Optimal Data Authentication from Directed Transitive Signatures

Philippe Camacho
Dept. of Computer Science, University of Chile,
Blanco Encalada 2120, Santiago, Chile.
pcamacho@dcc.uchile.cl

August 12, 2011

Abstract

An authenticated dictionary of size N is said to be optimal when an update operation or proof computation requires at most $O(\log N)$ accesses to the data-structure, and the size of a proof is $O(1)$ with respect to N.

In this note we show that an optimal authenticated dictionary (OAD) can be built using transitive signatures for directed graphs (DTS). As the existence of DTS and OAD are both still open, our result can be interpreted as following: if optimal authenticated dictionaries do not exist then transitive signatures for directed graphs do not exist either.

1 Introduction

In an authenticated dictionary the operations to access and update the data involve some guarantee that the answers are consistent with the actual state of the data-structure. We consider the following scenario, also called three party model [GTH02]. In this model a *Source* wants to delegate the access to the data-structure it owns through a *Replica* which may be corrupted. The final *User* interacts with the *Replica* to gain access to the data-structure. In order to avoid the *User* being fooled by the *Replica*, the *Source* publishes some short values that represent the state of the data-structure after each update and extra cryptographic data for the *Replica*. Then using this value and some proof computed by the *Replica*, the *User* gets the guarantee that the answer of the *Replica* is authentic.

All existing constructions for this model, in particular [CHKO08, CKS09, CL02, BGG94, Ngu05, PT10], involve a trade-off between the size of the proof and the number of accesses to update the data-structure/compute the proof. Roughly, if the proof is small, the time to update the data-structure/compute the proof will be large. Table 1 summarizes the complexities of the current constructions. An interesting open problem is to know whether we can build optimal authenticated dictionaries (OAD) where the time to update the data-structure/compute the proof would involve only $O(\log N)$ accesses to the data-structure and the size of the proof would remain constant in N. It seems that actual techniques based only collision resistant hash functions, the RSA crypto-system, bilinear maps and recently lattices [PT10] are not enough to reach such a goal[1]. In this note we present an optimal authenticated dictionary (OAD) based on the existence of transitive signatures for directed graphs (DTS). The existence of such schemes was proposed as a challenging open problem in [MR02]. So our result does not solve any of these two open problems, but establishes the following relation between these two primitives.

Theorem 1. *$DTS \rightarrow OAD$*

As a corollary, this result provides a possible path to prove the non-existence of DTS by showing a lower bound $\omega(\log N)$ on the number of accesses to update the data-structure (or compute the proof) while keeping the size of the proof $O(1)$.

RELATED WORK. Transitive signatures first appeared in [MR02] where the authors proposed an efficient implementation for non-directed graphs based on groups where the discrete logarithm is hard. Other constructions - still for undirected graphs - based on RSA-related assumptions and bilinear maps appeared in [BN05, SSM05]. Hohenberger in [Hoh03] showed that the existence of DTS is a very challenging problem as it would imply the existence of a group for which the internal law can be computed efficiently but where the inversion is hard. Directed transitive signatures for trees have been studied in [Yi07, Nev08]. However these solutions, as noted by Neven, do not fall strictly into the definition of DTS as the size of signature grows with the number of nodes of the tree.

In a similar work to ours [PTT10] it is shown that using multi-linear forms [BS02], an optimal dictionary could be built in the two-party model[2]. As no concrete implementation for multi-linear forms is known to the date, an impossibility result for an OAD in the two-party model would invalidate the complexity assumption for multi-linear forms which was proposed in [PTT10].

2 Preliminaries

In the following $||$ denotes the concatenation operator. A function $f : \mathbb{N} \rightarrow \mathbb{R}$ is said to be negligible if for every polynomial $p(.)$ there exists n_0 such that $\forall n > n_0 : f(n) < 1/p(n)$. If κ is the security parameter then an algorithm is said to be PPT if it is probabilistic and runs in polynomial time in κ.

$G = (V, E)$ denotes a graph G with set of vertices V and set of edges E. If $P \in V$ is a vertex of G, then P^* is the transitive closure of P, that is the set of vertices: $\{Q \in V :$ there exist a path from P to Q $\}$. Similarly the transitive closure of a graph G, denoted G^* is the union of the transitive closures of every vertex of G. A path from node R to node H is written $R \rightarrow H$. In the case that G is a balanced binary tree we consider the following conventions. The depth d of a node N is the number of edges on a path from the root R to N. In particular the depth for the root R is 0. Let R be the root of the tree. If $b_i \in \{0, 1\}$, and $0 \leq i \leq l \leq d-1$, then $R_{b_0 b_1 \ldots b_l}$ denotes the node reachable from R by taking the path $b_0 b_1 \ldots b_l$, where 0 means left child and 1 means right child. By $R_{[b_0 b_1 \ldots b_l]}$ we denote the path from R to $R_{b_0 b_1 \ldots b_l}$ formed by the nodes

[1]Obviously lattices still are a promising path as they seem in some aspect more powerful than bilinear maps or other more classical primitives after Gentry's breakthrough [Gen09].

[2]In this model, the server that has limited storage capacity, interacts with an untrusted memory that handles the data-structure. See [GTH02] for more details.

Work	Assumption	T_{Upd}	$T_{ProofGen}$	T_{Verif}
[CHKO08]	CHRF	$O(\log N)$	$O(\log N)$	$O(\log N)$
[CL02]	S-RSA	$O(\epsilon \cdot N^{\frac{1}{\epsilon}})$	$O(\epsilon \cdot N^{\frac{1}{\epsilon}})$	$O(\epsilon)$
[CL02]	S-RSA	$O(N \log N)$	$O(1)$	$O(1)$
[CKS09]	q-DHE	$O(\epsilon \cdot N^{\frac{1}{\epsilon}})$	$O(\epsilon \cdot N^{\frac{1}{\epsilon}})$	$O(\epsilon)$
[BGG94]	D-Log	$O(1)$	$O(1)$	$O(N)$
[Ngu05]	q-SDH	$O(N^2)$	$O(N^2)$	$O(1)$
[PT10]	GAPSVP	$O(1)$	$O(\log N)$	$O(\log N)$

Figure 1: Tradeoff for authenticated dictionaries.
Complexities are relatives to the number of accesses to the data-structure. T_{Upd} is the time to update the data-structure, $T_{ProofGen}$ the time to compute a proof and T_{Verif} is the time to check the proof. The abbreviations *CHRF, S-RSA, q-DHE, q-SDH, GAPSVP* stand respectively for *Collision-Resistant Hash Hunction families, Strong RSA Assumption, q-Exponent Diffie-Hellman Assumption*, *q-Strong Diffie-Hellman Assumption* and the *Gap Version of the Shortest Vector Problem in lattices.*

$R, R_{b_0}, R_{b_0b_1}, ..., R_{b_0b_1...b_{l-1}}, R_{b_0b_1...b_l}$. If we index the leaves from left to right starting from 0 until $N-1 = 2^d - 1$ we note that L_i, the i-th leaf of the tree with root R is such that $L_i = R_{b_0b_1...b_{d-1}}$ where $i = \Sigma_{j=0}^{d-1} b_{d-1-j}2^j$. In other words, $(b_0, ..., b_{d-1})$ is the binary decomposition of i where b_0 is the bit of strongest weight. We consider val, the function that "extracts" the value v contained a in message M parsed as $M = i||v||r$. More precisely, val takes as argument a message of the form $i||v||r$ and returns v.

(Transitive) Signatures. We denote by $\mathfrak{SSig} = (\mathsf{SKG}, \mathsf{SSig}, \mathsf{SVf})$ a standard signature scheme. $(sk, pk) \leftarrow \mathsf{SKG}(1^\kappa)$ is the pair of private/public keys created at the beginning of the scheme. Then for a message $M \in \{0,1\}^*$ its associated signature is $\sigma_M = \mathsf{SSig}(sk, M)$. The validation of a signature σ on M is done by running $\mathsf{valid}, \perp \leftarrow \mathsf{SVf}(pk, M, \sigma)$ where valid means σ is valid and $\perp$ means the opposite. For common digital signatures, we use the standard notion of existential unforgeability under chosen message attack presented in [GMR88]. In the experiment the adversary $\mathcal{A}$ is given a public key from the signing oracle. Then $\mathcal{A}$ asks the oracle to sign a polynomial number of messages he chose. At the end of the game $\mathcal{A}$ outputs a pair (M', σ) such that M' has not been signed by the oracle. The scheme is said to be secure if the probability that $\mathsf{SVf}(pk, M'\sigma) = \mathsf{valid}$ is negligible in κ for any PPT *Adversary.* In a transitive signature scheme, the signer can sign the vertices of some graph but also the edges. Then without the secret, these edge signatures allow to compute the signature for any path that is in the transitive closure of the graph.

Definition 1. *(Transitive Signature Scheme, [MR02, Nev08]) A transitive signature scheme (for directed graph) is a tuple* $\mathfrak{DTS} = (\mathsf{TSKG}, \mathsf{TSign}, \mathsf{TSComp}, \mathsf{TSVf})$ *where*

- $\mathsf{TSKG}(1^\kappa)$: *returns a pair of private/public keys* (tsk, tpk)*.*
- $\mathsf{TSign}(tsk, i, j)$: *returns* $\tau_{(i,j)}$*, the signature of an edge* (i, j)*. Note that signing an edge*

means implicitly that every vertex is also signed. This can be done with a standard signature scheme.

- $\mathsf{TSComp}((i,j),\tau_{(i,j)},(j,k),\tau_{(j,k)},tpk)$*: returns the signature* $\tau_{(i,k)}$ *of the edge* (i,k)*. Note that the secret key is not required, neither some description of the current graph.*

- $\mathsf{TSVf}((i,j),\tau,tpk)$: *returns* valid *if the* τ *is a valid signature for the edge* (i,j) *and* $\perp$ *otherwise.*

Note that in this definition the time of execution of TSComp should not depend on previous computations, namely the size of the current graph. So we say that a transitive signature scheme is admissible if there exists a fixed polynomial $p(\cdot)$ such that all the algorithms of $\mathfrak{DTS}$ run in time bounded by $p(\kappa)$.

A transitive signature scheme is said to be secure if there is no adversary $\mathcal{A}$ that can compute of signature of a path (i,j) that does not lie in the transitive closure of the graph.

Definition 2. *(Security for transitive signatures schemes [Nev08]) Let* $\mathfrak{DTS}$ *be a transitive signature scheme. Consider the following experiment: the adversary* $\mathcal{A}$ *is given the public parameters of the scheme by running* TSKG*. Then* $\mathcal{A}$ *asks for edge signatures to the signing oracle* $\mathsf{TSign}(tsk,\cdot,\cdot)$*. After a polynomial number of queries* $\mathcal{A}$ *finally outputs* (i,j,τ)*. The set of signed edges form the (directed) graph* $G=(V,E)$*. The advantage of* $\mathcal{A}$ *is defined by*

$$Adv^{tuf-cma}_{\mathfrak{DTS},\mathcal{A}}(\kappa)=\Pr\left[\,\mathsf{TSVf}((i,j),\tau,tpk)=\mathsf{valid}\wedge(i,j)\notin G^*\,\right]$$

where the probability is taken over the coins of the adversary and the algorithms of the scheme. $\mathfrak{DTS}$ *is said to be transitively unforgeable under chosen message attack if for any PPT* $\mathcal{A}$*,* $Adv^{tuf-cma}_{\mathfrak{DTS},\mathcal{A}}(\kappa)$ *is negligible.*

Data authentication. An authenticated dictionary D in the three-party model works as the following. The owner of the dictionary or *Source* publishes the data-structure with some additional cryptographic information. Moreover the *Source* also publishes a short value corresponding to the current state of the dictionary. With this short value a *User* can ask to a *Replica* for a proof that $D[i]=v$ for some index i and value v. The security requirement states that the *Replica* should only be able to compute that proof if indeed $D[i]=v$, otherwise the verification algorithm run by the *User* should detect the forgery. When the dictionary is updated, the *Source* must publish a new state and also the necessary data to allow proof computations by the *Replica*. Next we define formally the syntax, correctness and security for an authenticated dictionary.

Definition 3. *(Authenticated dictionary in the 3-party model)*

Let $\kappa\in\mathbb{N}$ *be the security parameter. An authenticated dictionary scheme* $\mathfrak{AuthDict}$ *consists of the following algorithms.*

- $\mathsf{Setup}(1^\kappa,N)$*: this probabilistic algorithm takes* κ *in unary as input and* N*, a bound of the number of elements of the dictionary* D*. It returns a pair of public and private keys* (PK,SK)*, the initial auxiliary information of the data-structure* $\mathfrak{m}_0$ *and a short value* $\hat{\mathfrak{m}}_0$

that represents the state of $\mathfrak{m}_0$*. Note that the auxiliary information is public*[3]*. The* $\mathfrak{m}_0$ *data-structure represents a dictionary* D *which set of keys is* $\{0, 1, ..., N-1\}$*.*

- $\mathsf{Verify}(i, v, \Pi, PK, \hat{\mathfrak{m}})$*: given an index* i*, a value* v*, a proof* Π*, the state* $\hat{\mathfrak{m}}$ *and the public key* PK*, return* valid *meaning that* $D[i] = v$*, or* $\perp$ *otherwise. This algorithm is run by a* User *.*

- $\mathsf{ProofGen}(i, \mathfrak{m}, PK)$*: this algorithm returns a proof* Π *associated to the value* v *and index* i *such that* $D[i] = v$ *where* D *is represented by* $\mathfrak{m}$*. This algorithm is run by the* Replica*.*

- $\mathsf{Update}(i, v, \mathfrak{m}_{before}, PK, SK)$*: this algorithm computes the new state of the auxiliary data-structure* $\mathfrak{m}_{after}$ *where* $D[i] = v$ *and the rest of the dictionary remains unchanged. Moreover the auxiliary information* $\mathfrak{m}_{before}$ *is also updated to* $\mathfrak{m}_{after}$ *to allow further proof computations. This algorithm is run by the* Source*.*

An authenticated dictionary is said to be correct if the *Replica* is always able to compute valid proofs for pairs (i, v) such that $D[i] = v$.

Definition 4. *(Correctness) Let* $\mathfrak{m}_1, \mathfrak{m}_2, ..., \mathfrak{m}_h$ *be a sequence of updates, then we say the scheme is correct if for any* $0 \le i \le N-1$ *and any* $1 \le j \le h$ *we have that*
$\Pr[\Pi = \mathsf{ProofGen}(i, \mathfrak{m}_j, PK) \wedge \mathsf{Verify}(i, v, \Pi, PK, \hat{\mathfrak{m}_j}) = \mathsf{valid}] = 1$ *where the probability is taken over the random coins of the algorithms of the scheme.*

The authenticated dictionary is said to be secure if the probability for any PPT $\mathcal{A}$ to compute a proof Π for a pair (i, v') that passes the verification step, and such that $D[i] \neq v'$, is negligible.

Definition 5. *(Security for authenticated data-structures [CL02])*
Let $\mathfrak{AuthDict}$ *be an authenticated dictionary. We consider the notion of security denoted* $uf-ad$ *described by the following experiment: on input the security parameter* κ*, the adversary* $\mathcal{A}$ *has access to an oracle* $\mathcal{O}(\cdot)_{AD}$ *that replies to queries by playing the role of the* Source*. Using the oracle, the adversary asks for updates to the dictionary a polynomial number of times. The oracle* $\mathcal{O}(\cdot)_{AD}$ *replies with the new state of the data-structure and also the necessary data to update the auxiliary information of the data-structure. Recall that, as the auxiliary information is public, the adversary can compute a proof for any index of the table. Finally, the adversary is required to output a tuple* (i, v, Π)*.*
The advantage of the adversary $\mathcal{A}$ *is defined by:*

$$Adv^{uf-ad}_{\mathfrak{AuthDict},\mathcal{A}}(\kappa) \quad = \quad \Pr[\mathsf{Verify}(i, v, \Pi, \hat{\mathfrak{m}}, PK) = \mathsf{valid} \wedge D[i] \neq v]$$

where PK *is the public key generated by* Setup*, and* $\mathfrak{m}$ *is the state of the dictionary* D *at the end of the experiment. The scheme* $\mathfrak{AuthDict}$ *is said to be secure if for every PPT adversary* $\mathcal{A}$*,* $Adv^{uf-ad}_{\mathfrak{AuthDict},\mathcal{A}}(\kappa)$ *is negligible.*

We need now to define what does the expression *optimal authenticated dictionary* mean. The intuition is simply that the time to perform an operation in an optimal authenticated dictionary should not be much larger than the time required to run a non-authenticated dictionary.

[3]In some works, in particular [CL02], *auxiliary information* has another meaning and is secret.

However precision is required as the size of an authenticated dictionary is related to the security parameter, that is $N = q(\kappa)$, where N is the size of the authenticated dictionary and $q(\cdot)$ is a polynomial.

Our definition for authenticated dictionary differs a bit from the one of [PTT10] because it explicitly mentions the security parameter κ and allows the Setup algorithm to run in polynomial time in N. However both definitions are in essence equivalent.

Definition 6. *Let N be the size of the dictionary. An authenticated dictionary $\mathfrak{AuthDict}$ scheme is optimal if and only if:*

- *It is correct and secure under definitions 4 and 5 respectively.*
- *There exist two polynomials $m(\cdot), l(\cdot)$ such that:*
  - Setup *runs in $O(m(\kappa) \cdot l(N))$ time.*
  - Verify *runs in $O(m(\kappa))$ time and the proof Π has size $O(\kappa) = O(1)$ w.r.t N.*
  - ProofGen *runs in $O(m(\kappa) \cdot \log(N))$ time, and involves at most $O(\log N)$ accesses to the dictionary.*
  - Update *runs in $O(m(\kappa) \cdot \log(N))$, and involves at most $O(\log N)$ accesses to the dictionary.*

Let us comment and justify the definition. As $N = q(\kappa)$ for some polynomial $q(\cdot)$, then an authenticated data-structure will be slower than the un-authenticated data-structure only by a factor of $m(\kappa)$ which is fixed and does not depend on N. Moreover the number of accesses to the dictionary will remain the same. In practice $m(\kappa)$ is the time to perform some cryptographic operation (hash, signature,...), so it is reasonable to consider that $m(\kappa) \ll N = q(\kappa)$. Note that the trivial construction like the Merkle trie [**?**], does not fit this definition. The reason is that although the algorithms Verify, ProofGen, Update require $O(m(\kappa))$ time where $m(\kappa) = \kappa^2$, the size of the proof is $O(\kappa^2) = \omega(\kappa)$ and the number of accesses to the data-structure to compute a proof is $O(\kappa) = \omega(\log(N))$.

3 Our construction

The idea of our construction is to build a balanced binary tree of $N = 2^d$ leaves, where each leaf $L_i : 0 \leq i \leq N-1$ will store the value $D[i]$ of the dictionary. The edges will be signed by the $\mathfrak{DTS}$ scheme and the root R^h will "represent" D at some point of time h. To perform an update for the index j, which corresponds to the leaf L^h_j (that is the leaf at position j reachable from root R^h) a new root R^{h+1} is created. Along with this new root, a parallel path from $P^h = R^h_{[b_0 b_1 \dots b_{d-1}]}$ the old root R^h to the leaf L^h_i is created. That is $P^{h+1} = R^{h+1}_{[c_0 c_1 \dots c_{d-1}]}$ where $\forall i : 0 \leq i \leq d-1 : c_i = b_i$. Each internal node in P^{h+1} is assigned a random label and the leaf $R^{h+1}_{c_0 c_1 \dots c_d} = L^{h+1}_j$ will contain the new value v such that $D[j] = v$. Then this new path will be linked to the other nodes of the previous tree to maintain consistency of the dictionary, in the sense that except for the index $k = j$, all the leaves L^h_k must remain reachable from the new root R^{h+1}. This is done simply by connecting each internal node of the new path to the corresponding sibling of the node belonging to the old path. Again, every new edge is signed using a transitive signature scheme. So now, proving that $D[i] = v$ is equivalent to exhibit a signature for the path from the current root R^h to a leaf L_i^h such that $val(L) = v$. We can then note that the time to update the balanced binary tree requires $O(\log N)$ signature computations and the proof consists only in a single signature of a path. The security of the construction is based on the fact that the only directed path from a root R^h to a given leaf L_i^h will always correspond to the current value of the dictionary at time h and index i.

Definition 7. *(Authenticated Dictionary from DTS)*
Let $\mathfrak{AuthDict}_{DTS}$ be the authenticated dictionary defined by the following algorithms.

- $\mathsf{Setup}(1^\kappa, N)$*: generates the parameters (tsk, tpk) for the a directed transitive signature scheme $\mathfrak{DTS}$. Then we set $SK = tsk$ and $PK = tpk$. Let N be the size of the dictionary D, and assume w.l.o.g. that N is a power of 2. Build a balanced binary tree $\mathfrak{m}_0$ where each leaf at position $0 \le i \le N-1$ has got the value $M = i||v||r$ with r a random value* [4]*. Every internal node is filled with a random value. Then sign the tree with $\mathfrak{DTS}$ scheme. Publish the root node $\hat{\mathfrak{m}}_0$ as the state of the dictionary.*

- $\mathsf{Verify}(i, v, \Pi, PK, \hat{\mathfrak{m}})$*: parse Π as (σ_l, σ_p), then extract from $\hat{\mathfrak{m}}$ the root node R and verify that σ_l is the signature of message $M = i||v||r$ for some (random) value r. Then check that σ_p is a valid signature for the path $R \to M$.*

- $\mathsf{ProofGen}(i, \mathfrak{m}, PK)$*: using $\mathfrak{m}$ compute the digital signature σ_p for the path $R \to M$ where R is the root of the tree and M is the value at the i^{th} leaf, i.e. $M = i||v||r$ where r is random. Return (σ_p, σ_l) with $\sigma_l = \mathsf{TSign}(M)$.*

- $\mathsf{Update}(i, v, \mathfrak{m}, PK, SK)$*:*
  - *Create a new leaf L_i^{h+1} with value $M = i||v||r$ for a random r. Compute the signature $\sigma_{L_i^{h+1}} = \mathsf{SSig}(tsk, M)$.*
  - *Let $R_{[b_1 b_2 \ldots b_d]}$ be the path from the root $R_h \in G$, the graph corresponding to $\mathfrak{m}$, to leaf L_i^h. Create a new root node R^{h+1} and new intermediate nodes $R_{b_0}^{h+1}, R_{b_0 b_1}^{h+1}, \ldots, R_{b_0 \ldots b_d - 1}^{h+1} = L_i^{h+1}$. All these new nodes contain random values and are signed. The edges $(R^{h+1}, R_{b_0}^{h+1})$, $(R_{b_0}^{h+1}, R_{b_0 b_1}^{h+1})$, $(R_{b_0 b_1}^{h+1}, R_{b_0 b_1 b_2}^{h+1})$, ..., $(R_{b_0 b_1 \ldots b_{d-2}}^{h+1}, R_{b_0 b_1 b_2 \ldots b_{d-1}}^{h+1})$ are signed too using the algorithm* TSign*.*
  - *Then the nodes of this path are connected (by signing edges) to the siblings of the old path that has been replaced. That is for every $0 \le i \le d-1$ sign edges $(R_{b_0 \ldots b_i}^{h+1}, R_{b_0 \ldots b_i b_{1-b_{i+1}}}^{h})$ and also $(R^{h+1}, R_{1-b_0}^h)$.*
  - *Publish the new value $\hat{\mathfrak{m}_{after}} = R^{h+1}$ as the new state of the dictionary.*

[4]Indeed it would be sufficient to handle vertices labels of the form $i||v||c$ where c is a counter as to guarantee that all labels are distinct. Note that this method would make the scheme stateful and also leaks some information about the history of updates of the dictionary.

To prove correctness and the security of our construction we need the following lemma.

Lemma 1. *Let $\{R^h : h \in \mathbb{N}\}$ denotes the sequence of root nodes generated by the consecutive executions of the* Update *algorithm. Then with probability at least $P(q) = 1 - \frac{q^2}{2^{\kappa+1}}$, the two following properties hold:*

1. *$\forall h \in \mathbb{N}$, $(R^h)^*$ is a directed balanced binary tree with N leaves $(L_0^h, L_1^h, ..., L_{N-1}^h)$.*
2. *if D^h denotes the state the dictionary after the same sequence of updates, then $\forall i : 0 \leq i \leq N-1 : D^h[i] = val(L_i^h)$.*

where q' is the number of graph nodes created during the successive updates.

Proof. We assume first that every node is filled with different values.

By induction on h. For $h = 0$ the claim is verified. Assume that for any $h \in \mathbb{N}$, R_h^* is a directed balanced binary tree with N leaves and that $\forall 0 \leq i \leq N-1 : D^h[i] = val(L_i^h)$. After the update $(R^{h+1})^*$ is also a directed binary tree because it is formed by the new nodes of the path $R_{[b_1 ... b_d]}^{h'+1}$ that are connected to the nodes of $(R^h)^*$ which are by induction root of balanced directed binary trees of depth $d-1, d-2, ..., 0$ respectively. Let i be the index of the dictionary that is updated. Then we can also see that $\forall 0 \leq j \leq N-1 \wedge j \neq i : D^h[j] = val(L_j^{h+1})$ as the only new nodes that are reachable from R^{h+1} are those on the new path $R_{b_1 ... b_d}^{h+1}$ where $i = \Sigma_{j=0}^{d-1} b_{d-j} 2^j$. The other nodes, including the leaves (except L_i^{h+1}) are still reachable from R^h. So we have that the set of leaves that are reachable from R^{h+1} is formed by the leaves reachable from R^h except for L_i^h that is replaced by the new leaf L_i^{h+1}. Thus, if $(R^h)^*$ was the tree such that $\forall i, 0 \leq i \leq N-1 : D^h[i] = val(L_i^h)$, we can deduce that the leaves of $(R^{h+1})^*$ represent D^{h+1}.

These two properties hold as long as all values in each node of the graph $G = G_1 \cup G_2 \cup \cdots \cup G_{h+1}$ are different. As by construction these values are chosen randomly in a universe of size 2^κ, we have that $P(q)$ the probability that every two values are different after q nodes creation is such that $1 - P(q) \leq \sum_{i=1}^{q} \frac{1}{2^\kappa} \leq \frac{q^2}{2^{\kappa+1}}$. Thus both properties are true with probability $P(q) \geq 1 - \frac{q^2}{2^{\kappa+1}}$.

Note finally that the fact that the graph $G = G_1 \cup G_2 \cup \cdots G_{h+1}$ is directed is essential as otherwise every node would be reachable from any root R^j, thus invalidating the lemma.

□

From the lemma we can directly deduce that,

Proposition 1. *The authenticated dictionary $\mathfrak{AuthDict}_{DTS}$ is correct.*

Proposition 2. *Let ϵ be the advantage of an adversary $\mathcal{A}$ for the $\mathfrak{AuthDict}$, and q_{AD} the number of queries made to the oracle $\mathcal{O}_{AD}$ by $\mathcal{A}$. Then we can build an adversary $\mathcal{B}$ such that*

$$Adv_{\mathfrak{DTS},\mathcal{A}}^{tuf-cma}(\kappa) \geq \epsilon(1 - \frac{q_{AD}^2}{2^\kappa})$$

Proof. Assume that there exist a PPT adversary $\mathcal{A}$ that breaks our scheme, then we build the the following adversary $\mathcal{B}$ that breaks the security of the transitive signature scheme $\mathfrak{DTS}$.

$\mathcal{B}$ obtains the public parameters of the $\mathfrak{DTS}$ scheme. With these parameters its computes the parameters of the $\mathfrak{AuthDict}$ scheme and sends them to $\mathcal{A}$. Then for each query of $\mathcal{A}$, $\mathcal{B}$ replies using the signing oracle of $\mathsf{TSign}(sk, \cdot, \cdot)$, the Update and $\mathsf{ProofGen}$ algorithm. At the end of the challenge for $\mathfrak{AuthDict}$, the adversary $\mathcal{A}$ sends a tuple (i, v, Π) such that $\mathsf{Verify}(i, v, \Pi, \mathfrak{m}^h, PK) = \mathsf{valid} \wedge D^h[i] \neq v$. As $D^h[i] \neq v$, from the lemma we can deduce that, with probability at least $1 - \frac{q_{AD}^2}{2^\kappa}$, there is no leaf L reachable from R^h such that $val(L) = v$ and L is in position i. This means that the adversary $\mathcal{B}$ has been able forge a signature σ for $\mathfrak{DTS}$.

$\square$

With respect to the complexity we can observe that the update algorithm requires $O(\log N)$ signature computations of the DTS scheme, the proof computation consists in combining $O(\log N)$ DTS signatures and verifying the value of a dictionary at some index requires a DTS signature verification. Thus, if DTS is an admissible transitive signature scheme for directed graphs, the authenticated dictionary is optimal and theorem 1 is proved.

References

[BGG94] Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. Incremental Cryptography: The Case of Hashing and Signing. In Yvo G. Desmedt, editor, *CRYPTO 1994*, volume 839 of *LNCS*, pages 216–233–233. Springer-Verlag, July 1994.

[BN05] Mihir Bellare and Gregory Neven. Transitive Signatures: New Schemes and Proofs. *IEEE Transactions on Information Theory*, 51(6):2133–2151, June 2005.

[BS02] Dan Boneh and Alice Silverberg. Applications of Multilinear Forms to Cryptography. http://eprint.iacr.org/2002/080, 2002.

[CHKO08] Philippe Camacho, Alejandro Hevia, Marcos Kiwi, and Roberto Opazo. *Strong Accumulators from Collision-Resistant Hashing.* 2008.

[CKS09] Jan Camenisch, Markulf Kohlweiss, and Claudio Soriente. An Accumulator Based on Bilinear Maps and Efficient Revocation for Anonymous Credentials. In Stanisaw Jarecki and Gene Tsudik, editors, *PKC 2009*, volume 5443 of *Irvine*, pages 481–500. Springer Berlin / Heidelberg, 2009.

[CL02] Jan Camenisch and Anna Lysyanskaya. Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 61–76. Springer-Verlag, 2002.

[Gen09] Craig Gentry. *Fully homomorphic encryption using ideal lattices.* ACM Press, New York, New York, USA, May 2009.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. *SIAM Journal on Computing*, 17(2):281, April 1988.

[GTH02] Michael T. Goodrich, Roberto Tamassia, and Jasminka Hasic. An Efficient Dynamic and Distributed Cryptographic Accumulator. In *ISC '02 Proceedings of the 5th International Conference on Information Security*, pages 372–388, September 2002.

[Hoh03] Susan Hohenberger. The Cryptographic Impact of Groups with Infeasible Inversion. http://groups.csail.mit.edu/cis/theses/hohenberger-masters.ps, 2003.

[MR02] Silvio Micali and Ronald Rivest. Transitive Signature Schemes. In Bart Preneel, editor, *CT-RSA 2002*, volume 2271 of *LNCS*, pages 236–243. Springer / Berlin Heidelberg, February 2002.

[Nev08] Gregory Neven. A simple transitive signature scheme for directed trees. *Theoretical Computer Science*, 396(1-3):277–282, May 2008.

[Ngu05] Lan Nguyen. Accumulators from Bilinear Pairings and Applications. In Alfred Menezes, editor, *Topics in Cryptology CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 275–292–292, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[PT10] Charalampos Papamanthou and Roberto Tamassia. Update-Optimal Authenticated Structures Based on Lattices. http://eprint.iacr.org/2010/128, 2010.

[PTT10] Charalampos Papamanthou, Roberto Tamassia, and Nikos Triandopoulos. *Optimal Authenticated Data Structures with Multilinear Forms*, volume 6487 of *Lecture Notes in Computer Science*, pages 246–264–264. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[SSM05] Siamak Fayyaz Shahandashti, Mahmoud Salmasizadeh, and Javad Mohajeri. A Provably Secure Short Transitive Signature Scheme from Bilinear Group Pairs. In Carlo Blundo and Stelvio Cimato, editors, *Security in Communication Networks*, volume 3352 of *LNCS*, pages 60–76. Springer / Berlin Heidelberg, 2005.

[Yi07] Xun Yi. Directed Transitive Signature Scheme. In Masayuki Abe, editor, *CT-RSA*, volume 4377 of *LNCS*, pages 129–144. Springer Berlin / Heidelberg, 2007.