Cryptography (SS2006) Prof. Michael Backes

Solutions for Exercise Sheet 9

Out: 07/14/2006 Saarland University

Problem 1: Certificate Revocation Trees

Recall that the signature provided by the root-VA contains the value of the root node of the hash tree, as well as the depth of the tree, i.e., the distance between the leaf nodes and the root node.

To simplify matters we let the certificate ID stored in the left-most leaf be 0 and the certificate ID stored in the right-most leaf node be the maximal possible ID. Then we do not need to treat these cases separately.

further recall that, to prove that a certificate with ID c was not revoked, the VA has to send two certificate IDs with $c_1 < c < c_2$, such that c_1 and c_2 are located on adjacent leaf-nodes. It also has to include enough information such that the user can reconstruct the root-node and then test if this matches the one in the signature.

For example, the following information can be sent: Starting with the root node, the input is given as a tuple (a_{11}, a_{12}). If either c_1 or c_2 is a child, then we proceed recursivly, e.g., $(a_{11}, ((a_{35}, c_1), (c_2, a_{38})))$

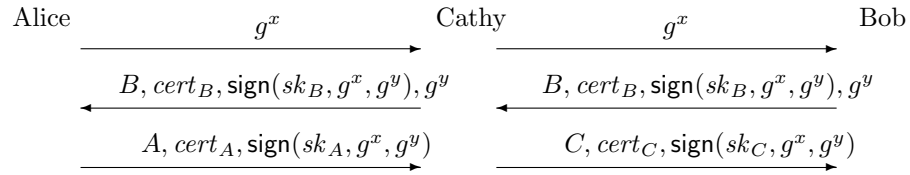
The user the reconstructs the necessary part of the tree, i.e., along the paths from the root node to the two leafs c_1 and c_2.

Then an argument (almost) literally as for Merkle hash trees tells us that this reconstructed part of the tree is identical to the corresponding part in the original hash tree, if the underlying hash function is collision resistant.

Problem 2: Authenticated Diffie-Hellman

(a) In the well-known man-in-the-middle attack against the Diffie-Hellman key exchange, an active adversary intercepts the communication between Alice and Bob, sending an $g^{x'}$ to Bob in the first step, and an $g^{y'}$ to Alice in the second step, where he knows values x', y'. This attack cannot be carried out in the above protocol, as the value g^y is signed with Bob's key.

(b) It is sufficient to modify the last message, the first and the second message are simply forwarded.



Note that the signature $\mathsf{sign}(sk_C, g^x, g^y)$ in the last step can be computed by Cathy, as she already has seen g^x and g^y in plain in steps one and two.

(c) Suppose Bob runs a chatroom and Alice earns money by giving expert advice in this chatroom. A natural realization would be that Alice authenticates to the server and vice versa. Then she can read from and write to the board using the session key they agreed upon in the authentication protocol. Then for each advice Alice gives on the board she is credited money by Bob.

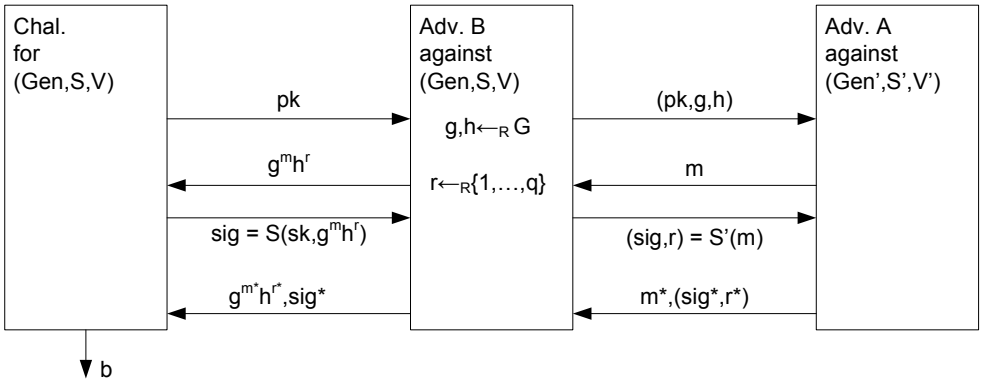
If Cathy can manipulate this authentication process such that Alice still believes she is talking with Bob (respectively the chatroom), but Bob thinks he is speaking with Cathy, then Bob will credit the money to Cathy's account, not to Alice's. This is obviously bad, thus we regard this a threat.

Problem 3: Offline Signatures

(a) The value $\mathsf{H}(m)$ is a random value if the hash function is treated as a random function. Although we cannot come up with a formal proof, there is no known way to speed up this computation such that the online phase, i.e., when the actual message is know, can be computed without an exponentiation. The purpose of this was to let you discover why all possible attempts fail and to show you why the following more complicated construction is necessary.

(b) The signer does the following: It chooses a random $s \leftarrow_{\mathcal{R}} \{1, \ldots, q\}$ and signs g^s by computing $sig' := \mathsf{sign}(sk, g^s)$. If the signer later gets a message m' he has to sign he computes $r' := (s - m')x^{-1}$. Then $g^s = g^{m'} h^{r'}$, thus the tuple (sig, r') is a valid signature for m'.

(c) The following picture describes how an attacker A against the constructed scheme $(\mathsf{Gen}', \mathsf{S}', \mathsf{V}')$ can be turned into an attacker B against the underlying scheme $(\mathsf{Gen}, \mathsf{S}, \mathsf{V})$:



It remains to be shown that the attacker B is successful if A was successful. This follows immediately from the collision resistance of the hash function $(m, r) \mapsto \mathsf{H}(m, r) := g^m h^r$: If $\mathsf{H}(m^*, r^*), sig^*$ was queried before by B as $(\mathsf{H}(m_i, r_i), sig_i)$, but $(m^*, (sig^*, r^*))$ was not queried before by A, then (m^*, r^*), (m_i, r_i) is a collision for H.