Analysis of a Proposed Hash-Based Signature Standard, rev. 4

Jonathan Katz*

Abstract

We analyze the concrete security provided by a signature scheme described in a recent Internet Draft by McGrew and Curcio.

1 Overview

McGrew and Curcio [6] recently proposed the *LMS scheme* for hash-based digital signatures. The proposed construction instantiates Merkle's tree-based approach [7, 8] with a one-time signature scheme (called the *LM-OTS scheme*) based on work of Lamport, Diffie, Winternitz, and Merkle [4, 7, 8] plus modifications proposed by Leighton and Micali [5] as suggested by [2]. Here, we analyze the concrete security of the LM-OTS scheme in the *multi-instance setting*, where multiple public keys are generated and an attack is successful if it results in a forged signature with respect to any of those keys. This, in turn, is used to analyze the concrete security of the full LMS scheme.

2 Description of the LM-OTS Scheme

We begin with a detailed description of the LM-OTS scheme, following [6]. Let $H : \{0,1\}^* \rightarrow \{0,1\}^{8n}$ be a function that we will treat in our analysis as a random oracle. Fix $w \in \{1, 2, 4, 8\}$ as a parameter of the scheme, and set $e \stackrel{\text{def}}{=} 2^w - 1$. Set $u \stackrel{\text{def}}{=} 8n/w$; note that the output of H can be viewed as a sequence of u integers, each w bits long. Set $v \stackrel{\text{def}}{=} \lceil \lfloor \log u \cdot (2^w - 1) + 1 \rfloor / w \rceil$, and $p \stackrel{\text{def}}{=} u + v$. Define a function $\mathsf{checksum} : (\{0,1\}^w)^u \rightarrow \{0,1\}^{wv}$ as follows:

$$\mathsf{checksum}(h_0, \ldots, h_{u-1}) \stackrel{\text{def}}{=} \sum_{i=0}^{u-1} (2^w - 1 - h_i),$$

where each $h_i \in \{0,1\}^w$ is viewed as an integer in the range $\{0, \ldots, 2^w - 1\}$ and the result is expressed as an integer using exactly wv bits.[1] For positive integers i, b with $i < 2^{8b}$, we let $[i]_b$ denote the b-byte representation of i. For a string s and positive integer j, set $H_s^0(x; j) \stackrel{\text{def}}{=} x$. For positive integers $i \geq 1$ and j, define

$$H_s^i(x; j) \stackrel{\text{def}}{=} H\left(H_s^{i-1}(x; j + i - 2), s, [j + i - 1]_1, \texttt{0x00}\right).$$

*Dept. of Computer Science, University of Maryland. Email: `jkatz@cs.umd.edu`. Work performed under a consultancy agreement with University Technical Services, Inc. on behalf of the National Security Agency. We thank Laurie E. Law and Jerome A. Solinas for suggesting we write this paper.

[1] In [6] the result is expressed as a 16-bit integer, but only the top wv bits are used.

Define the LM-OTS scheme as follows:

Key-generation algorithm Gen

Key generation takes as input $\mathsf{id} = (I, q)$, where I is a 31-byte *identifier* and q is a 4-byte *diversification factor*. The algorithm proceeds as follows:

1. Choose p uniform values $x_0, \ldots, x_{p-1} \in \{0,1\}^{8n}$.
2. For $i = 0$ to $p-1$, compute $y_i := H^e_{\mathsf{id},[i]_2}(x_i; 0)$.
3. Compute $pk := H(\mathsf{id}, y_0, \ldots, y_{p-1}, \mathtt{0x01})$.

The public key is pk, and the private key is $sk = (x_0, \ldots, x_{p-1})$.

Signing algorithm Sign

Signing takes as input a private key $sk = (x_0, \ldots, x_{p-1})$ and a message $M \in \{0,1\}^*$ as usual, as well as $\mathsf{id} = (I, q)$ as above. It does:

1. Choose uniform $C \in \{0,1\}^{8n}$.
2. Compute $Q := H(M, C, \mathsf{id}, \mathtt{0x02})$ and $c := \mathsf{checksum}(Q)$. Set $V := Q \| c$, and parse V as a sequence of w-bit integers $V_0, \ldots, V_{p-1}$.
3. For $i = 0, \ldots, p-1$, compute $\sigma_i := H^{V_i}_{\mathsf{id},[i]_2}(x_i; 0)$.
4. Return the signature $\sigma = (C, q, \sigma_0, \ldots, \sigma_{p-1})$.

Verification algorithm Vrfy

Verification takes as input a message $M \in \{0,1\}^*$ and a signature $(C, q, \sigma_0, \ldots, \sigma_{p-1})$ as usual, as well as I as above. It sets $\mathsf{id} = (I, q)$ and does:

1. Compute $Q := H(M, C, \mathsf{id}, \mathtt{0x02})$ and $c := \mathsf{checksum}(Q)$. Set $V := Q \| c$, and parse V as a sequence of w-bit integers $V_0, \ldots, V_{p-1}$.
2. For $i = 0, \ldots, p-1$, compute $y_i := H^{e-V_i}_{\mathsf{id},[i]_2}(\sigma_i; V_i)$.
3. Output $H(\mathsf{id}, y_0, \ldots, y_{p-1}, \mathtt{0x01})$.

We note that, in contrast to the usual convention, Vrfy outputs a string rather than a bit and does not take a public key as input. A signature σ on some message M is valid relative to some fixed public key pk if the output of Vrfy is equal to pk.

One can verify that correctness holds in the following sense: for any I, q, any (pk, sk) output by $\mathsf{Gen}(I, q)$, and any message M, we have $\mathsf{Vrfy}(M, \mathsf{Sign}(sk, M, I, q), I) = pk$.

3 Security of the LM-OTS Scheme

We adapt the standard notion of security for one-time signature schemes (see [3]) to the multi-instance setting, where multiple (independent) instances of the scheme are run and the attacker is considered successful if it generates a signature forgery with respect to any of those instances. (Any scheme secure in the usual sense is also secure in the multi-instance setting. Here, though, we are interested in a tighter security bound than is implied by that fact.) In addition, we also explicitly handle the values I, q used as an additional input to the various algorithms of the scheme.

If values $\mathsf{id} = (I, q)$ are used for key generation in some instance of the scheme, we refer to id as the *identifier* for that instance. Let t be an upper bound on the number of instances overall. We assume[2] some fixed set $\{\mathsf{id}^i = (I^i, q^i)\}_{i=1}^t$ of identifiers, where $\mathsf{id}^i \neq \mathsf{id}^j$ for $i \neq j$.

We are interested in bounding the attacker's success probability in the following experiment. (Since we prove security when the hash function H is modeled as a random oracle, we explicitly incorporate random choice of H into the experiment.)

1. A random function $H : \{0,1\}^* \to \{0,1\}^{8n}$ is chosen.

2. For $i = 1, \ldots, t$, the key-generation algorithm is run using identifier id^i to obtain (pk^i, sk^i). The attacker is given $(\mathsf{id}^1, pk^1), \ldots, (\mathsf{id}^t, pk^t)$.

3. The attacker is given oracle access to H, plus a signing oracle $\mathsf{Sign}(\cdot,\cdot)$ such that $\mathsf{Sign}(i, M)$ returns $\mathsf{Sign}(sk^i, M, \mathsf{id}^i)$. For each i, the attacker may make at most one query $\mathsf{Sign}(i, \star)$.

   Without loss of generality we assume the attacker makes exactly one signing query $\mathsf{Sign}(i, M^i)$ for each value of i. We also assume that when the attacker is given a signature, it is additionally given the answers to all the H-queries needed to verify that signature.

4. The attacker outputs (i, M, σ) with $M \neq M^i$. The attacker succeeds if σ is a valid signature on M in the ith instance of the scheme, i.e., if $\mathsf{Vrfy}(M, \sigma, I^i) = pk^i$. Without loss of generality we assume the attacker has previously made (or has been given the answers to) all the H-queries needed to run the verification algorithm on these inputs.

Instantiating the security experiment above with the algorithms of the LM-OTS scheme, and performing some syntactic changes that do not change the probability space, we obtain the following experiment (we use $\|$ for string concatenation when using commas would cause confusion):

1. Initialize an empty set H. (H will contain defined query/answer pairs for the function H. That is, if $(x, y) \in H$ then $H(x) = y$.)

2. For $i = 1, \ldots, t$, do:

   (a) For $j = 0, \ldots, p-1$, choose uniform $x^i_{j,0} \in \{0,1\}^{8n}$.

   (b) For $j = 0, \ldots, p-1$ and $k = 0, \ldots, e-1$, choose uniform value $x^i_{j,k+1} \in \{0,1\}^{8n}$ and add $\left(x^i_{j,k} \,\|\, \mathsf{id}^i \,\|\, [j]_2 \,\|\, [k]_1 \,\|\, \mathtt{0x00},\ x^i_{j,k+1}\right)$ to H. Define $y^i_j := x^i_{j,e}$.

   (c) Choose uniform $pk^i \in \{0,1\}^{8n}$. Add $\left(\mathsf{id}^i \| y^i_0 \| \cdots \| y^i_{p-1} \| \mathtt{0x01},\ pk^i\right)$ to H.

   (d) Choose uniform $C^i \in \{0,1\}^{8n}$ and $Q^i \in \{0,1\}^{8n}$.

   (e) Give (id^i, pk^i) to the attacker.

3. When the attacker makes a query $H(x)$, answer it as follows:

   (a) If there is an entry $(x, y) \in H$ for some y, then return y.

[2] These identifiers could be chosen adaptively by the attacker (subject to being distinct) without any significant change to the proof in the following section, but for simplicity we treat them as fixed in advance. When LM-OTS is used in the LMS scheme, the identifiers can be viewed as being fixed in advance.

(b) Otherwise, choose uniform $y \in \{0,1\}^{8n}$, return y to the attacker, and store (x, y) in H.

4. When the attacker makes a query $\mathsf{Sign}(i, M^i)$, answer it as follows:

   (a) If there is an entry $(M^i \| C^i \| \mathsf{id}^i \| \mathtt{0x02},\ Q) \in H$ for some Q, then redefine $Q^i := Q$. Store $(M^i \| C^i \| \mathsf{id}^i \| \mathtt{0x02},\ Q^i)$ in H.

   (b) Let $c^i := \mathsf{checksum}(Q^i)$, and set $V^i := Q^i \| c^i$. Parse V^i as a sequence of w-bit integers $V_0^i, \ldots, V_{p-1}^i$.

   (c) Return the signature $(C^i, x^i_{0,V_0^i}, \ldots, x^i_{p-1,V^i_{p-1}})$.

5. The attacker outputs (i, M, σ) with $M \neq M^i$. The attacker succeeds if $\mathsf{Vrfy}(M, \sigma, I^i) = pk^i$.

We define the following events in the above experiment:

- $\mathsf{Coll}_{1,i}$ is the event that the attacker queries $H(I^i, q, y_0, \ldots, y_{p-1}, \mathtt{0x01})$ with $(q, y_0, \ldots, y_{p-1}) \neq (q^i, y_0^i, \ldots, y_{p-1}^i)$, and receives the response pk^i.
- $\mathsf{Coll}_{2,i}$ is the event the attacker queries $H(\star, C^i, \mathsf{id}^i, \mathtt{0x02})$ before making the query $\mathsf{Sign}(i, \star)$.
- $\mathsf{Coll}^*_{2,i}$ is the event that either $\mathsf{Coll}_{2,i}$ occurs, or either of the following occur: (1) before making the query $\mathsf{Sign}(i, \star)$, the attacker queries $H(\star, \star, \mathsf{id}^i, \mathtt{0x02})$ and receives the response Q^i, or (2) after making the query $\mathsf{Sign}(i, M^i)$, the attacker queries $H(M, \star, \mathsf{id}^i, \mathtt{0x02})$ with $M \neq M^i$, and receives the response Q^i.
- $\mathsf{Coll}_{3,i,j,k}$ is the event that the attacker queries $H(x^i_{j,k}, \mathsf{id}^i, [j]_2, [k]_1, \mathtt{0x00})$ either before making the query $\mathsf{Sign}(i, \star)$, or after making the query $\mathsf{Sign}(i, \star)$ but with $k < V_j^i$.
- $\mathsf{Coll}^*_{i,j,k}$ is the event that either $\mathsf{Coll}_{i,j,k}$ occurs, or the attacker queries $H(x, \mathsf{id}^i, [j]_2, [k]_1, \mathtt{0x00})$ with $x \neq x^i_{j,k}$, and receives the response $x^i_{j,k+1}$.

We first observe that the probability of forgery can be upper-bounded by the probability that one of the above events occurs.

Claim 1. *If the attacker succeeds, then either* $\mathsf{Coll}_{1,i}$ *or* $\mathsf{Coll}^*_{2,i}$ *occur for some* $i \in \{1, \ldots, t\}$*, or else* $\mathsf{Coll}^*_{i,j,k}$ *occurs for some* $i \in \{1, \ldots, t\}, j \in \{0, \ldots, p-1\}$*, and* $k \in \{0, \ldots, e-1\}$*.*

Proof. Say the attacker outputs (i, M, σ) with $M \neq M^i$ and σ a valid signature on M with respect to I^i, pk^i. By assumption, all the H-queries needed to verify σ on M with respect to I^i, pk^i are defined when the attacker outputs (i, M, σ). Parse σ as $(C, q, \sigma_0, \ldots, \sigma_{p-1})$ and set $\mathsf{id} = (I^i, q)$. Define $Q = H(M, C, \mathsf{id}, \mathtt{0x02})$ and $c = \mathsf{checksum}(Q)$, and let $V_0, \ldots, V_{p-1} = Q \| c$ and $y_j = H^{e-V_j}_{\mathsf{id},[j]_2}(\sigma_j; V_j)$ be the values computed by running the verification algorithm with respect to I^i, pk^i on the message M and signature σ. Since the attacker succeeds, $H(\mathsf{id}, y_0, \ldots, y_{p-1}, \mathtt{0x01}) = pk^i$.

We show that if $\mathsf{Coll}_{1,i}$ and $\mathsf{Coll}^*_{2,i}$ have not occurred, then $\mathsf{Coll}^*_{i,j,k}$ must have occurred for some j, k. If $\mathsf{Coll}_{1,i}$ has not occurred, we must have $(q, y_0, \ldots, y_{p-1}) = (q^i, y_0^i, \ldots, y_{p-1}^i)$ and so $\mathsf{id} = \mathsf{id}^i$. If $\mathsf{Coll}^*_{2,i}$ (and hence $\mathsf{Coll}_{2,i}$) has not occurred, the value of Q^i was not changed during the experiment, and also $Q \neq Q^i$. By construction of $\mathsf{checksum}$, we must therefore have $V_j < V_j^i$ for some j. But then one can verify by inspection that $\mathsf{Coll}^*_{3,i,j,k}$ must have occurred for some k. $\square$

Thus, to bound the success probability of the attacker it suffices to bound the probabilities of the above events.

Claim 2. *For all i,* $\Pr[\mathsf{Coll}_{1,i}] \leq q_{1,i} \cdot 2^{-8n}$, *where $q_{1,i}$ is the number of H-queries of the form* $H(I^i, \star, \star, \ldots, \star, \texttt{0x01})$.

Proof. Any query $H(I^i, q, y_0, \ldots, y_{p-1}, \texttt{0x01})$ with $(q, y_0, \ldots, y_{p-1}) \neq (q^i, y_0^i, \ldots, y_{p-1}^i)$ returns a uniform value in $\{0,1\}^{8n}$ that is independent of pk^i. The claim follows. □

Claim 3. *For all i,* $\Pr[\mathsf{Coll}_{2,i}] \leq q_{2,i} \cdot 2^{-8n}$, *where $q_{2,i}$ is the number of H-queries of the form* $H(\star, \star, \mathsf{id}^i, \texttt{0x02})$.

Proof. C^i is a uniform $8n$-bit string, and the attacker has no information about C^i until it queries $\mathsf{Sign}(i, \star)$. The claim follows. □

Claim 4. *For all i,* $\Pr[\mathsf{Coll}^*_{2,i}] \leq 2q_{2,i} \cdot 2^{-8n}$, *where $q_{2,i}$ is as in the previous claim.*

Proof. We have $\Pr[\mathsf{Coll}^*_{2,i}] \leq \Pr[\mathsf{Coll}_{2,i}] + \Pr[\mathsf{Coll}^*_{2,i} \mid \neg\mathsf{Coll}_{2,i}]$. The previous claim provides an upper bound on the first term. As for the second term, when $\mathsf{Coll}_{2,i}$ does not occur, the value of Q^i does not change during the experiment. Each time the attacker queries $H(\star, \star, \mathsf{id}^i, \texttt{0x02})$ before making the query $\mathsf{Sign}(i, \star)$, or queries $H(M, \star, \mathsf{id}^i, \texttt{0x02})$ with $M \neq M^i$ after the query $\mathsf{Sign}(i, M^i)$, the value returned is uniform in $\{0,1\}^{8n}$ and independent of Q^i. The claim follows. □

Claim 5. *For all i, j, k,*

$$\Pr\left[\mathsf{Coll}_{3,i,j,k} \mid \bigwedge_{\ell=0}^{k-1} \neg\mathsf{Coll}^*_{3,i,j,\ell}\right] \leq q_{3,i,j,k} \cdot 2^{-8n},$$

where $q_{3,i,j,k}$ is the number of H-queries of the form $H(\star, \mathsf{id}^i, [j]_2, [k]_1, \texttt{0x00})$.

Proof. When $\mathsf{Coll}^*_{3,i,j,k-1}$ does not occur, the attacker gets no information about $x^i_{j,k}$ until it queries $H(x^i_{j,k}, \mathsf{id}^i, [j]_2, [k]_1, \texttt{0x00})$ or $\mathsf{Sign}(i, M^i)$ with $V^i_j \leq k$. In the latter case $\mathsf{Coll}_{3,i,j,k}$ cannot occur once the signature query is made. Since $x^i_{j,k}$ is uniform in $\{0,1\}^{8n}$, the claim follows. □

Claim 6. *For all i, j, k,*

$$\Pr\left[\mathsf{Coll}^*_{3,i,j,k} \mid \bigwedge_{\ell=0}^{k-1} \neg\mathsf{Coll}^*_{3,i,j,\ell}\right] \leq 2q_{3,i,j,k} \cdot 2^{-8n},$$

where $q_{3,i,j,k}$ is as in the previous claim.

Proof. We have

$$\begin{aligned}
&\Pr\left[\mathsf{Coll}^*_{3,i,j,k} \mid \bigwedge_{\ell=0}^{k-1} \neg\mathsf{Coll}^*_{3,i,j,\ell}\right] \\
&\leq \Pr\left[\mathsf{Coll}_{3,i,j,k} \mid \bigwedge_{\ell=0}^{k-1} \neg\mathsf{Coll}^*_{3,i,j,\ell}\right] \\
&\quad + \Pr\left[\mathsf{Coll}^*_{3,i,j,k} \mid \bigwedge_{\ell=0}^{k-1} \neg\mathsf{Coll}^*_{3,i,j,\ell} \bigwedge \neg\mathsf{Coll}_{3,i,j,k}\right].
\end{aligned}$$

The previous claim provides an upper bound on the first term. As for the second term, note that when $\mathsf{Coll}_{3,i,j,k}$ does not occur then whenever the attacker queries $H(\star, \mathsf{id}^i, [j]_2, [k]_1, \texttt{0x00})$, the value returned is uniform in $\{0,1\}^{8n}$ and independent of $x^i_{j,k+1}$. The claim follows. □

Claim 7. *For all i, j, $\Pr\left[\bigvee_{k=0}^{e-1} \mathsf{Coll}^*_{3,i,j,k}\right] \leq 2 \cdot \sum_{k=0}^{e-1} q_{3,i,j,k} \cdot 2^{-8n}$, where $q_{3,i,j,k}$ is as in the previous claim.*

Proof. We have

$$\Pr\left[\bigvee_{k=0}^{e-1} \mathsf{Coll}^*_{3,i,j,k}\right] \leq \sum_{k=0}^{e-1} \Pr\left[\mathsf{Coll}^*_{3,i,j,k} \mid \bigwedge_{\ell=0}^{k-1} \neg\mathsf{Coll}^*_{3,i,j,\ell}\right] \leq \sum_{k=0}^{e-1} 2q_{3,i,j,k} \cdot 2^{-8n},$$

using the previous claim. □

Putting everything together, we have:

Theorem 8. *For any adversary attacking arbitrarily many instances of the LM-OTS scheme, and making at most q hash queries of the form $H(\star, n)$ with $n \in \{\texttt{0x00}, \texttt{0x01}, \texttt{0x02}\}$, the probability with which the adversary forges a signature with respect to any of the instances is at most $2q \cdot 2^{-8n}$.*

Proof. Let t denote the number of instances of the scheme. Using Claim 1 and a union bound, the probability with the the adversary forges a signature is at most

$$\sum_{i=1}^{t} \Pr[\mathsf{Coll}_{1,i}] + \sum_{i=1}^{t} \Pr[\mathsf{Coll}^*_{2,i}] + \sum_{i=1}^{t} \sum_{j=0}^{p-1} \Pr\left[\bigvee_{k=1}^{e-1} \mathsf{Coll}^*_{3,i,j,k}\right].$$

Using Claims 2, 4, and 7, the above is at most

$$\begin{aligned}
&\sum_{i=1}^{t} q_{1,i} \cdot 2^{-8n} + 2 \cdot \sum_{i=1}^{t} q_{2,i} \cdot 2^{-8n} + 2 \cdot \sum_{i=1}^{t} \sum_{j=0}^{p-1} \sum_{k=0}^{e-1} q_{3,i,j,k} \cdot 2^{-8n} \\
&\qquad \leq 2 \cdot \left(\sum_{i=1}^{t} q_{1,i} + \sum_{i=1}^{t} q_{2,i} + \sum_{i=1}^{t} \sum_{j=0}^{p-1} \sum_{k=0}^{e-1} q_{3,i,j,k} \right) \cdot 2^{-8n}.
\end{aligned}$$

Each of the adversary's H-queries of the stated form increases the value of at most one of $q_{1,i}$, $q_{2,i}$, or $q_{3,i,j,k}$ and so the sum in the parentheses is at most q. This proves the theorem. □

4 Description of the LMS Scheme

An instance of the LMS scheme is defined by computing a Merkle tree of height h using 2^h LM-OTS public keys at the leaves. We give a formal definition now.

Key-generation algorithm Gen'

Key generation takes as input a 31-byte *identifier* I and a parameter h. Set $N = 2^h - 1$. The algorithm proceeds as follows:

1. For $q = 0, \ldots, N$, compute $(pk^q, sk^q) \leftarrow \mathsf{Gen}(I, q)$.
2. For $r = 2^h, \ldots, 2^{h+1} - 1$, set $T[r] := H(pk^{r-2^h}, I, r, \texttt{0x03})$.
3. For $r = 2^h - 1, \ldots, 1$, set $T[r] := H(T[2r], T[2r+1], I, r, \texttt{0x04})$.

The public key is $pk = (h, I, T[1])$, and the private key is $sk = (0, sk^0, \ldots, sk^N)$.

Signing algorithm Sign'

Signing takes as input a private key $(q, sk^0, \ldots, sk^N)$ and a message $M \in \{0,1\}^*$ as usual, as well as I as above. It sets $\mathsf{id} = (I, q)$ and does:

1. Compute $\sigma := \mathsf{Sign}(sk^q, M, \mathsf{id})$.
2. Also compute $p_0, \ldots, p_{h-1}$, the siblings of the nodes on the path from leaf q to the root in the Merkle tree.
3. Return the signature $\Sigma = (\sigma, p_0, \ldots, p_{h-1})$.

After generating a signature, the value of q is incremented. (Signing is stateful.) If $q = 2^h$ the key is erased, and no more signatures can be issued.

Verification algorithm Vrfy'

Verification takes as input a public key (h, I, T), a message $M \in \{0,1\}^*$, and a signature $\Sigma = (\sigma, p_0, \ldots, p_{h-1})$. It does:

1. Compute $pk := \mathsf{Vrfy}(M, \sigma)$.
2. Extract value q from σ. Compute $T[q + 2^h] := H(pk, I, q + 2^h, \mathtt{0x03})$.
3. Using $p_0, \ldots, p_{h-1}$, compute a value $T[1]$. Return 1 if and only if $T[1] = T$.

5 Security of the LMS Scheme

Security of the LMS scheme can be proven generically based on any one-time signature scheme and any second preimage-resistant hash function. However, since the hash function H was modeled as a random oracle in our analysis of the LM-OTS scheme, we continue to model it as a random oracle here. Note also that although the same function H is used both to compute the Merkle tree and in the underlying one-time signature scheme, the fact that domain separation is used means that we can cleanly separate these two usages.

Here, we are interested in the attacker's success probability in the following experiment:

1. A random function $H : \{0,1\}^* \to \{0,1\}^{8n}$ is chosen.
2. The key-generation algorithm for the LMS scheme is run using I and h to obtain (pk, sk). The attacker is given pk.
3. The attacker is given oracle access to H, plus a stateful signing oracle $\mathsf{Sign}'(\cdot)$ such that $\mathsf{Sign}'(M)$ returns $\mathsf{Sign}'(sk, M, I)$ and updates the private key.

   We assume that when the attacker is given a signature, it is additionally given the answers to all the H-queries needed to verify that signature.
4. The attacker outputs (M, Σ), where M was not previously submitted to its signing oracle. The attacker succeeds if Σ is a valid signature on M, i.e., if $\mathsf{Vrfy}'(pk, M, \Sigma) = 1$. Without loss of generality we assume the attacker has previously made (or has been given the answers to) all the H-queries needed to run the verification algorithm on these inputs.

We remark that we consider the single-instance setting for simplicity; one can verify that security does not degrade in the multi-instance setting as long as each instance uses a distinct I value.

Considering an execution of the above experiment, let $pk^0, \ldots, pk^{2^h-1}$ be the LM-OTS public keys at the leaves, and let $T[r]$ denote the intermediate values computed during the course of key generation. Denote the components of the signature output by the attacker by $\Sigma = (\sigma, p_0, \ldots, p_{h-1})$. (We may assume Σ has this form, since otherwise the signature will surely be invalid. In particular, we may assume without loss of generality that Σ consists of a value σ in the format of an LMS-OTS signature and h values $p_0, \ldots, p_{h-1}$.) Let q be the value contained in σ, and let pk be the value computed during verification of Σ on M. Let Forge_1 be the event that that attacker succeeds and $pk = pk^q$, and let Forge_2 be the event that the attacker succeeds but $pk \neq pk^q$.

We have

Claim 9. $\Pr[\mathsf{Forge}_1] \leq 2q_1 \cdot 2^{-8n}$*, where* q_1 *is the number of* H*-queries of the form* $H(\star, n)$ *with* $n \in \{\mathtt{0x00}, \mathtt{0x01}, \mathtt{0x02}\}$*.*

Proof. Let $\mathcal{A}$ be an adversary attacking the LMS scheme; we construct an attacker $\mathcal{A}'$ attacking the LM-OTS scheme.

Fix some I, h, and let $\mathsf{id}^q = (I, q)$ for $q = 0, \ldots, 2^h - 1$. Attacker $\mathcal{A}'$ is given public keys $pk^0, \ldots, pk^{2^h-1}$ and does as follows:

1. Compute $T[1]$ from $pk^0, \ldots, pk^{2^h-1}$ as in algorithm Gen'. Give public key $pk = (h, I, T[1])$ to $\mathcal{A}$.
2. When $\mathcal{A}$ requests the ith signature on some message M (for $i = 0, \ldots, 2^h - 1$), attacker $\mathcal{A}'$ queries $\mathsf{Sign}(i, M)$ to obtain σ. It then computes $p_0, \ldots, p_{h-1}$ as in algorithm Sign', and returns the signature $(\sigma, p_0, \ldots, p_{h-1})$ to $\mathcal{A}$.
3. $\mathcal{A}'$ answers H-queries of $\mathcal{A}$ by forwarding them to its own H-oracle.
4. When $\mathcal{A}$ outputs a forgery $(M, \Sigma = (\sigma, p_0, \ldots, p_{h-1}))$, adversary $\mathcal{A}'$ extracts the value q contained in σ and outputs (q, M, σ).

Observe that $\mathcal{A}'$ succeeds if Forge_1 occurs. Moreover, although $\mathcal{A}'$ may make H-queries in addition to those made by $\mathcal{A}$ (to compute $T[1]$), all those queries are of the form $H(\star, n)$ with $n \in \{\mathtt{0x03}, \mathtt{0x04}\}$; the number of H-queries of the form $H(\star, n)$ with $n \in \{\mathtt{0x00}, \mathtt{0x01}, \mathtt{0x02}\}$ is exactly the same as the number made by $\mathcal{A}$. Theorem 8 thus implies the claim. □

We turn to bounding Forge_2. For some fixed I, h, define the following events:

- Coll_r, for $r = 2^h, \ldots, 2^{h+1} - 1$, is the event that the attacker queries $H(pk, I, r, \mathtt{0x03})$ with $pk \neq pk^{r-2^h}$ and receives the response $T[r]$.
- Coll_r, for $r = 1, \ldots, 2^h - 1$, is the event that the attacker queries $H(T, T', I, r, \mathtt{0x04})$ with $(T, T') \neq (T[2r], T[2r+1])$ and receives the response $T[r]$.

Claim 10. $\Pr[\mathsf{Forge}_2] \leq q' \cdot 2^{-8n}$*, where* q_2 *is the number of* H*-queries of the form* $H(\star, n)$ *with* $n \in \{\mathtt{0x03}, \mathtt{0x04}\}$*.*

Proof. If Forge_2 occurs then Coll_r occurs for some r. It is also easy to see that $\Pr[\mathsf{Coll}_r] \leq q_r \cdot 2^{-8n}$, where q_r is the number of H-queries of the form $H(\star, I, r, \star)$. Since each of the adversary's queries of the stated form increases the value of at most one q_r, the claim follows. □

Theorem 11. *For any adversary attacking the LMS scheme and making at most q hash queries, the probability with which the adversary can forge a signature is at most $2q \cdot 2^{-8n}$.*

Proof. The probability that the attacker forges a signature is $\Pr[\mathsf{Forge}_1] + \Pr[\mathsf{Forge}_2]$. By Claims 9 and 10, this is bounded by $2q_1 \cdot 2^{-8n} + q_2 \cdot 2^{-8n}$, where q_1, q_2 are as in those claims. Since each H-query by the attacker increases the value of at most one of q_1 or q_2, the claimed bound follows. $\square$

References

[1] J. Buchmann, E. Dahmen, and M. Szydlo. Hash-based digital signature schemes. Technical Report, Technische Universitat Darmstadt, 2008.

[2] J. Katz. Analysis of a Proposed Hash-Based Signature Standard. Contribution to IRTF, 2015. Available at `http://www.cs.umd.edu/~jkatz/papers/HashBasedSigs.pdf`

[3] J. Katz and Y. Lindell. *Introduction to Modern Cryptography, 2nd edition.* Chapman & Hall/CRC Press, 2014.

[4] L. Lamport. Constructing digital signatures from a one-way function. Tehcnical Report SRI-CSL-98, SRI Intl. Computer Science Laboratory, 1979.

[5] F.T. Leighton and S. Micali. Large provably fast and secure digital signature schemes based on secure hash functions. US Patent 5,432,852, July 11, 1995.

[6] D. McGrew and M. Curcio. Hash-based signatures. Internet Draft draft-mcgrew-hash-sigs-04, March 21, 2016.

[7] R.C. Merkle. Secrecy, authentication, and public-key systems. PhD Thesis, Stanford University, 1979.

[8] R.C. Merkle. A certified digital signature. *Advances in Cryptology—Crypto '89*, LNCS vol. 435, pages 218–238, Springer-Verlag, 1989.