## 数据动态更新

方案通过基于 rank 的 Merkle 哈希树来支持高效的数据动态更新操作。数据拥有者在将自己的数据外包给去中心化存储提供商后，可能要对外包的数据进行修改，即对数据进行动态操作，数据拥有者可以对存储在去中心化存储提供商中的数据进行插入、删除和更新。然而，在具有连续性验证能力的检查方案中，一个存储周期内需要对数据进行多次检查，如果在这个期间内对数据进行更改，会改变数据对应的验证标签，进而影响了整个数据的累加器值。因此，引入了一个与用户更新相对应的索引表。用户的每一次更新都会改变数据的累加器值和辅助参数，并将这些参数应用到下一次验证过程中，即如果本次验证过程中发生了更新，则下一次验证中用到的累加器值和辅助参数也会随之更新。

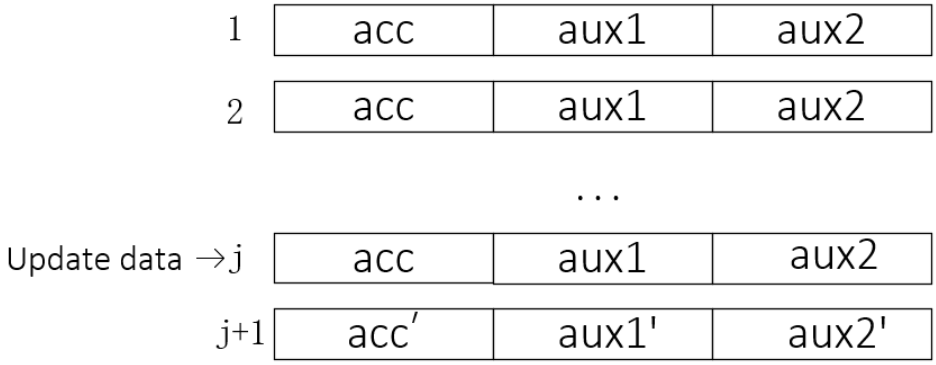


图 3.7 对应检查次数的辅助参数的索引表

对于数据块修改，数据块修改是用户使用最频繁的操作，即用户用新的数据块替代特定的旧的数据块，相比数据块删除和数据块插入，数据块修改只是修改了数据块的内容，并不会改变 MHT 的数据结构。假定客户想要将第 $i$ 个数据块 $b_i$ 修改成 $b'$，首先用户端会计算出新的数据块 $b'$ 的验证标签 $\sigma'$ 和索引标签 $\tau^* = H(b'||k_H)$，然后生成一个数据块更新请求信息 $update = \{M, i, b', \tau^*\}$，将其发送给存储对应数据块的存储提供商，$M$ 代表数据块修改操作。用户将需要修改的数据块 $b'$ 的验证标签 $\sigma'$ 发送给第三方验证者，即更新请求 $updateverify = \{M, i, \sigma'\}$。

第三方验证者收到更新请求信息 $updateverify$ 后会立即执行更新验证参数操作 $ExecUpdateVerify(\phi, updateverify)$：

（1）TPA 用新的验证标签，$\sigma'$ 替换掉旧的验证标签 $\sigma_i$，并生成新的验证签名集合 $\phi'$。

（2）TPA 用新的验证签名集合计算验证元数据 $acc_B{}' = acc_B{}^{\frac{(\sigma_i+s)}{(\sigma'+s)}}$。

（3）TPA 用新的验证辅助参数 $aux_1{}' = (acc_B{}', e, g_1, g_2, \phi')$ 替换掉旧的辅助

参数$aux_1$。

（4）TPA 为第 $k$ 个存储提供商重新计算参数$p_k{}'$=$\prod_{\sigma_i \in \phi' \setminus \phi_k{}'}(\sigma_i + s)$。

（5）TPA 将更新后的证明生成辅助参数$aux_2{}' = (g_2, g_2{}^s, \ldots, g_2{}^{s^n}, p_k{}', v_k)$发送给对应的第 $k$ 个存储提供商。

存储提供商收到更新请求信息后，则会立即执行更新操作$ExecUpdate(F, update, aux_2{}')$：

（1）存储提供商用新的数据块$b'$替换掉旧的数据块$b_i$。

（2）存储提供商用新的辅助参数$aux_2{}'$替换掉旧的辅助参数$aux_2$。

（3）存储提供商用新的索引签名$\tau^*$替换掉旧的签名$\tau_i$，并生成新的索引签名集合$\theta_k{}'$。

（4）存储提供商更新 MHT 得到新的根$h_R{}^*$。

经过数据块修改操作的 MHT 结构如图 3.8 所示。

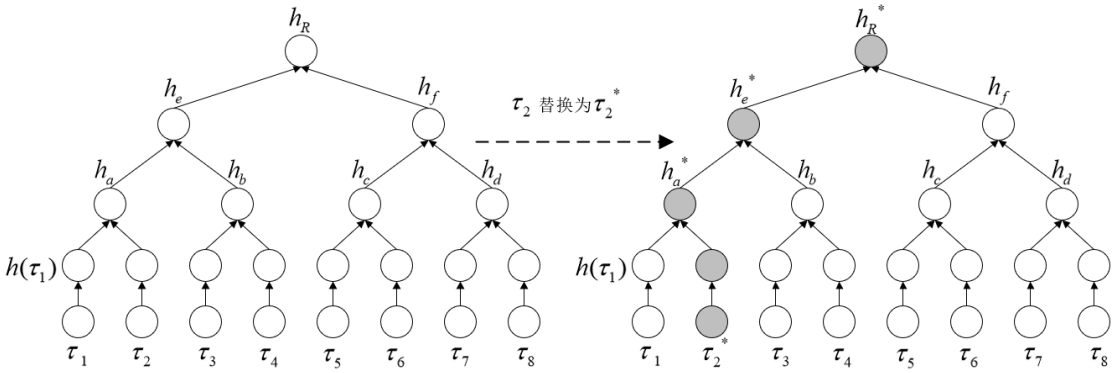


图 3.8 数据块修改

对于数据块插入，数据块插入是指再特定的位置插入新的数据块，相比不会修改 MHT 数据结构的数据块修改操作来说，数据块插入操作会在 MHT 的特定位置添加新的数据块。假定用户想要再第 $i$ 个数据块$b_i$之后插入新的数据块$b'$，更新操作的流程与数据块修改类似。首先用户端会计算出新的数据块$b'$的验证标签$\sigma'$和索引标签$\tau^* = H(b'||k_H)$，然后生成一个数据块更新请求信息$update = \{I, i, b', \tau^*\}$，将其发送给存储对应数据块的存储提供商，$I$ 代表数据块插入操作。用户将需要插入的数据块$b'$的验证标签$\sigma'$发送给第三方验证者，即更新请求$updateverify = \{I, i, \sigma'\}$。

第三方验证者收到更新请求信息$updateverify$后会立即执行更新验证参数操作$ExecUpdateVerify(\phi, updateverify)$：

（1）TPA 添加新的验证标签$\sigma'$到验证标签集合$\phi$并输出一个新的验证签名集合$\phi'$。

（2）TPA 用新的验证签名集合计算验证元数据$acc_B{}'$=$acc_B{}^{(\sigma'+s)}$。

（3）TPA 用新的辅助参数$aux_1{}' = (acc_B{}', e, g_1, g_2, \phi')$替换辅助参数$aux_1$。

（4）TPA 为第 $k$ 个存储提供商重新计算参数$p_k{}'$=$\prod_{\sigma_i \in \phi' \setminus \phi_k{}'}(\sigma_i + s)$。

（5）TPA 将更新后的证明生成辅助参数$aux_2{}' = (g_2, g_2{}^{s}, \ldots, g_2{}^{s^{n+1}}, p_k{}', v_k)$发送给对应的第 $k$ 个存储提供商。

存储提供商收到更新请求信息后，则会立即执行更新操作$ExecUpdate(F, update, aux_2{}')$：

（1）存储提供商会存储新的数据块$b'$。

（2）存储提供商用新的辅助参数$aux_2{}'$替换掉旧的辅助参数$aux_2$。

（3）添加新的数据块$b'$的索引标签$\tau^*$到索引标签集合$\theta_k$并输出一个新的签名集合$\theta_k{}'$。

（4）存储提供商更新 MHT 得到新的根$h_R{}^*$。

经过数据块插入操作的 MHT 结构如图 3.9 所示。图 3.9 介绍了一个数据块插入操作实例，客户想要在第 2 块数据块之后插入一个新的数据块$b'$，节点$h(\tau^*)$和一个内部节点被添加在了 MHT 的数据结构中，内部节点值为$h(h(\tau_2)||h(\tau_2{}^*))$。

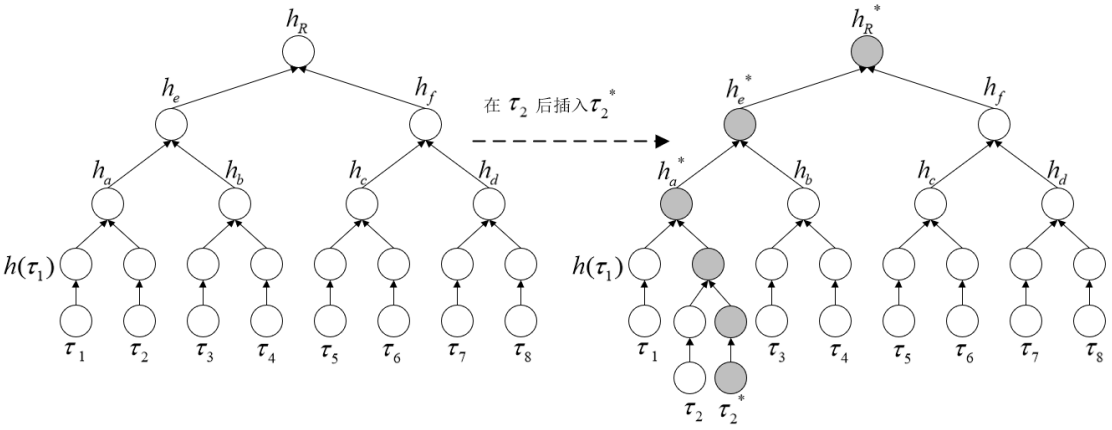


图 3.9 数据块插入

对于数据块删除，数据块删除是指删除特定位置的数据块，与数据块插入类似，同样也会改变 MHT 的数据结构。假定用户想要删除第 $i$ 个数据块$b_i$，客户端生成一个更新请求$update = \{D, i\}$，将其发送给存储对应数据块的存储提供商，$D$ 代表数据块删除操作，用户将需要删除的数据块$b_i$的验证标签$\sigma_i$发送给第三方验证者，即更新请求$updateverify = \{D, i, \sigma_i\}$。

第三方验证者收到更新请求信息$updateverify$后会立即执行更新验证参数操作$ExecUpdateVerify(\phi, updateverify)$：

（1）TPA 删除验证标签$\sigma_i$，并生成新的验证签名集合$\phi'$。

（2）TPA 用新的验证签名集合计算验证元数据$acc_B{}'$=$acc_B{}^{\frac{1}{(\sigma'+s)}}$。

（3）TPA 用新的验证辅助参数$aux_1{}' = (acc_B{}', e, g_1, g_2, \phi')$替换掉旧的辅助参数$aux_1$。

（4）TPA 为第 $k$ 个存储提供商重新计算参数$p_k{}'$=$\prod_{\sigma_i \in \phi' \setminus \phi_k{}'}(\sigma_i + s)$。

（5）TPA 将更新后的证明生成辅助参数$aux_2{}' = (g_2, g_2{}^s, \ldots, g_2{}^{s^{n-1}}, p_k{}', v_k)$发送给对应的第 $k$ 个存储提供商。

存储提供商收到更新请求信息后，则会立即执行更新操作$ExecUpdate(F, update, aux_2{}')$：

（1）存储提供商删除数据块$b_i$。

（2）存储提供商用新的辅助参数$aux_2{}'$替换掉旧的辅助参数$aux_2$。

（3）存储提供商删除数据块$b_i$的索引签名$\tau_i$，并输出新的签名集$\theta_k{}'$。

（4）存储提供商更新 MHT 得到新的根$h_R{}^*$。

经过数据块删除操作的 MHT 结构如图 3.10 所示。图 3.10 介绍了一个数据块删除操作实例，用户想要删除第 2 个块，只有节点$h(\tau_2)$被删除。

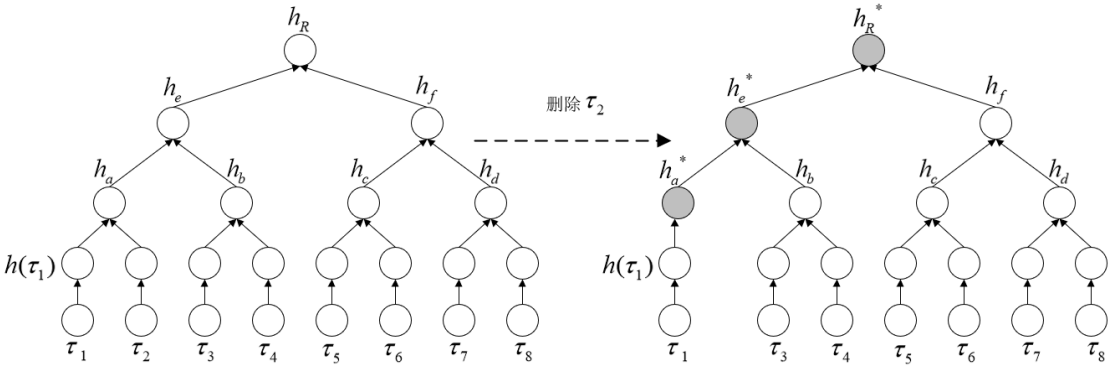


图 3.10 数据块删除

存储提供商会生成一个更新证明$P_{update} = (h(\tau_i), h_R{}^*)$并将其发送给客户端，客户端，客户端收到更新证明信息后，会利用区块链上未修改的 MHT 以及$h(\tau_i)$生成一个新的根$h_R{}'$，并与证明信息中的$h_R{}^*$进行比较，从而判断存储提供商是否按照要求进行了数据块修改操作。如果相同，输出 TRUE，否则证明存储提供商没有按照要求进行数据块更新操作。之后客户端会用私钥对新的根$h_R{}'$进行签名$sig_{sk}(h_R{}')$，并将其发送给存储提供商。存储提供商收到信息后，会更新区块链上的相关信息，包括更新之后的 MHT，$sig_{sk}(h_R{}')$以及索引标签集$\theta_k{}'$。最后客户端会通过比较区块链上的根与计算出的新根是否相等，进而可以保证区块链上信息的正确性。数据更新和验证阶段方案图如图 3.11 所示。

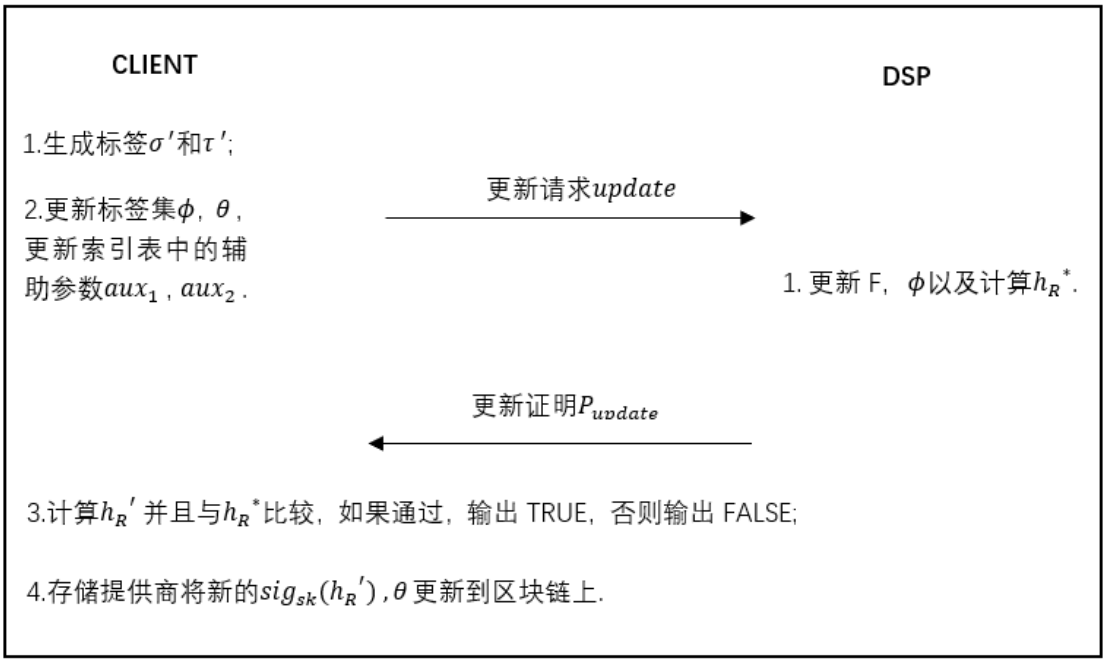


图 3.11 数据块更新和验证