Digital Signatures

Q1: A (digital) signature scheme consists of three probabilistic polynomial-time algorithms (Gen, Sign, Vrfy). Explain each algorithm with its input and output.

A1: These 3 algorithms are:

- Gen: on input a security parameter 1^n, outputs public key pk and secret key sk.
- $Sign$: on input a private key sk and a message m, outputs a signature σ.
- $Vrfy$: on input a public key pk, message m and a signature σ, outputs 1 or 0.

Q2: Compare signature schemes with MAC schemes.

A2: For MACs:

- Only the holder of the key can verify it.
- You can't transfer a MAC to some other party.
- Does not have non-repudiation.
- Shorter in length and faster to compute.

For Digital Signatures:

- Anyone can verify it (with the public key).
- You can transfer the signature to some other party.
- Has non-repudiation.
- Longer keys than MAC, slower to compute.

Q3: Formally define security (existential unforgeability against adaptive chosen message attacks) of a signature scheme.

A3: The $\texttt{Sig-Forge}_{\mathcal{A},\Pi}(n)$ experiment is defined in the algorithm 1 below.

Q4: Let Π be a secure signature scheme for messages of fixed length l. Construct a signature scheme based on Π that is secure for arbitrary-length messages. You can make use of any crypto-primitive you have learned. (No security proof is needed.)

A4: This is known as Hash-and-Sign paradigm. Let Π be $(Gen, Sign, Vrfy)$, we will construct Π'.

- Gen': Same as Gen.
- $Sign$: on input a private key sk and a message m, outputs a $Sign_{sk}(H(m))$.

Algorithm 1 $\texttt{Sig-Forge}_{\mathcal{A},\Pi}(n)$ experiment

$\texttt{Chal}$ generates a public and private key pair $(pk, sk) \leftarrow Gen(1^n)$.
$\mathcal{A}$ obtains pk, and is given oracle access to $Sign(.)$. It can send a message m' and it will have in return $\sigma' = Sign_{sk}(m')$.
After some time, $\mathcal{A}$ send m, σ to $\texttt{Chal}$, where m has not been used in oracle access before.
if $Vrfy_{pk}(m, \sigma) = 1$ **then**
  output 1, $\mathcal{A}$ wins.
else
  output 0.
end if

- $Vrfy$: on input a public key pk, message m and a signature σ, outputs $Vrfy_{pk}(H(m), \sigma)$.

Here H is a collision resistant hash function.

Q5: Assume Charlie, acting as a certificate authority (CA), issues a certificate for Bob. Charlie's signature public key is pk_C and he always issues certificates only to the trustworthy people. Assume further that Bob's key pk_B is a public key for a signature scheme. Charlie's certificate for Bob looks like $cert_{C \to B} = Sign_{sk_C}(\text{Bob's key is } pk_B)$. Bob issues a certificate for Alice of the form $cert_{B \to A} = Sign_{sk_B}(\text{Alice's key is } pk_A)$. Now, Alice wants to communicate with some fourth party Dave who knows Charlie's public key pk_C (but not Bob's). Assume Charlie is offline now. Suggest a solution by which Alice can convince Dave that pk_A is her authentic key, without any party having access to Charlie (since Charlie is offline).

A5: Alice sends $\langle \text{Alice}, pk_A, cert_{B \to A} \rangle$ to Dave. Dave can then ask Bob for $cert_{C \to B}$, which Dave can verify with pk_C. After that verification, Dave knows that B must be trustworthy because C issued a certificate to it. Furthermore, it can trust pk_B and use this pk_B to verify $cert_{B \to A}$, which reveals that pk_A is the correct public key of Alice.