Indistinguishability of One-Way Accumulators

Hermann de Meer, Manuel Liedel, Henrich C. Pöhls, Joachim Posegga, Kai Samelin
demeer@uni-passau.de,
manuel.liedel@wiwi.uni-regensburg.de,
{hp,jp,ks}@sec.uni-passau.de,

Technical Report, Number MIP-1210
Department of Informatics and Mathematics
University of Passau, Germany
December 2012

Indistinguishability of One-Way Accumulators

Hermann de Meer[1,4], Manuel Liedel[2*], Henrich C. Pöhls[3,4**],
Joachim Posegga[3,4], Kai Samelin[1,4***]

[1] Chair of Computer Networks and Communications, University of Passau, Germany
[2] University of Regensburg, Germany
[3] Chair of IT-Security, University of Passau, Germany
[4] Institute of IT-Security and Security Law (ISL), University of Passau, Germany
`demeer@uni-passau.de`, `manuel.liedel@wiwi.uni-regensburg.de`,
`{hp,jp,ks}@sec.uni-passau.de`

Abstract. *One-Way Accumulators* have been introduced by *Benaloh* and *de Mare* at Eurocrypt '93. They allow to hash a potentially very large set into a short digest, called the accumulator. The accumulator allows to verify the membership of a given element using corresponding witnesses. State-of-the-Art research focuses on the collision-resistance of the resulting schemes. However, there are many applications, where the accumulator must be hiding, i.e., if a third party does not have all members, it should not be able to decide how many additional members a given accumulator has. This behavior of *indistinguishability* is already used in many cryptographic applications, but has neither been formalized nor formally proven. In this paper, we close this gap by proving that the construction by *Barić* and *Pfitzmann*, presented at Eurocrypt '97, fulfills our new notion. In particular, their accumulator is perfectly indistinguishable. Moreover, we show that the accumulator presented at FSE '96 by *Nyberg* does not fulfill this requirement.

Keywords: One-Way Accumulators, Privacy, Hash-Functions

1 Introduction

Cryptographic accumulators allow to hash a potentially very large set $\mathcal{M} = \{y_1, \ldots, y_\ell\}$ with ℓ elements into a short digest a, called the ac-

* The research leading to these results was supported by "Regionale Wettbewerbsfähigkeit und Beschäftigung", Bayern, 2007-2013 (EFRE) as part of the SECBIT project (`www.secbit.de`).

** Is funded by BMBF (FKZ:13N10966) and ANR as part of the ReSCUeIT project

*** The research leading to these results was supported by "Regionale Wettbewerbsfähigkeit und Beschäftigung", Bayern, 2007-2013 (EFRE) as part of the SECBIT project (`www.secbit.de`) and the European Community's Seventh Framework Programme through the EINS Network of Excellence (grant agreement no. [288021]).

cumulator. They have first been introduced by *Benaloh* and *de Mare* at Eurocrypt '93 [3]. Its applications are broad and range from storage efficient protocols to anonymous credential systems [3,7]. The most used main building block is a (one-way) function $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{X}$, which fulfills an additional property named quasi-commutativity:

$$\forall x \in \mathcal{X}, y_1, y_2 \in \mathcal{Y} : f(f(x, y_1), y_2) = f(f(x, y_2), y_1)$$

The accumulator introduced in [3] uses the RSA-function [21] as the basic underlying function. In particular, it lets $(x, y) \mapsto x^y \bmod n$, where $n = pq$ is a RSA-modulus with safe primes, i.e., $p = 2p' + 1$ and $q = 2q' + 1$ and p' and q' are primes. Clearly, f is quasi-commutative:

$$\forall x \in \mathcal{X}, y_1, y_2 \in \mathcal{Y} : (x^{y_1} \bmod n)^{y_2} \bmod n = (x^{y_2} \bmod n)^{y_1} \bmod n$$

Consequently, to extent this to more than two elements, one simply calculates:

$$a = b^{\prod_{i=1}^{\ell} y_i} \bmod n$$

where $b \in_R \mathcal{X}$, and $\mathcal{X} = \mathcal{Y} = (\mathbb{Z}/n\mathbb{Z})$. Note, it is required that the elements y_i are hashed using a random oracle $\mathcal{H} : \{0,1\}^* \to \{0,1\}^\lambda$, prior to accumulating to be collision-resistant [2,3]. Here, λ is a security parameter. To keep the introduction simple, this hashing step is left out for now. b is a randomly chosen "starting value" [3]. The modulus n can be seen as the key of f. For the rest of this paper, we drop this index, if it is appropriate to do so. To verify that a given element $y_i \in \mathcal{M}$ was actually part of the calculation of the accumulator a, a corresponding witness p_i can be calculated, which is essentially a y_i^{th}-root of a, i.e., $p_i = \sqrt[y_i]{a} \bmod n$, or, in other words, $p_i = b^{\prod_{j=1, i \neq j}^{\ell} y_j} \bmod n$. Therefore, $p_i^{y_i} = a \bmod n$ yields, and a third party can verify that the value y_i was accumulated into a. Hence, for each element $y_i \in \mathcal{M}$, there exists a witness $p_i \in \mathcal{X}$ which proves that y_i was actually accumulated. Obviously, to result in a meaningful cryptographic construction, the function needs to be one-way, i.e., it is hard to find a **new** pair $(x', y') \neq (x, y)$, for which $f(x', y') = f(x, y)$ yields, if the factorization of n is unknown [3]. Note, we do not consider that roots can be calculated, if the factorization of $n = pq$ is known. In other words, the entity actually generating the parameters for the accumulator is not considered adversarial. We give an algorithmic description and formal security definitions in Sect. 2.

Motivation. All existing work on the primitive of accumulators only focuses on the unforgeability or its possible usages. As an example, the original application scenario given in [3] proposes using accumulators for time-stamping [11]. In particular, all documents are accumulated into the accumulator a, which is finally time-stamped by the time-stamping service. Following this approach, only a has to be time-stamped, while neither elements nor the corresponding witnesses need to be known by the time-stamping service. Hence, the time-stamping service does not know what documents it timestamps. However, as we show in Sect. 3, not all existing accumulators hide the amount of the members actually contained inside the accumulator a. As an example, the accumulator introduced by *Nyberg* at FSE '96 does not hide the amount of digested elements [18]. We prove this claim formally in Sect. 3. Assume that the entity which needs to time-stamp a set of documents only lets its invoices be time-stamped. The one-way property of the accumulator definition already implies that the time-stamping service cannot derive which invoices are actually contained, but it may derive *how many* invoices are time-stamped. This is already leaks, potentially critical, financial information about the business, as it allows to infer how many invoices have been issued and therefore "how good the business is performing". This insider information can be used in the stock market to help forecasting the business' share prices. From the privacy perspective this is obviously not acceptable. Consequently, one requires a privacy-preserving accumulator to hide the amount of members, i.e., a third party must not be able to derive how many elements were used to calculate the given accumulator. This behavior has already been utilized and assumed in many applications, e.g., in authenticated dictionaries [10], redactable signatures [1,12,19,20], sanitizable signatures [8,14], the already mentioned time-stamping scenario [3], revocation checks [13,17] and anonymous credentials [7]. We show that, however commonly used, not all accumulator constructions do fulfill the indistinguishability requirement. In particular, we prove that the accumulator introduced in [2] does hide the amount of elements, if the parameters are chosen correctly, while the accumulator by *Nyberg* cannot achieve our indistinguishability notion [18].

State of the Art. The first one-way accumulator has been introduced by *Benaloh* and *de Mare* at Eurocrypt '93 [3]. They have been extended to collision-free accumulators in [2]. Based on this work, *Sander* derived a trapdoor-free RSA-accumulator by generating the modulus n with unknown factorization in a verifiable way. A different approach to implement trapdoor-free accumulators, based on Bloom-Filters [4], has been proposed by *Nyberg* [18]. Her approach however, is not indistinguishable due to the underlying Bloom-Filter. We prove this claim in Sect. 3. Undeniable accumulators have then been introduced in [6]. These accumulators do not allow to generate a *non-membership* witness, if the corresponding value has been accumulated, and vice versa. Universal accumulators, introduced in [7], allow to dynamically add and remove values from the accumulator. Additional trapdoor-free accumulators have been introduced in [16,22].

All approaches focus on the unforgeability, i.e., the collision-resistance, of the resulting accumulator, while there exists no work on the hiding property of accumulators. This paper addresses this gap. We want to emphasize that the scenario where the generator of the parameters is malicious, was left as open work in [3], tackled by introducing trusted third parties in [15] and finally solved by *Sander* [22] and also, in a slightly different setting, by *Lipmaa* [16] and *Nyberg* [18]. *Lipmaa* uses a concept related to the CRS-model [5] to verify that the parameters were generated honestly [16], while *Nyberg* relies on a Bloom-Filter. In this work, we focus on the original setting, i.e., the entity generating the parameters is not considered adversarial. Moreover, we do not discuss more sophisticated accumulators which allow removing elements [7] or generating non-membership witnesses [15]. We also do not discuss batch-updates, as introduced in [23]. However, our results, in particular the formal notion of indistinguishability, remain generally applicable.

Our Contribution and Outline. This paper proves that there exists one-way accumulators which hide the actual amount of accumulated elements. On the other hand, we also show that there are one-way accumulators which do not fulfill our new privacy notion of indistinguishability. The rest of the paper is structured as follows: in Sect. 2, all preliminaries are presented. The new notion of indistinguishability is introduced in Sect. 3. This section also covers the proofs that *Nyberg*'s accumulator [18] is not indistinguishable, while the one of *Barić* and *Pfitzmann* [2] is. This result shows that only some accumulators can be used in applications scenarios where our stronger privacy guarantee is crucial, e.g., hiding the contents **and the number** of documents submitted to time-stamping.

The rest of the paper is structured as follows. In Sect. 2, the nomenclature and the existing security model are revisited. We discuss the new notion of indistinguishability in Sect. 3. This section also contains the proofs that the accumulator by *Barić* and *Pfitzmann* [2] is indistinguishable if used correctly, while *Nyberg*'s [18] is not. We conclude our work in Sect. 4.

2 Preliminaries

For a set $\mathcal{M} = \{y_1, \ldots, y_\ell\}$, we call y_i an element and ℓ the element count, i.e., ℓ is the cardinality of $\mathcal{M}$. p and q always denote safe primes, i.e, $p = 2p' + 1$ and $q = 2q' + 1$, where p' and q' are also primes. $n = pq$ denotes a product of two safe primes. $\mathcal{H} : \{0,1\}^* \to \{0,1\}^\lambda$ denotes a random oracle, while λ denotes the security parameter. The following notations and the algorithmic description are derived from [9] and [16]. $\varphi : \mathbb{N} \to \mathbb{N}$ is *Euler*'s totient function. $\mathbb{P}_n$ denotes the set of odd primes less or equal to n, i.e., $\mathbb{P}_n = \{m \mid 2 < m \leq n, m \text{ is prime}\}$. With $\mathrm{ord}_p(n)$, we denote the order of the element n in the group $(\mathbb{Z}/p\mathbb{Z})$.

Definition 1 (Cryptographic Accumulators). *A cryptographic accumulator* ACC *consists of five efficient (PPT) algorithms. In particular,* $\mathsf{ACC} := (\mathsf{Setup}, \mathsf{Dig}, \mathsf{Proof}, \mathsf{Verf})$ *such that:*

Setup. *The algorithm* Setup *is the parameter generator. On input of the security parameter* λ*, it outputs the public parameter* parm*, i.e.,* $\mathsf{parm} \leftarrow \mathsf{Setup}(1^\lambda)$*. Hence,* Setup *can be interpreted as the instance generator*

Dig. *The algorithm* Dig *takes as input the set* $\mathcal{M} = \{y_1, \ldots, y_\ell\}$*,* $y_i \in \mathcal{Y}_{\mathsf{parm}}$ *to accumulate, the public parameters* parm *and outputs an accumulator value* a*, i.e,* $a \leftarrow \mathsf{Dig}(\mathsf{parm}, \mathcal{M})$

Proof. *The algorithm* Proof *takes as input the public parameters* parm*, a value* $y_i \in \mathcal{Y}_{\mathsf{parm}}$ *and returns a witness* p_i *from a witness space* $\mathcal{P}_{\mathsf{parm}}$*, if* $y_i \in \mathcal{M}$ *was input to* Dig*, i.e.,* $\mathsf{Dig}(\mathsf{parm}, \mathcal{M})$*, and* $\perp$ *otherwise. Hence, it outputs* $p_i \leftarrow \mathsf{Proof}(\mathsf{parm}, y_i, \mathcal{M})$

Verf. *The verification algorithm* Verf *takes as input the public parameters* parm*, an accumulator* $a \in \mathcal{X}_{\mathsf{parm}}$*, a witness* p_i*, and a value* $y_i \in \mathcal{Y}_{\mathsf{parm}}$ *and outputs a bit* $d \in \{\mathtt{true}, \mathtt{false}\}$ *indicating whether* p_i *is a valid proof that* y_i *has been accumulated into* a*. Hence, it outputs a decision* $d \leftarrow \mathsf{Verf}(\mathsf{parm}, a, y_i, p_i)$

We also require the correctness requirements to hold. In particular, for any security parameter $\lambda \in \mathbb{N}^+$, any $\mathsf{parm} \leftarrow \mathsf{Setup}(1^\lambda)$, any $\mathcal{M} = \{y_1, \ldots, y_\ell\}$ let $a \leftarrow \mathsf{Dig}(\mathsf{parm}, \mathcal{M})$. We require:

$$\forall y_i \in \mathcal{M} : \mathsf{Verf}(\mathsf{parm}, a, y_i, \mathsf{Proof}(\mathsf{parm}, y_i, \mathcal{M})) = \mathtt{true}$$

```
**Experiment** $\mathsf{Indistinguishability}_{\mathcal{A}}^{\mathsf{ACC}}(\lambda)$
    $\mathsf{parm} \leftarrow \mathsf{Setup}(1^{\lambda})$
    $b \xleftarrow{\$} \{0,1\}$
    $d \leftarrow \mathcal{A}^{\mathsf{LoRHash}(\cdot,\cdot,b,\mathsf{parm})}(\mathsf{parm})$
        where oracle $\mathsf{LoRHash}$ for input $\mathcal{S}, \mathcal{R}$:
        $\{z\} \xleftarrow{\$} \mathcal{Y}_{\mathsf{parm}}$
        if $b = 1$:
            return $(\mathsf{Dig}(\mathsf{parm}, \mathcal{S} \cup \mathcal{R} \cup \{z\}), \{(y_i, p_i) \mid p_i \leftarrow \mathsf{Proof}(\mathsf{parm}, y_i, \mathcal{S} \cup \mathcal{R} \cup \{z\}), y_i \in \mathcal{S}\})$
        if $b = 0$:
            return $(\mathsf{Dig}(\mathsf{parm}, \mathcal{S} \cup \{z\}), \{(y_i, p_i) \mid p_i \leftarrow \mathsf{Proof}(\mathsf{parm}, y_i, \mathcal{S} \cup \{z\}), y_i \in \mathcal{S}\})$
    return 1, if $d = b$
```

Fig. 1. Game for Indistinguishability

3 Indistinguishability

In this section, we introduce the formal definition of indistinguishability. In a nutshell, indistinguishability requires that an adversary cannot decide how many members a given accumulator has. We define this notion formally with the next definition.

Definition 2 (Indistinguishability). *We call an accumulator indistinguishable, if for any $\mathcal{PPT}$ algorithm $\mathcal{A}$ the probability that the experiment depicted in Fig. 1 returns* 1 *is negligibly close to* $\frac{1}{2}$.

The basic idea is that an adversary can choose two sets. The oracle either digests the union of it or just the first one. Additionally, it adds a blinding value z, chosen at random. This randomly chosen element accounts for deterministic accumulators. z can be seen as the starting value b for the standard RSA-accumulator. The adversary is then given only the proofs for the first set. It has then to decide if both sets or just the first set along with z has been digested. This definition also covers accumulators without a starting value, e.g., the one introduced by *Nyberg* [18]. We prove that this additional blinding value does not have an impact on our proofs.

3.1 *Nyberg*'s Accumulator

Here, we restate the construction of the accumulator by *Nyberg* [18]. We use this construction to prove afterwards, that her accumulator does not fulfill our notion of indistinguishability.

Construction 1 (*Nyberg*'s Accumulator) *Let, ACC* $:= (\mathsf{Setup}, \mathsf{Dig}, \mathsf{Proof}, \mathsf{Verf})$ *such that:*

Setup. *Let* $N = 2^d$ *be an upper bound of elements to be accumulated. Furthermore, let* $r \in \mathbb{N}^+$ *be an additional integer. Let* $\mathcal{H} : \{0,1\}^* \rightarrow \{0,1\}^\lambda$ *be a one-way function, modeled as a random oracle, where* $\lambda = rd$*. Output* $\mathsf{parm} = (r, d, N, \mathcal{H})$*. Note, we omit the key space of* $\mathcal{H}$ *for simplicity*

Dig. *Let* $\mathcal{Z} = \{z_i \mid z_i \leftarrow \mathcal{H}(y_i), y_i \in \mathcal{M}\}$, $\mathcal{M} = \{y_1, \ldots, y_\ell\}$ *denote the set of hashed elements. Note, the length of an element* z_i *is equal to* $\lambda = rd$ *due to* $\mathcal{H}$*. Let* $z_i = (z_{i,1}, \ldots, z_{i,r})$ *denote the list of bit strings of* r *elements with length* d *corresponding to* z_i*. Map each element* z_i *to a binary string* $b_j = (b_{i,1}, \ldots, b_{i,r})$ *of length* r*, where* $b_{i,j} = 0$*, if* $z_{i,j} = 0$ *and* $b_{i,j} = 1$*, if* $z_{i,j} \neq 0$*. Informally this means replacing the* d *bits of* $z_{i,j}$ *by 0 or 1 depending on whether* $z_{i,j} = 0$ *or* $z_{i,j} = 1$*. The accumulated hash value* a *is now defined as the coordinate-wise product* $\bmod 2$ *of each binary* r*-tuple* b_i*, i.e.,* $a_i \leftarrow \prod_{j=1}^{|\mathcal{S}|} b_{i,j} \bmod 2$*. Output* $a = (a_1, \ldots, a_r)$

Proof. *The algorithm* Proof *returns* $\mathsf{Dig}(\mathsf{parm}, \mathcal{M})$*, i.e., the accumulator* a

Verf. *To verify that a given value* y_i *was accumulated into* $a = (a_p, \ldots, a_r)$*, one calculates* $z_i \leftarrow \mathcal{H}(y_i)$ *and the corresponding bit string* $b_i = (b_{i,1}, \ldots, b_{i,r})$*. Afterwards, it checks that for all* $j = 1, \ldots, r$ *that, if* $b_{i,j} = 0$ *then* $a_j = 0$*. Note, the accumulator itself is considered the proof*

Non-Indistinguishability of *Nyberg*'s Construction. In this section, we prove that *Nyberg*'s construction is not indistinguishable and therefore cannot be used in applications where this privacy notion is required.

Theorem 1 (*Nyberg*'s Accumulator is not Indistinguishable). *The accumulator by Nyberg [18] does not fulfill our notion of indistinguishability. Please note that we do not prove anything related to the collision-resistance of the accumulator.*

Proof. For this proof, we assume that each bit is independently set to 0, or 1 resp., by $\mathcal{H}$ with probability exactly 0.5, which is implied by the use of the random oracle. This has already been assumed in *Nyberg*'s original work to prove the collision-resistance of her accumulator [18]. Hence, also following *Nyberg* [18], the probability that a $b_{i,j}$ is equal to 0 is 2^{-d}, i.e., $\Pr[b_{i,j} = 0] = 2^{-d}$. Hence, the expected number of $a_i = 0$, i.e., $\sum_{a_i=0} 1$, is equal to $r2^{-d}$ for a single element accumulated. This single element is treated as $\{z\}$, as defined in the game given in Fig. 1.

For m hashed elements, the expected number of $a_i = 1$ equals $r((1 - 2^{-d})^m)$. Obviously, this is a monotonous function $f_{r,d}(m)$, decreasing with m, as d and r are constants. In terms of the formal game, the adversary chooses a random string a. It sets $\mathcal{R} = \emptyset$ and $\mathcal{S} = \{a\}$. It follows that $r((1 - 2^{-d})^0) \geq r((1 - 2^{-d})^1)$. The case that adding a new element does not change the accumulator only happens with negligible probability, as this implies a collision. Refer to [18] for a thorough discussion and the corresponding probabilities concerning collisions. Thus, the number of 0s, or 1s resp., allows an approximation of the number of elements accumulated. Hence, the adversary wins the game with non-negligible probability. This proves Th. 1. $\square$

3.2 *Barić* and *Pfitzmann*'s Accumulator

In this section, we restate the construction by *Barić* and *Pfitzmann* [2] and prove the indistinguishability under certain assumptions. Note, our proofs only focus on the indistinguishability, as the collision-resistance has already been proven in the original work [2].

Construction 2 (*Barić* and *Pfitzmann*'s Accumulator) *A cryptographic accumulator* ACC *consists of five efficient (PPT) algorithms. In particular,* $\mathsf{ACC} := (\mathsf{Setup}, \mathsf{Dig}, \mathsf{Proof}, \mathsf{Verf})$ *such that:*

Setup. *The algorithm* Setup *is the parameter generator. On input of the security parameter* λ*, it outputs the parameter* parm*, i.e., the RSA-modulus* n*. To do so, it picks two safe primes* p *and* q *of bit-length* λ*. Additionally, it chooses a hash-function* $\mathcal{H} : \{0,1\}^* \to \mathbb{P}_n)$*, modeled as a random oracle. Finally, it outputs* $(\mathcal{H}, n)$*, where* $n = pq$*.*

Dig. *The algorithm* Dig *takes as input the set* $\mathcal{M} = \{y_1, \ldots, y_\ell\}$ *to accumulate, the public parameters* $\mathsf{parm} = (\mathcal{H}, n)$ *and outputs an accumulator value* a*. It picks a random starting value* $b \xleftarrow{\$} (\mathbb{Z}/n\mathbb{Z})^\times$*. Afterwards, it sets* $a \leftarrow b^{\prod_{i=1}^{\ell} \mathcal{H}(y_i)} \bmod n$*. Finally, it outputs* a

Proof. *The algorithm* Proof *takes as input the public parameters* $\mathsf{parm} = (\mathcal{H}, n)$*, the set* $\mathcal{M} = \{y_1, \ldots, y_\ell\}$ *and an element* $y_i \in \mathcal{M}$*. It outputs* $p_i \leftarrow b^{\prod_{j=1, i \neq j}^{\ell} \mathcal{H}(y_j)} \bmod n$

Verf. *The verification algorithm* Verf *takes as input the public parameters* parm*, an accumulator* $a \in \mathcal{X}_{\mathsf{parm}}$*, a witness* p_i*, and a value* $y_i \in \mathcal{Y}_{\mathsf{parm}}$ *and outputs a bit* $d \in \{\mathtt{true}, \mathtt{false}\}$ *indicating whether* p_i *is a valid proof that* y_i *has been accumulated into* a*. Finally, it outputs a decision* $d \leftarrow \mathsf{Verf}(\mathsf{parm}, a, y_i, p_i)$

The security of the hash function has already been discussed in [2].

Indistinguishability of *Barić* and *Pfitzmann*'s Construction. This section proves that the accumulator by *Barić* and *Pfitzmann* accumulator [2] is indistinguishable following our definition. We give some additional proofs prior to giving the main theorem to increase readability.

Theorem 2 (The probability that $\mathcal{H}$ outputs a prime r, such that r is not coprime to $\varphi(n) = 4p'q'$ is negligible). *The probability that the random oracle* $\mathcal{H}$ *outputs a prime* r *such that* $\gcd(\varphi(n), r) \neq 1$ *is negligible in the security parameter* λ*.*

Proof. Assuming that $\mathcal{H} : \{0,1\}^* \to \mathbb{P}_n$ is a random oracle always returning uniformally distributed odd prime numbers $2 < q_i \leq n$, i.e., $q_i \in \mathbb{P}_n$, we can derive that the probability that it returns a r, such that $r \mid \varphi(n) = 4p'q'$ is negligible. Obviously, the only primes dividing $\varphi(n)$ are $\{2, p', q'\}$, as every other divisor must be a multiple of one of the elements contained in $\{2, p', q'\}$. As only p' and q' are members of $\mathbb{P}_n$, we have exactly 2 primes not fulfilling our definition. This is obviously negligible in λ.

Hence, we can assume that $\mathcal{H}$ only outputs primes which are coprime to $\varphi(n)$:

$$\mathcal{H} : \{0,1\}^* \to (\mathbb{P}_n \setminus \{a \mid \gcd(a, \varphi(n)) \neq 1\}) \tag{1}$$

Theorem 3. *If* $\gcd(u, \varphi(n)) = 1$*,* $f_a : (\mathbb{Z}/n\mathbb{Z}) \to (\mathbb{Z}/n\mathbb{Z}), a \mapsto a^u \bmod n$ *is bijective.*

Proof. We prove this theorem by showing that the kernel of f_a is trivial, i.e., $\operatorname{kern}(f_a) = \{1\}$. It is obvious that f_a describes a group homomorphism. Let $f_a(x) = 1$, i.e., $a^u = 1$. It follows that $\operatorname{ord}_n(a) \mid u$ and following

Lagrange $\text{ord}_n(a) \mid \varphi(n)$. But since $\gcd(u, \varphi(n)) = 1$, $a = 1$ follows. Thus, injectivity is proven. Since domain and range are equal, the function is therefore also bijective. □

Definition 3 (The inverse of f_a). *We define the inverse of f_a as $f_a^{-1} : (\mathbb{Z}/n\mathbb{Z}) \to (\mathbb{Z}/n\mathbb{Z}), a \mapsto \sqrt[u]{a} \bmod n$.*

Theorem 4 (There exists always a uniformally distributed b', for all subsets of $\mathcal{M}$). *For every set $\mathcal{M}$, every starting value b, every subset $\mathcal{M}' \subset \mathcal{M}$, there exists a b', which is also uniformally distributed.*

Proof. Let $\mathcal{H}$ s.t. it outputs only odd primes, coprime to $\varphi(n)$, which as shown in Th. 2 has negligible impact. If $\mathcal{M}' = \mathcal{M}$ we are already done, as $b' = b$ and b is chosen at random. For $\mathcal{M}' \subsetneq \mathcal{M}$ we have to show that for every $\mathcal{M}' = \{y_1, \dots, y_\ell\}$ there exists a b', which is also uniformally distributed, if the original b is. To do so, we let

$$b' = \sqrt[\mathcal{H}(y_\ell)]{\dots \sqrt[\mathcal{H}(y_2)]{\sqrt[\mathcal{H}(y_1)]{a}}} \bmod n$$

As Th. 3 states, the z_i^{th} root is uniquely determined for radicands in $(\mathbb{P}_n \setminus \{a \mid \gcd(a, \varphi(n)) \neq 1\})$. Hence, b' is defined as a composition of isomorphisms. This implies that b' is uniformally distributed as well, if b is. □

Theorem 5. *The accumulator value a is always uniformally distributed, if b is chosen at random.*

Proof. Analogue to Th. 4.

Theorem 6. *The proofs p_i are also uniformally distributed.*

Proof. For given value $z_i = \mathcal{H}(y_i)$, the proof is defined as $p_i = \sqrt[z_i]{a} \bmod n$ for a given accumulator a. Following Th. 4, we can derive that p_i is uniformally distributed as well, if a is uniformally distributed. This is given, since the starting value b is chosen uniformally.

Theorem 7. *The construction by Barić and Pfitzmann is indistinguishable, if b is chosen at random, the hash-function $\mathcal{H}$ is modeled as a random oracle always outputting uniformally distributed odd primes coprime to $\varphi(n)$.*

Proof. The starting value b is uniformally chosen by the oracle given in Fig. 1. Given the accumulator a, the adversary cannot decide how many values have been accumulated, as shown in Th. 5. The proofs are also uniformally distributed, as proven in Th. 6. Following our definition of indistinguishability, no additional information is given to the adversary. This proves the theorem. $\square$

3.3 Achieving Resilience Against Unbounded Adversaries

The last case we have to consider is if $\mathcal{H}$ outputs a prime r, s.t., $r \in \{p', q'\}$. Here, the roots are not uniquely determined and may not be uniformally distributed. Hence, the proofs themselves may give the adversary a non-negligible advantage. The same is true for the accumulator value a, which may therefore not be uniformally distributed. To counter this, we suggest to adjust the digest algorithm to check, if $\mathcal{H}$ outputs $r \in \{p', q'\}$. If so, the corresponding message has to be padded or rehashed using a different modulus. This simple alteration allows for perfect indistinguishability. This overhead only occurs with negligible probability, as proven in Th. 2.

3.4 Removing the Random Oracle

As already shown by *Barić* and *Pfitzmann*, the random oracle can be removed by restricting the input itself to prime numbers [2]. As we have restricted the random oracle to prime numbers, the same idea can be used in our case. Hence, perfect indistinguishability can even be achieved in the standard model. However, the input domain is reduced significantly. All proofs concerning the collision-resistance can be found in [2].

4 Conclusion and Open Questions

In this paper, we introduced the privacy notion of indistinguishability of one-way accumulators. This primitive has been used in many applications, but has not been shown for any existing construction yet. We have shown that the accumulator by *Barić* and *Pfitzmann* [2] is provably indistinguishable, while *Nyberg*'s accumulator [18] does not fulfill our notion. It remains an open question of other accumulators do fulfill our requirements, as we have shown that not all accumulators are indistinguishable. It remains an open question if other accumulators do fulfill the new security definition, e.g., [7,10,13,17].

References

1. J. H. Ahn, D. Boneh, J. Camenisch, S. Hohenberger, A. Shelat, and B. Waters. Computing on authenticated data. Cryptology ePrint Archive, Report 2011/096, 2011. `http://eprint.iacr.org/`.
2. N. Barić and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *EUROCRYPT*, pages 480–494, 1997.
3. J. Benaloh and M. de Mare. One-way accumulators: a decentralized alternative to digital signatures. In *Workshop on the theory and application of cryptographic techniques on Advances in cryptology*, EUROCRYPT '93, pages 274–285, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.
4. B. H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426, 1970.
5. M. Blum, P. Feldman, and S. Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *STOC*, pages 103–112, 1988.
6. A. Buldas, P. Laud, and H. Lipmaa. Accountable certificate management using undeniable attestations. In *ACM CCS*, pages 9–17, 2000.
7. J. Camenisch and A. Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *CRYPTO*, pages 61–76, 2002.
8. S. Canard and A. Jambert. On extended sanitizable signature schemes. In *CT-RSA*, pages 179–194, 2010.
9. N. Fazio and A. Nicolosi. Cryptographic accumulators: Definitions, constructions and applications. Available at http://www-cs.ccny.cuny.edu/ fazio/research.html, 2003.
10. M. T. Goodrich, R. Tamassia, and J. Hasic. An efficient dynamic and distributed cryptographic accumulator. In *ISC*, pages 372–388, 2002.
11. S. Haber and W. S. Stornetta. How to time-stamp a digital document. *Journal of Cryptology*, 3:99–111, 1991.
12. R. Johnson, D. Molnar, D. Song, and D.Wagner. Homomorphic signature schemes. In *Proceedings of the RSA Security Conference - Cryptographers Track*, pages 244–262. Springer, Feb. 2002.
13. H. Kikuchi. Dual rsa accumulators and its application for private revocation check. In *AINA (1)*, pages 237–242, 2006.
14. M. Klonowski and A. Lauks. Extended Sanitizable Signatures. In *ICISC*, pages 343–355, 2006.
15. J. Li, N. Li, and R. Xue. Universal accumulators with efficient nonmembership proofs. In *ACNS*, pages 253–269, 2007.
16. H. Lipmaa. Secure accumulators from euclidean rings without trusted setup. In *ACNS*, pages 224–240, 2012.
17. L. Nguyen. Accumulators from bilinear pairings and applications. In *CT-RSA*, pages 275–292, 2005.
18. K. Nyberg. Fast accumulated hashing. In *FSE*, pages 83–87, 1996.
19. H. C. Pöhls, K. Samelin, H. de Meer, and J. Posegga. Flexible redactable signature schemes for trees - extended security model and construction. In *SECRYPT*, pages 113–125, 2012.

20. H. C. Pöhls, K. Samelin, J. Posegga, and H. de Meer. Transparent mergeable redactable signatures with signer commitment and applications. Technical Report MIP-1206, University of Passau, 8 2012.
21. R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 26(1):96–99, 1983.
22. T. Sander. Efficient accumulators without trapdoor extended abstracts. In *ICICS*, pages 252–262, 1999.
23. P. Wang, H. Wang, and J. Pieprzyk. A new dynamic accumulator for batch updates. In *Proceedings of the 9th international conference on Information and communications security*, ICICS'07, pages 98–112, Berlin, Heidelberg, 2007. Springer-Verlag.