\Pi}(n) = 1]}{q(n)}. \end{aligned} \tag{13.2}$$

If the RSA problem is hard relative to GenRSA, there is a negligible function negl such that $\Pr[\mathsf{RSA\text{-}inv}_{\mathcal{A}',\mathsf{GenRSA}}(n) = 1] \leq \mathsf{negl}(n)$. Since $q(n)$ is polynomial, we conclude from Equation (13.2) that $\Pr[\mathsf{Sig\text{-}forge}_{\mathcal{A},\Pi}(n) = 1]$ is negligible as well. This completes the proof. ∎

RSA PKCS #1 standards. RSA PKCS #1 v1.5 specifies a signature scheme that is very similar to RSA-FDH. A more-complex scheme that can be viewed as a randomized variant of RSA-FDH has been included in the PKCS #1 standard since version 2.1.

13.5 Signatures from the Discrete-Logarithm Problem

Signature schemes can be based on the discrete-logarithm assumption as well, although the assumption does not lend itself as readily to signatures as the RSA assumption does. In Sections 13.5.1 and 13.5.2 we describe the Schnorr signature scheme that can be proven secure in the random-oracle model. In Section 13.5.3 we describe the DSA and ECDSA signature schemes; these standardized schemes are widely used even though they have no full proof of security.

13.5.1 Identification Schemes and Signatures

The underlying intuition for the Schnorr signature scheme is best explained by taking a slight detour to discuss (public-key) *identification schemes*. We then describe the *Fiat–Shamir transform* that can be used to convert identification schemes to signature schemes in the random-oracle model. Finally,

we present the Schnorr identification scheme—and corresponding signature scheme—based on the discrete-logarithm problem.

Identification Schemes

An identification scheme is an interactive protocol that allows one party to prove its identity (i.e., to *authenticate* itself) to another. This is a very natural notion, and it is common nowadays to authenticate oneself when logging in to a website. We call the party identifying herself (e.g., the user) the "prover," and the party verifying the identity (e.g., the web server) the "verifier." Here, we are interested in the public-key setting where the prover and verifier do not share any secret information (such as a password) in advance; instead, the verifier only knows the public key of the prover. Successful execution of the identification protocol convinces the verifier that it is communicating with the intended prover rather than an imposter.

We will only consider three-round identification protocols of a specific form, where the prover is specified by two algorithms $\mathcal{P}_1, \mathcal{P}_2$ and the verifier's side of the protocol is specified by an algorithm $\mathcal{V}$. The prover runs $\mathcal{P}_1(sk)$ using its private key sk to obtain an initial message I along with some state st, and initiates the protocol by sending I to the verifier. In response, the verifier sends a challenge r chosen uniformly from some set Ω_{pk} defined by the prover's public key pk. Next, the prover runs $\mathcal{P}_2(sk, \mathsf{st}, r)$ to compute a response s that it sends back to the verifier. Finally, the verifier computes $\mathcal{V}(pk, r, s)$ and accepts if and only if this results in the initial message I; see Figure 13.1. Of course, for correctness we require that if the legitimate prover executes the protocol correctly then the verifier should always accept.

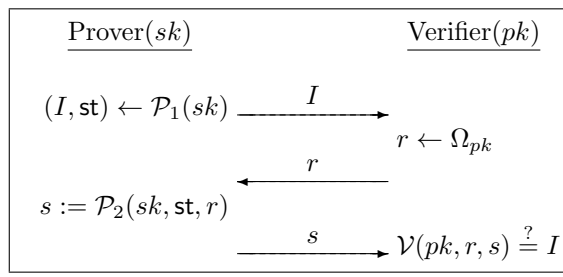

FIGURE 13.1: A three-round identification scheme.

For technical reasons, we assume identification schemes that are "non-degenerate," which intuitively means that there are many possible initial messages I, and none has a high probability of being sent. Formally, a scheme is *non-degenerate* if for every private key sk and any fixed initial message I, the

probability that $\mathcal{P}_1(sk)$ outputs I is negligible. (Any identification scheme can be trivially modified to be non-degenerate by sending a uniform n-bit string along with the initial message.)

The basic security requirement of an identification scheme is that an adversary who does not know the prover's secret key should be unable to fool the verifier into accepting. This should hold even if the attacker is able to passively eavesdrop on multiple (honest) executions of the protocol between the prover and verifier. We formalize such eavesdropping via an oracle Trans_{sk} that, when called without any input, runs an honest execution of the protocol and returns to the adversary the entire transcript (I, r, s) of the interaction.

Let $\Pi = (\mathsf{Gen}, \mathcal{P}_1, \mathcal{P}_2, \mathcal{V})$ be an identification scheme, and consider the following experiment for an adversary $\mathcal{A}$ and parameter n:

The identification experiment $\mathsf{Ident}_{\mathcal{A},\Pi}(n)$:

1. $\mathsf{Gen}(1^n)$ *is run to obtain keys* (pk, sk).
2. *Adversary* $\mathcal{A}$ *is given* pk *and access to an oracle* Trans_{sk} *that it can query as often as it likes.*
3. *At any point during the experiment,* $\mathcal{A}$ *outputs a message* I. *A uniform challenge* $r \in \Omega_{pk}$ *is chosen and given to* $\mathcal{A}$*, who responds with some* s. *(*$\mathcal{A}$ *may continue to query* Trans_{sk} *even after receiving* r*.)*
4. *The experiment outputs* 1 *if and only if* $\mathcal{V}(pk, r, s) \stackrel{?}{=} I$.

DEFINITION 13.8 *An identification scheme* $\Pi = (\mathsf{Gen}, \mathcal{P}_1, \mathcal{P}_2, \mathcal{V})$ *is* secure against a passive attack*, or just* secure*, if for all probabilistic polynomial-time adversaries* $\mathcal{A}$*, there exists a negligible function* negl *such that:*

$$\Pr[\mathsf{Ident}_{\mathcal{A},\Pi}(n) = 1] \leq \mathsf{negl}(n).$$

It is also possible to consider stronger notions of security, for example, where the adversary can also carry out *active* attacks on the protocol by impersonating a verifier and possibly sending maliciously chosen values r. We will not need this for our application to signature schemes.

From Identification Schemes to Signatures

The Fiat–Shamir transform (Construction 13.9) provides a way to convert any (interactive) identification scheme into a (non-interactive) signature scheme. The basic idea is for the signer to act as a prover, running the identification protocol *by itself*. That is, to sign a message m, the signer first computes I, and next generates the challenge r by applying some function H to I and m. It then derives the correct response s. The signature on m is (r, s), which can be verified by (1) recomputing $I := \mathcal{V}(pk, r, s)$ and then (2) checking that $H(I, m) \stackrel{?}{=} r$.

CONSTRUCTION 13.9

Let $(\mathsf{Gen}_{\mathsf{id}}, \mathcal{P}_1, \mathcal{P}_2, \mathcal{V})$ be an identification scheme, and construct a signature scheme as follows:

- Gen: on input 1^n, simply run $\mathsf{Gen}_{\mathsf{id}}(1^n)$ to obtain keys pk, sk.

  The public key pk specifies a set of challenges Ω_{pk}. As part of key generation, a function $H : \{0,1\}^* \to \Omega_{pk}$ is specified, but we leave this implicit.
- Sign: on input a private key sk and a message $m \in \{0,1\}^*$, do:
  1. Compute $(I, \mathsf{st}) \leftarrow \mathcal{P}_1(sk)$.
  2. Compute $r := H(I, m)$.
  3. Compute $s := \mathcal{P}_2(sk, \mathsf{st}, r)$.

  Output the signature (r, s).
- Vrfy: on input a public key pk, a message m, and a signature (r, s), compute $I := \mathcal{V}(pk, r, s)$ and output 1 if and only if

$$H(I, m) \stackrel{?}{=} r.$$

The Fiat–Shamir transform.

A signature (r, s) is "bound" to a specific message m because r is a function of both I and m; changing m thus results in a completely different r. If H is modeled as a random oracle mapping inputs uniformly onto Ω_{pk}, then the challenge r is uniform; intuitively, it will be just as difficult for an adversary (who does not know sk) to find a valid signature (r, s) on a message m as it would be to impersonate the prover in an honest execution of the protocol. This intuition is formalized in the proof of the following theorem.

THEOREM 13.10 *Let Π be an identification scheme, and let Π' be the signature scheme that results by applying the Fiat–Shamir transform to it. If Π is secure and H is modeled as a random oracle, then Π' is secure.*

PROOF Let $\mathcal{A}'$ be a probabilistic polynomial-time adversary attacking the signature scheme Π', with $q = q(n)$ an upper bound on the number of queries that $\mathcal{A}'$ makes to H. We make a number of simplifying assumptions without loss of generality. First, we assume that $\mathcal{A}'$ makes any given query to H only once. We also assume that after being given a signature (r, s) on a message m with $\mathcal{V}(pk, r, s) = I$, the adversary $\mathcal{A}'$ never queries $H(I, m)$ (since it knows the answer will be r). Finally, we assume that if $\mathcal{A}'$ outputs a forged signature (r, s) on a message m with $\mathcal{V}(pk, r, s) = I$, then $\mathcal{A}'$ had previously queried $H(I, m)$.

We construct an efficient adversary $\mathcal{A}$ that uses $\mathcal{A}'$ as a subroutine and attacks the identification scheme Π:

Algorithm $\mathcal{A}$:
The algorithm is given pk and access to an oracle Trans_{sk}.

1. Choose uniform $j \in \{1, \ldots, q\}$.
2. Run $\mathcal{A}'(pk)$. Answer its queries as follows:
   When $\mathcal{A}'$ makes its ith random-oracle query $H(I_i, m_i)$, answer it as follows:
   - If $i = j$, output I_j and receive in return a challenge r. Return r to $\mathcal{A}'$ as the answer to its query.
   - If $i \neq j$, choose a uniform $r \in \Omega_{pk}$ and return r as the answer to the query.

   When $\mathcal{A}'$ requests a signature on m, answer it as follows:
   (a) Query Trans_{sk} to obtain a transcript (I, r, s) of an honest execution of the protocol.
   (b) Return the signature (r, s).
3. If $\mathcal{A}'$ outputs a forged signature (r, s) on a message m, compute $I := \mathcal{V}(pk, r, s)$ and check whether $(I, m) \stackrel{?}{=} (I_j, m_j)$. If so, then output s. Otherwise, abort.

The view of $\mathcal{A}'$ when run as a subroutine by $\mathcal{A}$ in experiment $\mathsf{Ident}_{\mathcal{A},\Pi}(n)$ is *almost* identical to the view of $\mathcal{A}'$ in experiment $\mathsf{Sig\text{-}forge}_{\mathcal{A}',\Pi'}(n)$. Indeed, all the H-queries that $\mathcal{A}'$ makes are answered with a uniform value from Ω_{pk}, and all the signing queries that $\mathcal{A}'$ makes are answered with valid signatures having the correct distribution. The only difference between the views is that when $\mathcal{A}'$ is run as a subroutine by $\mathcal{A}$ it is possible for there to be an inconsistency in the answers $\mathcal{A}'$ receives from its queries to H: specifically, this happens if $\mathcal{A}$ ever answers a signing query for a message m using a transcript (I, r, s) for which $H(I, m)$ is already defined (that is, $\mathcal{A}'$ had previously queried (I, m) to H) and $H(I, m) \neq r$. However, if Π is non-degenerate then this only ever happens with negligible probability. Thus, the probability that $\mathcal{A}'$ outputs a forgery when run as a subroutine by $\mathcal{A}$ is $\Pr[\mathsf{Sig\text{-}forge}_{\mathcal{A}',\Pi'}(n) = 1] - \mathsf{negl}(n)$ for some negligible function negl.

Consider an execution of experiment $\mathsf{Ident}_{\mathcal{A},\Pi}(n)$ in which $\mathcal{A}'$ outputs a forged signature (r, s) on a message m, and let $I := \mathcal{V}(pk, r, s)$. Since j is uniform and independent of everything else, the probability that $(I, m) = (I_j, m_j)$ (even conditioned on the event that $\mathcal{A}'$ outputs a forgery) is exactly $1/q$. (Recall we assume that if $\mathcal{A}'$ outputs a forged signature (r, s) on a message m with $\mathcal{V}(pk, r, s) = I$, then $\mathcal{A}'$ had previously queried $H(I, m)$.) When both events happen, $\mathcal{A}$ successfully impersonates the prover. Indeed, $\mathcal{A}$ sends I_j as its initial message, receives in response a challenge r, and responds with s. But $H(I_j, m_j) = r$ and (since the forged signature is valid) $\mathcal{V}(pk, r, s) = I$. Putting everything together, we see that

$$\Pr[\mathsf{Ident}_{\mathcal{A},\Pi}(n) = 1] \geq \frac{1}{q(n)} \cdot \big(\Pr[\mathsf{Sig\text{-}forge}_{\mathcal{A}',\Pi'}(n) = 1] - \mathsf{negl}(n)\big)$$

or

$$\Pr[\mathsf{Sig\text{-}forge}_{\mathcal{A}',\Pi'}(n) = 1] \leq q(n) \cdot \Pr[\mathsf{Ident}_{\mathcal{A},\Pi}(n) = 1] + \mathsf{negl}(n).$$

If Π is secure then $\Pr[\mathsf{Ident}_{\mathcal{A},\Pi}(n) = 1]$ is negligible; since $q(n)$ is polynomial this implies that $\Pr[\mathsf{Sig\text{-}forge}_{\mathcal{A}',\Pi'}(n) = 1]$ is also negligible. Because $\mathcal{A}'$ was arbitrary, this means Π' is secure. ■

13.5.2 The Schnorr Identification/Signature Schemes

The Schnorr identification scheme is based on hardness of the discrete-logarithm problem. Let $\mathcal{G}$ be a polynomial-time algorithm that takes as input 1^n and (except possibly with negligible probability) outputs a description of a cyclic group $\mathbb{G}$, its order q (with $\|q\| = n$), and a generator g. To generate its keys, the prover runs $\mathcal{G}(1^n)$ to obtain $(\mathbb{G}, q, g)$, chooses a uniform $x \in \mathbb{Z}_q$, and sets $y := g^x$; the public key is $\langle \mathbb{G}, q, g, y \rangle$ and the private key is x. To execute the protocol (see Figure 13.2), the prover begins by choosing a uniform $k \in \mathbb{Z}_q$ and setting $I := g^k$; it sends I as the initial message. The verifier chooses and sends a uniform challenge $r \in \mathbb{Z}_q$; in response, the prover computes $s := [rx + k \bmod q]$. The verifier accepts if and only if $g^s \cdot y^{-r} \stackrel{?}{=} I$. Correctness holds because

$$g^s \cdot y^{-r} = g^{rx+k} \cdot (g^x)^{-r} = g^k = I.$$

Note that I is uniform in $\mathbb{G}$, and so the scheme is non-degenerate.

Before giving the proof, we provide some high-level intuition. A first important observation is that passive eavesdropping is of no help to the attacker. The reason is that the attacker can *simulate* transcripts of honest executions on its own, based only on the public key and *without* knowledge of the private key. To do this, the attacker just reverses the order of the steps: it first chooses uniform and independent $r, s \in \mathbb{Z}_q$ and then sets $I := g^s \cdot y^{-r}$. In an honest transcript (I, r, s), the initial message I is a uniform element of $\mathbb{G}$, the

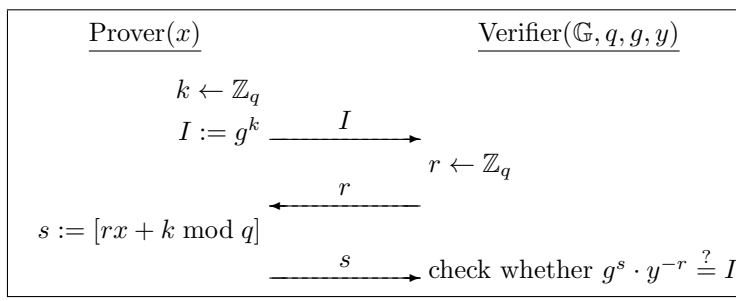

FIGURE 13.2: An execution of the Schnorr identification scheme.