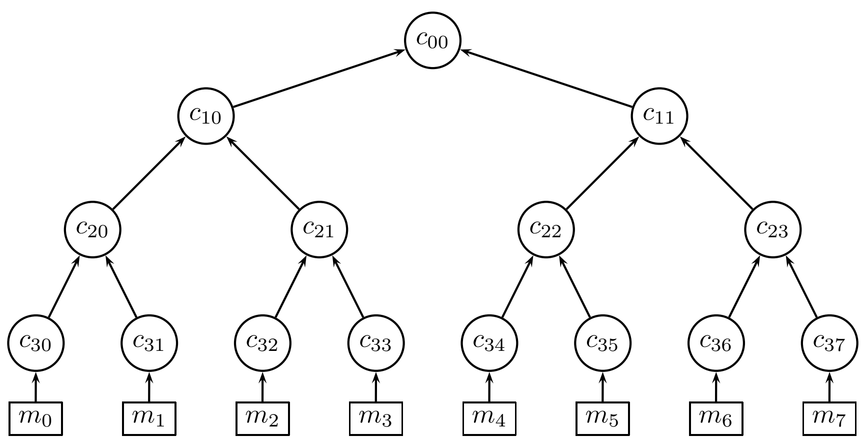
MTAT.07.003 Cryptology II
Spring 2012 / Exercise session ?? / Example Solution

Exercise (Merkle trees are binding commitments). *Show that Merkle tree is a binding commitment if the underlying hash function family $\mathcal{H}$ is (t, ε)-collision resistant. Recall that Merkle tree is a binary tree with vertices (c_{ij}), where intermediate leafs are computes as*

$$c_{ij} = h(c_{i+1,2j}, c_{i+1,2j+1}), \qquad i \in \{0, \ldots, k-1\}, \quad j \in \left\{0, \ldots 2^i - 1\right\}$$

and leafs $c_{k,j}$ for $j \in \left\{0, 2^k - 1\right\}$ are messages to be committed. The commitment digest is c_{00} and to open a message $c_{k,j}$ you have to open minimal number of leaks and intermediate vertices needed to compute c_{00}. A commitment is valid, if one indeed obtains c_{00} from the released messages.

Solution. Let us first illustrate how one uses Merkle tree to commit a bitstring m consisting of eight blocks $m_7, \ldots, m_0 \in \mathcal{M}$. Note that the hash function h used to compute the commitment digest c_{00} must be of type $h : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$. In order to commit the message m, we first treat its blocks as third level nodes in the Merkle tree and compute the values of intermediate nodes c_{ij} according to the specification. Let $\mathsf{GetRoot}$ be the corresponding algorithm that computes the root of the hash tree, as illustrated below.



In order to double open the commitment c_{00}, one must produce alternative message $\overline{m}$ consisting also from eight blocks $\overline{m}_7, \ldots, \overline{m}_0$ such that the digest computation leads to the same result. More generally, we are interested what is the best advantage against the binding game:

$$\begin{array}{l}\mathcal{G}\\ \left[\begin{array}{l} h \leftarrow \mathcal{H} \\ (c_{00}, m, \overline{m}) \leftarrow \mathcal{A}(h) \\ \textsf{if } c_{00} \neq \mathsf{GetRoot}(m) \textsf{ then } \mathbf{return}\ 0 \\ \textsf{if } c_{00} \neq \mathsf{GetRoot}(\overline{m}) \textsf{ then } \mathbf{return}\ 0 \\ \mathbf{return}\ [m \neq \overline{m}] \end{array}\right.\end{array}$$

where the third and fourth line check that the c_{00} is indeed a valid commitment to m and $\overline{m}$. Also, note that the public parameter of the commitment scheme is the description of a hash function h and public parameter generation is random sampling of an hash function.

It is straightforward to see that Merkle tree without additional restrictions is not binding at all. For example, let c_{00} be the digest corresponding to the message blocks $m_0, \ldots, m_7$. Then four block message $\overline{m}$ consisting intermediate values:

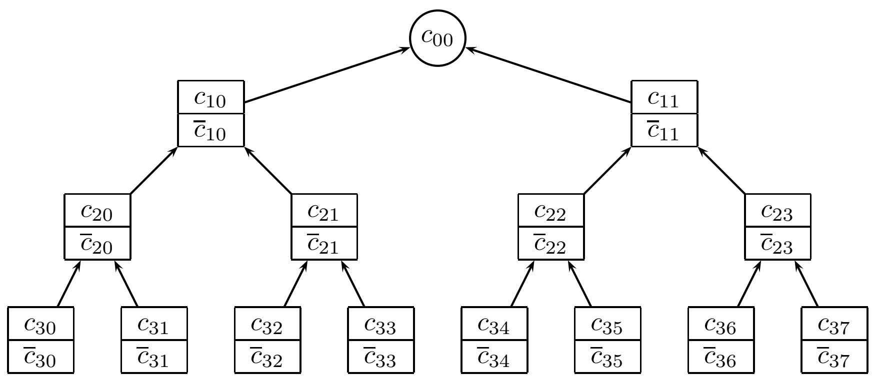
$$c_{20} = h(m_0, m_1), \quad c_{21} = h(m_2, m_3), \quad c_{22} = h(m_4, m_5), \quad c_{23} = h(m_6, m_7)$$

leads to the same digest c_{00}. Hence, we must clarify the definition of the Merkle tree commitments by requiring that the number of layers k is fixed, as implicitly suggested by the exercise text.

Next, we prove that commitment scheme based on the Merkle tree with k levels is a binding under the assumption that the hash function family $\mathcal{H}$ is (t, ε)-collision resistant. For that, we must convert an adversary $\mathcal{A}$ against the binding game $\mathcal{G}$ to another adversary $\mathcal{B}$ that can break collision resistance property of the underlaying hash function family $\mathcal{H}$. Recall that the collision resistance property of an hash function family is defined through the following game:

$$
\begin{array}{l}
\mathcal{Q} \\
\left[\begin{array}{l}
h \xleftarrow[u]{} \mathcal{H} \\
(x_0, x_1) \leftarrow \mathcal{B}(h) \\
\mathsf{if}\ x_0 = x_1\ \mathsf{then}\ \mathbf{return}\ 0 \\
\mathbf{return}\ [h(x_0) \stackrel{?}{=} h(x_1)]\ .
\end{array}\right.
\end{array}
$$

Assume that $\mathcal{A}$ returns a valid double opening $(c_{00}, m, \overline{m})$. Then there must be two instances of Merkle trees with the same root node that can be aligned, as illustrated below.



More formally, let c_{ij} denote the intermediate values corresponding to the message m and let $\overline{c}_{ij}$ denote intermediate values corresponding to the message $\overline{m}$. It is easy to see that if the root of a subtree $c_{i,j}$ has the same value has $\overline{c}_{i,j}$, then we have either identical children: $c_{i+1,2j} = \overline{c}_{i+1,2j}$ and $c_{i+1,2j+1} = \overline{c}_{i+1,2j+1}$ or there is an explicit hash collision:

$$
\begin{aligned}
(c_{i+1,2j}, c_{i+1,2j+1}) &\neq (\overline{c}_{i+1,2j}, \overline{c}_{i+1,2j+1})\ , \\
h(c_{i+1,2j}, c_{i+1,2j+1}) &= h(\overline{c}_{i+1,2j}, \overline{c}_{i+1,2j+1})\ .
\end{aligned}
$$

By applying this observation recursively, we either discover a hash collision or all vertices in the tree are identical. The latter cannot happen as $m \neq \overline{m}$ in case of valid double opening.

Hence, we can extract hash collision from a double opening by splitting the messages m and $\overline{m}$ into the kth layer values $c_{k,j}$ and $\overline{c}_{k,j}$ and then computing the values c_{ij} and $\overline{c}_{ij}$ of next layers until we find the hash collision. The corresponding adversary is depicted below:

$$
\begin{array}{l}
\mathcal{B}(h) \\
\left\lceil \begin{array}{l}
(c_{00}, m, \overline{m}) \leftarrow \mathcal{A}(h) \\
\text{Let } c_{k0}, \ldots, c_{k2^k-1} \text{ be the block representation of } m. \\
\text{Let } \overline{c}_{k0}, \ldots, \overline{c}_{k2^k-1} \text{ be the block representation of } \overline{m}. \\
\text{For } i \in (k, \ldots, 1) \text{ do} \\
\quad \left\lceil \begin{array}{l}
\text{For } j \in (0, \ldots 2^{k-1} - 1) \text{ do} \\
\quad \left\lceil \begin{array}{l}
x_0 \leftarrow (c_{i,2j}, c_{i,2j+1}) \\
x_1 \leftarrow (\overline{c}_{i,2j}, \overline{c}_{i,2j+1}) \\
c_{i,j} \leftarrow h(c_{i,2j}, c_{i,2j+1}) \\
\overline{c}_{i,j} \leftarrow h(c_{i,2j}, c_{i,2j+1}) \\
\hat{c}_{i-1,j} \leftarrow h(\overline{c}_{i,2j}, \overline{c}_{i,2j+1}) \\
\quad \mathsf{if} \;\; c_{i,j} = \overline{c}_{i,j} \wedge x_0 \neq x_1 \; \mathsf{then} \\
\quad \left[\textbf{return } (x_0, x_1) \right.
\end{array} \right.
\end{array} \right. \\
\textbf{return } \perp
\end{array} \right.
\end{array}
$$

Note that $\mathcal{B}$ is guaranteed to succeed if $\mathcal{A}$ provides a valid double opening, since the condition inside the second loop must be met for some iteration by the reasoning given above. Hence, we have established

$$
\Pr\left[\mathcal{Q}^{\mathcal{B}} = 1\right] \geq \Pr\left[\mathcal{G}^{\mathcal{A}} = 1\right] \quad .
$$

Note that $\mathcal{B}$ can be more successful than $\mathcal{A}$, as invalid double opening might still reveal the hash collision. Of course, the probability of such events is negligible for reasonable adversaries.

Note that the running-time of $\mathcal{B}$ is $t_{\mathcal{A}} + \Theta(2^k)$, where $t_{\mathcal{A}}$ is the running-time of $\mathcal{A}$ and k is the height of the tree. At first glance the overhead $\Theta(2^k)$ seems worrisome, as it seems to lead to exponential slowdown. However, note that k must be small in practical applications as the length of the committed message is also $\Theta(2^k)$ and the time needed to verify the digest is also $\Theta(2^k)$. In fact, the overhead of $\mathcal{B}$ roughly corresponds to the verification of both decommitments. As a result, we still obtain a tight connection between the collision resistance and binding property. Namely, if the hash function family $\mathcal{H}$ is (t, ε)-collision resistant, then Merkle tree commitment is $(t - \Theta(2^k), \varepsilon)$-binding.