Append-Only Signatures[⋆]

Eike Kiltz[⋆⋆], Anton Mityagin[⋆⋆⋆], Saurabh Panjwani[†], and Barath Raghavan[‡]

Department of Computer Science and Engineering,
University of California, San Diego, USA
{ekiltz, amityagin, panjwani, barath}@cs.ucsd.edu

Abstract. We present a new primitive – Append-only Signatures (AOS) – with the property that any party given an AOS signature $\texttt{Sig}[M_1]$ on message M_1 can compute $\texttt{Sig}[M_1 \| M_2]$ for any message M_2, where $M_1 \| M_2$ is the concatenation of M_1 and M_2. We define the security of AOS, present concrete AOS schemes, and prove their security under standard assumptions. In addition, we find that despite its simple definition, AOS is equivalent to Hierarchical Identity-based Signatures (HIBS) through efficient and security-preserving reductions. Finally, we show direct applications of AOS to problems in network security. Our investigations indicate that AOS is both useful in practical applications and worthy of further study as a cryptographic primitive.

Keywords: Algebraic Signatures, Append-only Signatures, Hierarchical Identity-based Signatures.

1 Introduction

In many real-world applications, users and programs alike require notions of delegation to model the flow of information. It is often required that delegation from one party to another enables the delegatee to "append" to the information it received but to do nothing more. For example, in wide-area Internet routing, each network passes a routing path advertisement to its neighboring networks, which then append to it information about themselves and forward the updated advertisement to their neighbors. For security, the route advertisements must be authenticated; intermediate networks must be incapable of modifying routes except according to the protocol (that is, by appending their names to already-received advertisements). Likewise, in the context of secure resource delegation for distributed systems, users need to delegate their share of resources to other users, who may then re-delegate to other users by including their own resources in

[⋆] Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.
[⋆⋆] Supported in by a DAAD postdoc fellowship.
[⋆⋆⋆] Supported in part by NSF grants ANR-0129617 and CCR-0208842.
[†] Supported in part by NSF grant 0313241.
[‡] Supported by a NSF Graduate Research Fellowship.

the pool. In many of these applications, it is desirable that delegation is possible without parties having to share any cryptographic keys and that the authenticity of any information received through a series of delegations is verifiable based only on the identity of the first party in the chain.

To directly address these needs, we present a new cryptographic primitive called Append-Only Signatures (AOS). An AOS scheme enables the extension of signed messages and update of the corresponding signatures, without requiring possession of the signing key. That is, any party given an AOS signature $\texttt{Sig}[M_1]$ on message M_1 can compute $\texttt{Sig}[M_1 \| M_2]$ for any message M_2, where $M_1 \| M_2$ is the concatenation of M_1 and M_2. The verifier of the final signature needs the initial signer's public key but does not need to know the public keys or any other information from intermediate signers except the message data appended. Clearly, such a scheme cannot be secure according to the standard notion of security for signatures. Instead, we define an AOS scheme to be secure if it is infeasible to forge signatures of messages that are not obtained by extending already-signed messages. A formal definition appears in Section 2.

In Section 3 we present several provably secure AOS schemes, offering different tradeoffs of flexibility and efficiency. Our first construction shows a generic approach to building AOS schemes from any standard digital signature scheme using certificate chains. The construction works as follows: The secret and public keys for the AOS scheme are obtained by running the key generator for $\mathcal{SIG}$. For any message $M = M_1 \| M_2 \| \cdots \| M_n$, each M_i being a symbol in some predetermined message space, the AOS signature of M is defined as a sequence of n public keys $pk_1, pk_2, \cdots, pk_n$ (generated using the key generator for $\mathcal{SIG}$) and a sequence of n certificates binding the message symbols to these public keys. The ith certificate in the chain binds the message symbol M_i to the corresponding public key pk_i and is signed using the secret key, sk_{i-1}, corresponding to pk_{i-1}. The secret key, sk_0, of the AOS scheme signs the first certificate in the chain while the secret key sk_n (corresponding to the last public key), is revealed as part of the AOS signature and is used for appending new symbols to M. We observe that if the message space is small enough, we can make use of "weaker", and more efficient, signature schemes without compromising the security of the resulting AOS scheme. Using aggregation techniques of [2, 10] one can reduce the length of the signature by a factor of two.

We also present a more efficient construction of AOS for applications in which the message space is constant size and the total number of append operations performed is also constant. This construction is based on a seemingly stronger assumption (than that of the existence of signature schemes) and makes use of pseudorandom generators and collision-resistant hash functions (CRHFs).

Relation to Hierarchical Identity-Based Signatures. Identity-Based Signature (IBS) schemes, due to Shamir [14], are signature schemes in which the identity of the signer (for example, her email address) plays the role of his public key. Such schemes assume the existence of a trusted authority that holds a master public-private key pair that is used to assign secret keys to users based on their identities. Anyone can verify signatures on messages signed by a user knowing only the master public key and the identity of that user. Hierarchical IBS (HIBS) schemes, proposed by Gentry and Silverberg [4], are identity-based signature schemes in which users are arranged in a hierarchy and a user at any level in this hierarchy can delegate secret keys to her descendants based on their identities and her own secret key. To verify the signature created by any user, one needs to know only the identity of the user (and her position in the hierarchy) and the public key of the root user.

HIBS can be implemented using certificate chains (as suggested in [4]) and the resulting construction bears a strong resemblance to the certificate-based construction of AOS we give in this paper. Upon closer examination, we find that the similarity between the two constructions is not accidental: it is an artifact of the close relationship between the two primitives themselves—AOS and HIBS are, in fact, tightly equivalent. This means that (a) there exist generic transformations from any HIBS scheme into a corresponding AOS scheme and, likewise, from any AOS scheme into a corresponding HIBS scheme; and (b) these transformations are extremely efficient (the derived scheme is as efficient as the scheme being derived from) and highly security-preserving (an adversary attacking the derived scheme can be transformed into an adversary attacking the original one, losing only a constant factor in efficiency and query complexity). Section 4 gives details.

A benefit of this equivalence is that it considerably simplifies the notion of HIBS and makes security analysis for HIBS schemes less onerous: AOS is simpler than HIBS, and, for any HIBS scheme, it is typically easy to find an equivalent AOS scheme whose security properties carry over to the corresponding HIBS scheme. For example, our security proof for certificate-based AOS translates to a security proof for certificate-based HIBS (originally proposed in [4]). Although this construction of HIBS was known prior to our work, it was never analyzed in the literature, and, to the best of our knowledge, we give the first proof of security for it. Furthermore, our construction of AOS based on pseudorandom generators and CRHFs yields a novel approach to designing HIBS and can be useful for some restricted scenarios (for example, in a constant-depth hierarchy wherein each user signs messages from a constant-size message space). We remark that both these constructions yield HIBS schemes in the standard model and neither involves the use of computationally intensive bilinear maps (this is in contrast with some recent results on HIBS [3]).

Application to Secure Routing. In Section 5 we discuss an important real-life application of AOS in internet routing security.

Related Work. Append-only signatures belong to a general class of primitives called *algebraic signatures*. Informally, an algebraic signature scheme allows the creation of signatures on a message M using the signatures on some known messages, $M_1, M_2, \ldots, M_n$, and the public key, provided M can be obtained from the known messages using some prespecified set of (n-ary) operations, say $\mathcal{O} = \{f_1, f_2, \cdots, f_m\}$. That is, given the signatures, $\mathtt{sig}[M_1], \ldots, \mathtt{sig}[M_n]$ and the public key, it is easy to compute $\texttt{sig}[f_i(M_1, \ldots, M_n)]$ for any $f_i \in \mathcal{O}$. In our setting, each f_i has arity 1 and appends some fixed message symbol M_i to an input message M. Security for algebraic signatures is defined in a manner similar to our approach to security of AOS (that is, it should be hard to forge signatures of messages that cannot be obtained by applying the operations in $\mathcal{O}$ to already-signed messages). Examples of algebraic signatures studied in the literature include transitive signatures by Micali and Rivest [12], homomorphic signatures by Johnson, Molnar, Song and Wagner [7], and graph-based algebraic signatures by Hevia and Micciancio [5].

Although no obvious relation exists between our primitive and any of the previously studied algebraic signature primitives, we do note that some of the techniques we use in our constructions parallel prior techniques. For example, our construction of AOS schemes using CRHFs can be viewed as a special instance of graph-based algebraic signature schemes studied in [5] (although the set of update operations considered there are different from the `append` operation that we consider).

2 Append-Only Signatures

Informally, append-only signatures (AOS) are signatures that enable the public extension of existing signatures. That is, any party given an AOS signature `Sig` on a message $(M_1, \ldots, M_n)$ can compute an AOS signature on any message $(M_1, \ldots, M_n, M_{n+1})$. (As in the introduction, one could represent the message $(M_1, \ldots, M_n)$ as the string $M_1 || \ldots || M_n$ which better captures the idea of appending. However, since we want to differentiate between the message "A" || "B" and the message symbol "AB", we prefer to think of messages as n-tuples. That is, in our example, we have the two different tuples (A, B) and (AB)). Besides the append operation, AOS is the same as ordinary signatures. That is, given only an AOS signature on the message $(M_1, \ldots, M_n)$ it should be infeasible to forge an AOS signature on any message not having $(M_1, \ldots, M_n)$ as a prefix.

Let $\mathsf{AOS.MSpace}$ be any set of symbols (for example, $\{0, 1\}$ or $\{0, 1\}^*$). A *message of length n* is an n tuple of symbols written as $M[1..n] = (M_1, M_2, \ldots, M_n)$. The special case of $n = 0$ is the empty message, denoted ε. We use the symbol $\sqsubseteq$ to denote the prefix relation over the messages. An append-only signature (AOS) scheme with respect to the message space $\mathsf{AOS.MSpace}$ is a collection of three polynomial-time algorithms: a setup algorithm ($\mathsf{AOS.Setup}$), an append algorithm ($\mathsf{AOS.Append}$), and a verify algorithm ($\mathsf{AOS.Vfy}$), defined as follows:

- $\mathsf{AOS.Setup}$ takes the security parameter as input and outputs a pair of keys: the public key $\mathsf{AOS.pk}$ and the secret key $\mathsf{Sig}[\varepsilon]$, which is the signature on the empty message ε.
- $\mathsf{AOS.Append}$ takes the public key $\mathsf{AOS.pk}$, a signature on a message $M[1..n-1] = (M_1, \ldots M_{n-1})$, of length $n-1$, and a symbol $M_n \in \mathsf{AOS.MSpace}$ and produces a signature on the message $M[1..n] = (M_1, \ldots, M_n)$.
- $\mathsf{AOS.Vfy}$ takes the public key $\mathsf{AOS.pk}$, a message $M[1..n]$, and a signature `sig`, and returns either `true` or `false`.

All algorithms can be randomized. Additionally, the scheme should have the property that for any pair $(\mathsf{AOS.pk}, \mathsf{Sig}[\varepsilon])$ and any message $M[1..n]$, the signature $\mathtt{sig}$ obtained by iteratively appending $M_1, \ldots, M_n$ to $\mathsf{Sig}[\varepsilon]$ should be accepted by $\mathsf{AOS.Vfy}$. Appendig the symbols one-by-one should be the only way of generating a signatures on the message $M[1..n]$. This fact ensures history independence of AOS: that is, no party, given an AOS signature, can tell whether the signature was created by the owner of the secret key or whether it passed through multiple parties that appended symbols at every step. History independence is a useful property to have in most applications, as already highlighted in previous work on algebraic signatures [7].

Definition 1. Let $\mathcal{AOS} = (\mathsf{AOS.Setup}, \mathsf{AOS.Append}, \mathsf{AOS.Vfy})$ be an AOS scheme, let k be the security parameter, and let $\mathcal{A}$ be an adversary. We consider the experiment:

```
Experiment $\mathbf{Exp}^{\text{aos-uf-cma}}_{\mathcal{AOS},\mathcal{A}}(k)$
    $MSGSet \leftarrow \emptyset$ ; $(\mathsf{AOS.pk}, \mathsf{Sig}[\varepsilon]) \xleftarrow{\$} \mathsf{AOS.Setup}(1^k)$
    $(M[1..n], \mathtt{sig}) \xleftarrow{\$} \mathcal{A}^{\text{AOSSign}(\cdot)}(\mathsf{AOS.pk})$
    if $\mathsf{AOS.Vfy}(\mathsf{AOS.pk}, M[1..n], \mathtt{sig}) = \mathtt{true}$
        and $\forall J[1..j] \sqsubseteq M[1..n] : J[1..j] \notin MSGSet$
        then return 1 else return 0

Oracle AOSSign$(M[1..n])$
    Add $M[1..n]$ to $MSGSet$
    return Extract$(M[1..n])$

Oracle Extract$(M[1..i])$ // defined recursively
  if $i = 0$ then return $\mathsf{Sig}[\varepsilon]$
  else if $\mathsf{Sig}[M[1..i]] =$ defined
    then return $\mathsf{Sig}[M[1..i]]$
    else $\mathsf{Sig}[M[1..i]] \xleftarrow{\$} \mathsf{AOS.Append}(\mathsf{AOS.pk}, M[1..i-1], \text{Extract}(M[1..i-1]), M_i)$
        return $\mathsf{Sig}[M[1..i]]$
```

The aos-uf-cma-advantage of an adversary $\mathcal{A}$ in breaking the security of the scheme $\mathcal{AOS}$ is defined as $\mathbf{Adv}^{\text{aos-uf-cma}}_{\mathcal{AOS},\mathcal{A}}(k) = \Pr[\,\mathbf{Exp}^{\text{aos-uf-cma}}_{\mathcal{AOS},\mathcal{A}}(k) = 1\,]$, and $\mathcal{AOS}$ is said to be *unforgeable under chosen message attacks* (aos-uf-cma secure) if the above advantage is a negligible function in k for all polynomial-time adversaries $\mathcal{A}$.

Note that $\mathcal{A}$ is given access to the oracle AOSSign$(\cdot)$, not to the oracle Extract$(\cdot)$ (the latter is used internally by AOSSign$(\cdot)$ to create intermediate signatures).

3 Efficient AOS Constructions

We briefly sketch our constructions for AOS in this section. More details (including proofs of all theorems) can be found in [9].

Certificate-Based Append-Only Signatures. We present an efficient construction of a provably-secure AOS scheme based on a public-key signature scheme. Let $\mathcal{SGN} = (\mathsf{SGN.G}, \mathsf{SGN.S}, \mathsf{SGN.V})$ be a signature scheme with a space of public keys $\mathsf{SGN.PKSpace}$ and message space $\mathsf{SGN.MSpace} = \mathsf{AOS.MSpace} \times$

$\mathsf{SGN.PKSpace}$. That is, messages to be signed by $\mathcal{SGN}$ are tuples of the form (M, pk), where $M \in \mathsf{AOS.MSpace}$ and $pk \in \mathsf{SGN.PKSpace}$. An AOS signature $\mathtt{Sig}$ of $M[1..n]$ is a tuple $(pk_1,\ sig_1,\ \ldots,\ pk_n,\ sig_n,\ sk_n)$, where for $1 \leq i \leq n$, (pk_i, sk_i) are random public/secret key pairs of the public-key signature scheme $\mathcal{SGN}$ and sig_i is a signature on the tuple (M_i, pk_i) under the secret key sk_{i-1}. The signature sig_0 is signed with the secret key sk_0, which is the signature of ε (the master secret key). Our AOS scheme $\mathcal{AOS}1$ with message space $\mathsf{AOS.MSpace}$ is specified as follows:

– $\mathsf{AOS.Setup}(1^k)$: Run $\mathsf{SGN.G}(1^k)$ to generate a pair (pk_0, sk_0) and returns it as AOS public/secret key pair.
– $\mathsf{AOS.Append}(\mathsf{AOS.pk}, \mathtt{Sig}[M[1..n]], M_{n+1})$: Parse $\mathtt{Sig}$ as $(pk_1,\ sig_1, \ldots, pk_n,\ sig_n,\ sk_n)$. Run $\mathsf{SGN.G}(1^k)$ to generate a pair (sk_{n+1}, pk_{n+1}). Compute $sig_{n+1} \leftarrow \mathsf{SGN.S}_{sk_n}(M_{n+1}, pk_{n+1})$. Return $(pk_1,\ sig_1,\ \ldots,\ pk_{n+1},\ sig_{n+1},\ sk_{n+1})$.
– $\mathsf{AOS.Vfy}(\mathsf{AOS.pk}, M[1..n], \mathtt{Sig})$: Parse $\mathtt{Sig}$ as $(pk_1, sig_1, \ldots, pk_n,\ sig_n, sk_n)$. Set pk_0 to be the master public key $\mathsf{AOS.pk}$. For $i = 1..n-1$ verify that $\mathsf{SGN.V}(pk_{i-1}, sig_i, (M_i, pk_i)) = \mathtt{true}$. If any of the verifications fail, return $\mathtt{false}$. If all the verifications succeed, verify that (sk_n, pk_n) is a valid secret key/public key pair (by signing and veryfing a signature on a random message under sk_n).

Theorem 2. The AOS scheme $\mathcal{AOS}1$ is aos-uf-cma secure assuming that the public-key signature scheme $\mathcal{SGN}$ is *unforgeable under choosen message attacks.*

SHORTER SIGNATURES VIA AGGREGATION. An aggregate signature scheme, $\mathcal{ASGN} = (\mathsf{ASGN.G}, \mathsf{ASGN.S}, \mathsf{ASGN.AGG}, \mathsf{ASGN.V})$, allows the aggregation of n signatures on n distinct messages from n distinct users into a single signature. Its verification algorithm, $\mathsf{ASGN.V}(n, \cdot)$, takes an aggregated signature, n messages, and n public keys and verifies that the n users signed the n messages. When using the certificate-based construction of AOS from Section 3, we can use sequential signature aggregation to shrink the size of the signature (without significantly decreasing security or efficiency). To be more precise, the length of an AOS signature of a message of length n can be condensed to one signature of $\mathcal{ASGN}$, n public keys of $\mathcal{ASGN}$, and one secret key of $\mathcal{ASGN}$. We note that there are two known signature aggregation techniques. The first scheme, given in [2], is based on bilinear maps. The second scheme (only supporting sequential aggregation) is from [10] and can be based on homomorphic trapdoor permutations (such as RSA). Both aggregation schemes are in the random oracle model.

AOS VIA HASH TREES. If the number of symbols in the alphabet $\mathsf{AOS.MSpace}$ is small, AOS can be efficiently implemented using hash trees [11]. This approach suffers from dramatic complexity blowup as the size of the message space increases, but uses only secret-key primitives and provides good security guarantees. We believe that this construction is useful in computationally constrained applications. Here we show how to construct an AOS scheme $\mathcal{AOS}2$ with message space $\mathsf{AOS.MSpace} = \{0,1\}$ and message length restricted to d. The construction uses a pseudorandom generator $G : \{0,1\}^k \to \{0,1\}^{2k}$ and a collision-resistant hash function $H : \{0,1\}^k \times \{0,1\}^k \to \{0,1\}^k$. We denote by $G_i : \{0,1\}^k \to \{0,1\}^k$ the i-th k-bit component of G for $i \in \{0,1\}$.

Consider the graph T depicted in the left part of Figure 1, whose nodes are denoted as shown on the figure (the upper part (UT) in round brackets and the lower part (LT) in square brackets). In general, the graph T has d levels in both upper and lower parts. For any node $u = \langle v_1, \ldots, v_j \rangle$ from the upper part of the graph, we define the complement of u, denoted $\mathrm{Comp}(u)$, to be the minimal set of nodes in $\mathrm{LT} - \{\tilde{\varepsilon}\}$ such that every path from ε to $\tilde{\varepsilon}$ passes through exactly one node from $\{u\} \cup \mathrm{Comp}(u)$. An example of a complement set is given on the right half of Figure 1.

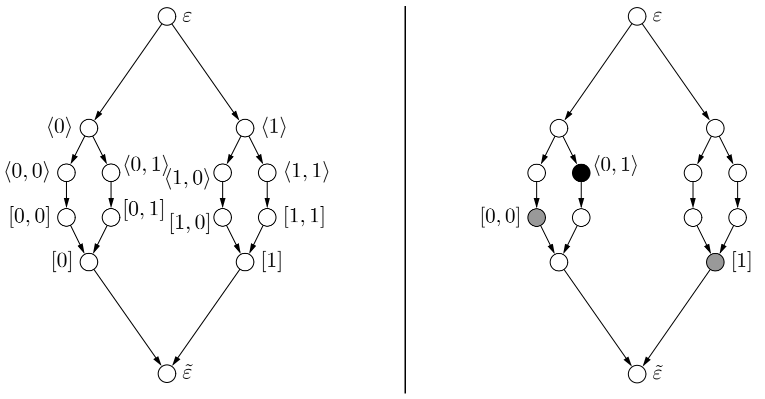

Fig. 1. Structure of the hash-tree construction for $d = 2$. The diagram on the left depicts the hash tree. The diagram on the right highlights the node $u = \langle 0,1 \rangle$ (shown in black) and the set of its complements, $\mathrm{Comp}(u)$ (shown in gray)

In $\mathcal{AOS}2$, a message $M = (M_1, \ldots, M_n)$ is assosiated with a node $u = \langle M_1, \ldots, M_n \rangle$ from the upper part of the graph. Each node u is assosiated with a k-bit value $\mathrm{key}(u)$, which is called the"key" of the message. The setup algorithm assigns these values in a top-down manner, starting from the root node ε. Initially, the root key $\mathrm{key}(\varepsilon)$ is chosen at random from $\{0,1\}^k$. Keys of all the other nodes in the upper part of T are obtain by repeated application of G: keys of left and right children of a node u are $G_0(\mathrm{key}(u))$ and $G_1(\mathrm{key}(u))$, respectively. Keys of the nodes on the highest lower level are obtained by applying G_0 to their parents' keys; keys of the other nodes in the lower part are obtained by applying H to their parents' keys. The setup algorithm outputs $\mathrm{key}(\tilde{\varepsilon})$ as the public key and $\mathrm{key}(\varepsilon)$ as the secret key of AOS.

The signature of a node u consists of the keys in the set $\{u\} \cup \mathrm{Comp}(u)$. Note that given a signature of u, one can compute the keys of all the descendants of u including the last node $\tilde{\varepsilon}$. Verification of a signature is done by computing the keys of all the descendants of u and comparing the obtained key($\tilde{\varepsilon}$) with the public key. The append algorithm, given a signature of $u = \langle M_1, \ldots, M_n \rangle$ and a bit M_{n+1}, computes the key of $u' = \langle M_1, \ldots, M_{n+1} \rangle$ (which is a child of u) and the keys of all the nodes in Comp(u') (which are descendants of u and Comp(u)). It returns these keys as a signature on $(M_1, \ldots, M_{n+1})$.

Theorem 3. If $G(\cdot)$ is a secure pseudorandom generator, $G_0(\cdot)$, $G_1(\cdot)$, are secure one-way functions and $H(\cdot, \cdot)$, $G_0(\cdot)$, and $G_1(\cdot)$ are all collision-resistant hash functions, then $\mathcal{AOS}2$ is aos-uf-cma secure.

AOS via One-Time Signatures. We observe that we can combine the ideas of the certificate-based AOS and the hash-tree AOS to gain a more efficient append-only signature scheme when the message space is small. Assume the message space $\mathsf{AOS.MSpace}$ consists of m elements, where m is a constant. Then we can use our certificate-based construction $\mathcal{AOS}1$ instantiated with an m-time signature scheme. m-time signatures can be efficiently constructed using hash-trees (see [5, 13] for the definition and efficient constructions of one-time and m-time signatures). In addition, the security proof of $\mathcal{AOS}1$ guarantees unforgeability if $\mathcal{SGN}$ is at least an $|\mathsf{AOS.MSpace}|$-time signature scheme.

Compact AOS. In the full version of the paper [9] we show how to use a recent technique of Boneh, Boyen and Goh [1] to get an AOS scheme in which the signature size is proportional to the square root of the length of the message.

4 Relations Between HIBS and AOS

In this section, we show that the concepts of AOS and Hierarchical Identity-based Signatures (HIBS) are in fact equivalent.

We start with a formal defintion of HIBS. Let $\mathsf{HIBS.IDSpace}$ be any set of identities (typically $\{0,1\}^*$). A *hierarchical identity of length* n is an n-tuple of identities from $\mathsf{HIBS.IDSpace}$, written as $I[1..n] = (I_1, I_2, \ldots, I_n)$. The *root identity* is denoted as $I[1..0]$ or ε. Again we use the symbol $\sqsubseteq$ to denote the prefix relation over the set of hierarchical identities.

A HIBS scheme over identity space $\mathsf{HIBS.IDSpace}$ is made up of four (possibly randomized) algorithms: a setup algorithm $\mathsf{HIBS.Setup}$, a key delegation algorithm $\mathsf{HIBS.KeyDel}$, a signing algorithm $\mathsf{HIBS.Sign}$ and a verification algorithm $\mathsf{HIBS.Vfy}$.

Definition 4. Given a HIBS scheme $\mathcal{HIBS}$ = ($\mathsf{HIBS.Setup}$, $\mathsf{HIBS.KeyDel}$, $\mathsf{HIBS.Sign}$, $\mathsf{HIBS.Vfy}$), security parameter k and adversary $\mathcal{A}$ consider the following experiment:

```
Experiment $\mathbf{Exp}^{\text{hibs-uf-cma}}_{\mathcal{HIBS},\mathcal{A}}(k)$
  $IDSet \leftarrow \emptyset$
  $(\mathsf{HIBS.pk}, \mathsf{HIBS.SK}[\varepsilon]) \leftarrow \mathsf{HIBS.Setup}(1^k)$
  $(I[1..n], M, \mathtt{sig})$
            $\leftarrow \mathcal{A}^{\text{Corrupt}(\cdot),\text{Sign}(\cdot,\cdot)}(\mathsf{HIBS.pk})$
  if $\mathsf{HIBS.Vfy}(I[1..n], M, \mathtt{sig}) = \mathtt{true}$
    and $\forall\ j \leq n\ I[1..j] \notin IDSet$
    and $(I[1..n], M) \notin MSGSet$
  then return 1 else return 0
```

```
Oracle Corrupt($I[1..n]$)
  $IDSet \leftarrow IDSet \cup \{I[1..n]\}$
  return Extract($I[1..n]$)

Oracle Sign($I[1..n], M$)
  $MSGSet \leftarrow MSGSet \cup \{(I[1..n], M)\}$
  $sk \leftarrow$ Extract($I[1..n]$)
  return $\mathsf{HIBS.Sign}(sk, I[1..n], M)$
```

```
Oracle Extract($I[1..i]$) // defined recursively
    if $i = 0$ return $\mathsf{HIBS.SK}[\varepsilon]$
    else if $\mathsf{HIBS.SK}[I[1..i]] =$ defined
        then return $\mathsf{HIBS.SK}[I[1..i]]$
        else $sk \leftarrow \mathsf{HIBS.KeyDel}(\mathsf{HIBS.pk}, I[1..i-1], \text{Extract}(I[1..i-1]), I_i)$
            return $sk$
```

The hibs-uf-cma-advantage of an adversary $\mathcal{A}$ in breaking the security of the scheme $\mathcal{HIBS}$ is defined as $\mathbf{Adv}^{\text{hibs-uf-cma}}_{\mathcal{HIBS},\mathcal{A}}(k) = \Pr[\,\mathbf{Exp}^{\text{hibs-uf-cma}}_{\mathcal{HIBS},\mathcal{A}}(k) = 1\,]$, and $\mathcal{HIBS}$ is said to be *existentially unforgeable under chosen message attacks* (hibs-uf-cma secure) if the above advantage is a negligible function in k for all polynomial time adversaries $\mathcal{A}$.

Constructing AOS from HIBS. We set $\mathsf{AOS.MSpace} = \mathsf{HIBS.IDSpace}$ and associate an AOS message $(M_1, \ldots, M_n)$ of length n with the hierarchical identity $I[1..n] = (M_1, \ldots, M_n)$ of depth n. We then define the signature of this message as the secret key $\mathsf{HIBS.SK}[I[1..n]]$ of $I[1..n]$. Given the above analogy between signatures of messages and secret keys of hierarchical identities, we construct an AOS scheme given a HIBS scheme as follows. Appending to a given signature in $\mathcal{AOS}$ is done using key delegation in $\mathcal{HIBS}$. The verification of an AOS signature $\mathsf{HIBS.SK}[I[1..n]]$ is done by signing a random message $M \in \mathsf{HIBS.MSpace}$ under the secret key $\mathsf{HIBS.SK}[I[1..n]]$ and verifying that the resulting signature is valid. A formal construction is given in the full version [9].

Theorem 5. If the HIBS scheme $\mathcal{HIBS} = (\mathsf{HIBS.Setup}, \mathsf{HIBS.KeyDel}, \mathsf{HIBS.Sign}, \mathsf{HIBS.Vfy})$ is hibs-uf-cma secure, then the above AOS scheme is aos-uf-cma secure.

Constructing HIBS from AOS. A naive approach to building a HIBS scheme from an AOS scheme would be as follows: for any hierarchical identity $I[1..n]$, define $\mathsf{HIBS.SK}[[]I[1..n]]$ as the AOS signature on $I[1..n]$ and the HIBS signature created with $\mathsf{HIBS.SK}[[]I[1..n]]$ on message M as the AOS signature formed by appending M to $\mathsf{HIBS.SK}[[]I[1..n]]$. However, it can be shown that such a scheme is insecure. Our tweak is to insert a unique identifier to separate identities and messages. Let $\mathcal{AOS} = (\mathsf{AOS.Setup}, \mathsf{AOS.Append}, \mathsf{AOS.Vfy})$ be an AOS scheme with message space $\mathsf{AOS.MSpace}$. Let $\mathsf{HIBS.IDSpace}$ and $\mathsf{HIBS.MSpace}$ be subsets of $\mathsf{AOS.MSpace}$ such that there is some symbol Δ from the AOS message space which is not a valid identity for the HIBS scheme (Δ can still be in the

HIBS message space). Then we can construct a HIBS scheme with identity space HIBS.IDSpace and message space HIBS.MSpace as follows:

Construction 6. $\mathcal{HIBS} = (\mathsf{HIBS.Setup}, \mathsf{HIBS.KeyDel}, \mathsf{HIBS.Sign}, \mathsf{HIBS.Vfy})$:

– $\mathsf{HIBS.Setup}(1^k)$: Run the $\mathsf{AOS.Setup}(1^k)$ to generate a pair ($\mathsf{AOS.pk}$, $\mathsf{Sig}[\varepsilon]$); output it as the master public/private key pair for $\mathcal{HIBS}$.
– $\mathsf{HIBS.KeyDel}(\mathsf{HIBS.pk}, \mathsf{HIBS.SK}[I[1..n]], I_{n+1})$: The delegation algorithm interprets $\mathsf{HIBS.SK}[I[1..n]]$ as an $\mathcal{AOS}$ signature of $I[1..n]$. It appends to the signature a symbol I_{n+1} and outputs the resulting signature as the secret key of $I[1..n+1]$.
– $\mathsf{HIBS.Sign}(\mathsf{HIBS.pk}, \mathsf{HIBS.SK}[I_n], M)$: The signing algorithm for $\mathcal{HIBS}$ interprets $\mathsf{HIBS.SK}[I[1..n]]$ as an $\mathcal{AOS}$ signature of $I[1..n]$. It appends a symbol $\varDelta$ to $\mathsf{HIBS.SK}[I[1..n]]$ and then appends the message M to the resulting AOS signature to get the final signature `sig`.
– $\mathsf{HIBS.Vfy}(\mathsf{HIBS.pk}, I[1..n], M, \mathtt{sig})$: The verification algorithm for $\mathcal{HIBS}$ verifies if `sig` is a valid AOS signature of $(I_1, \ldots, I_n, \varDelta, M)$.

The following theorem is proven in the version [9]:

Theorem 7. If the AOS scheme $\mathcal{AOS} = (\mathsf{AOS.Setup}, \mathsf{AOS.Append}, \mathsf{AOS.Vfy})$ is aos-uf-cma secure, then the HIBS scheme $\mathcal{HIBS}$ from Construction 6 is hibs-uf-cma secure.

5 Applications

An important application of AOS is in the construction of secure routing protocols for the Internet. The Border Gateway Protocol (BGP), which is the primary routing protocol used today in the Internet, has some well-known security weaknesses which require cryptographic solutions. While there have been many proposals for securing BGP in the past [8, 6], each must develop its own cryptographic constructions due to the lack of any primitive designed specifically for this application. In the discussion below, we briefly describe Internet routing and explain how our primitive is useful for ensuring important security requirements in BGP.

The Internet is composed of various autonomous systems (ASes), each having control over some portion of the IP address space. BGP is the protocol used to spread information about the routes to all IP addresses in this network of ASes. Initially, all ASes advertise the IP addresses they own to their neighboring ASes. Upon receipt of such advertisements, each neighbor records this information, appends itself to the advertized route and sends the new information further down. The process repeats with the next AS in the chain and eventually, all ASes learn a route to the originating AS (In case an AS receives two or more routes to the same IP address, it selects one of them based on some local policy). Authenticity of route announcements is essential for ensuring the correct behaviour of BGP for otherwise, malicious ASes can play havoc with the Internet traffic. For example, if an AS truncates the route in some advertisement or modifies it selectively, it could convince its neighbors to forward all their traffic to it, which it could then modify or drop at will (incidents of this nature have indeed occured in the recent past [6]).

Append-only Signatures are a useful tool in addressing this problem. Suppose that an AS R_0 wishes to announce routes for some IP prefix it owns. It first generates an AOS public-private key pair, distributes the public key $\mathsf{AOS.pk}$ throughout the network (this can be done through a PKI as in [8, 6]) and to every neighboring AS R_{i_1}, sends the usual BGP information along with the AOS signature $\mathsf{AOS.Append}(\mathsf{AOS.pk}, \mathsf{Sig}[\varepsilon], R_{i_1})$. In order to continue the advertisement process, R_{i_1} sends to each of its own neighbors R_{i_2} a BGP announcement containing the route (R_0, R_{i_1}) and the signature $\mathsf{AOS.Append}(\mathsf{AOS.pk}, \mathtt{Sig}[R_{i_1}], R_{i_2})$. In other words, R_0 appends the label of its neighbor R_{i_1} into the AOS signature chain and R_{i_1} further appends the label of R_{i_2} into it. The advertisement process continues in this manner until all ASes in the network receive information about a route to R_0. Each recipient can verify the validity of the announced route using the public key $\mathsf{AOS.pk}$. If the AOS scheme is secure, then all that a malicious AS can do now is to append one of its neighbors into the AOS signature chain (since each R_i can check that the AS it receives a route from was the last to be appended before R_i). In practice, the number of path advertisements received by an AS from any given source AS is extremely small: as observed in real routing data [6], the odds that an AS receives more than 15 path advertisements coming from the same source are about 1 in a 1000. This enables us to use m-time signature schemes (with $m = 15$) for an efficient AOS with reasonable security guarantee. For more details and other applications of AOS, see [9].

6 Final Remarks and Open Problems

Finalization of AOS Signature. A property of append-only signature schemes which might be needed by some applications is the ability to "finalize" the signature, that is, to modify the signature of a message in a way that prohibits any further appending. The general solution to this problem is to use a special symbol Θ (from the message space) to denote the end of the message. When one wants to finalize the signature of some message, he should append Θ to the signature. Messages that contain symbol Θ in the middle of the message (not as the last symbol) are therefore considered to be invalid.

Restricted AOS. In AOS, anyone can append and verify signatures. In certain scenarios, however, one may want to restrict the ability to append messages to a limited group of users. Still, anyone should be able to verify the signatures. We call this extension of AOS *Restricted Append-Only Signatures* (RAOS). Using a symmetric encryption scheme we show in the full version [9] how to modify a given AOS scheme to get an RAOS scheme.

Shorter AOS signatures. Given that wide-area routing protocols propagate a large number of messages, compact signatures are desirable. Thus we raise an open problem of whether it is possible to build an AOS scheme with constant signature length (in both message length and maximal message length). This problem is equivalent to building a HIBS scheme where secret keys of the users have constant length (in the depth of the given user in the hierarchy and in the maximal depth of the hierarchy).

Acknowledgments

We thank Mihir Bellare (for suggesting an improvement to the proof of Theorem 2) and Daniele Micciancio (for useful insight about the definition of AOS). Thanks also to the anonymous reviewers for helpful comments.

References

1. D. Boneh, X. Boyen and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In *Proceedings of EUROCRYPT 2005*, LNCS, 2005.
2. Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *Proceedings of EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 416–432, 2003.
3. S. S. M. Chow, L. C. K. Hui, S. M. Yiu, and K. P. Chow. Secure hierarchical identity based signature and its application. In *Proceedings of ICICS 2004*, pages 480–494, 2004.
4. Craig Gentry and Alice Silverberg. Hierarchical id-based cryptography. In *Proceedings of ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 548–566, 2002.
5. Alejandro Hevia and Daniele Micciancio. The provable security of graph-based one-time signatures and extensions to algebraic signature schemes. In *Proceedings of ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 379 – 396, 2002.
6. Yih-Chun Hu, Adrian Perrig, and Marvin Sirbu. SPV: secure path vector routing for securing BGP. In *Proceedings of the ACM SIGCOMM* , pages 179–192, 2004.
7. Robert Johnson, David Molnar, Dawn Xiaodong Song, and David Wagner. Homomorphic signature schemes. In *Proceedings of CT-RSA 2002*, volume 2271 of *LNCS*, pages 244–262, 2002.
8. Stephen Kent, Charles Lynn, and Karen Seo. Secure border gateway protocol (S-BGP). In *IEEE Journal on Selected Areas in Communications*, 18(4):582–592, 2000.
9. Eike Kiltz, Anton Mityagin, Saurabh Panjwani and Barath Raghavan. Append-Only Signatures. Full version. `http://eprint.iacr.org/2005/124`
10. Anna Lysyanskaya, Silvio Micali, Leonid Reyzin, and Hovav Shacham. Sequential aggregate signatures from trapdoor permutations. In *Proceedings of EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 74–90, 2004.
11. Ralph C. Merkle. A digital signature based on a conventional encryption function. In *Proceedings of CRYPTO'87*, volume 293 of *LNCS*, pages 369–378, 1988.
12. Silvio Micali and Ronald L. Rivest. Transitive signature schemes. In *Proceedings of CT-RSA 2002*, volume 2271 of *LNCS*, pages 236–243, 2002.
13. Leonid Reyzin and Natan Reyzin. Better than biba: Short one-time signatures with fast signing and verifying. In *Proceedings of 7th Australasian Conference ACSIP*, 2002.
14. Adi Shamir. Identity-based cryptosystems and signature schemes. In *Proceedings of CRYPTO'84*, volume 196 of *LNCS*, pages 47–53, 1985.