Presentation

*i*SHAKE: Incremental Hashing With SHAKE128 and SHAKE256 for the Zettabyte Era

Danilo Gligoroski[1] and Simona Samardjiska[12]

[1] Department of Telematics, Norwegian University of Science and Technology (NTNU), Trondheim, NORWAY, {danilog, simonas}@item.ntnu.no
[2] "Ss Cyril and Methodius" University, Faculty of Computer Science and Engineering (FINKI), Skopje, MACEDONIA

Abstract. We propose two incremental hash functions based on SHAKE128 and SHAKE256 that have the corresponding security levels of 128 and 256 bits.

1 Introduction

The idea of incremental hashing was introduced by Bellare, Goldreich and Goldwasser in [2]. Later on, they proposed some more efficient designs in [3], compared to the original proposal. In a nutshell, the idea is that if we have already computed the hash value for some document, and this document is modified in one part, then instead of re-computing the hash value for the whole document from scratch, we just need to update the hash value based on the old value, the old changed part and the new changed part.

There have been many proposals for a concrete design of an incremental hash function but the concept, up to now, was not accepted by the industry with a more massive acceptance rate. The reasons for this are mainly two:

1. The security level for the incremental hash functions is detached from the size of the produced hash value, that is usually several thousands of bits long. This is different from the ordinary cryptographic hash functions such as SHA-1, SHA-2, SHA-3, where the size of the hash value correspond to the claimed bit-security level of the hash function.
2. In order to achieve a certain level of security (for example 2^{128} or 2^{256}), the proposed incremental hash functions need to perform expensive modular operations over large prime integers. That makes them one or more orders of magnitude slower than the ordinary cryptographic hash functions.

On the other hand we want to point out to the following:

It is an indisputable fact that our modern civilization has entered the era where the total size of the digital universe (the total size of every digital information that our civilization is producing and copying annually) has surpassed the zettabyte size and is entering the zettabyte communication era with a fast pace. For example "The EMC Digital Universe study - with research and analysis by IDC" [5] reports that in 2013 the size of our digital universe was 4.4 zettabytes, and it projects that *"by 2020 the digital universe - the data we create and copy annually - will reach 44 zettabytes, or 44 trillion gigabytes."* In another report, the Cisco Visual Networking Index [4] predicts that *"Annual global IP traffic will pass the zettabyte threshold by the end of 2015, and will reach 1.4 zettabytes per year by 2017."*

Alongside these reports are the analyses about the exponential drop (in US$/MB) of the cost of computer memory and storage (for example see [1]).

Observation 1. *In the Zettabyte era the trend for reducing the cost of data storage will diminish the importance of Reason 1. for not using incremental hash functions.*

Observation 2. *In the Zettabyte era there will be an increased need for an efficient and secure cryptographic primitive that will perform incremental hashing.*

The motivation for this presentation is based on Observation 1 and Observation 2 and the recent [6] draft NIST proposal for SHA-3 Extendable-Output Functions SHAKE128 and SHAKE256.

2 Mathematical preliminaries

We will use the following adapted definition from [3, Sec. 3.1] for an incremental hash function:

Definition 1.

1. *Let $h : \{0,1\}^b \to \{0,1\}^k$ be a compression function that maps b bits into k bits.*
2. *Let the message M be represented as a concatenation of n blocks, where $n < N$ for some predefined number N which is larger than the number of blocks in any message we plan to hash, i.e., $M = M_1||M_2||\ldots||M_n$.*
3. *The size of each block M_i is determined by the following relation: $|M_i| = b - \lg(N)$.*
4. *For each block M_i, $i = 1, \ldots, n$, prepend a $\lg(N)$-bit binary encoding $\langle i \rangle$ of the block index i to the block content M_i to get an augmented block $\overline{M_i} = \langle i \rangle || M_i$.*
5. *For each $i = 1, \ldots, n$, apply h to $\overline{M_i}$ to get a hash value $y_i = h(\overline{M_i})$.*
6. *Let $(G, \bigodot)$ be a commutative group with operation $\bigodot$ where $G \subseteq \{0,1\}^k$.*
7. *Combine $y_1, \ldots, y_n$ via a combining group operation $\bigodot$ to get the final hash value $y = y_1 \bigodot y_2 \bigodot \ldots \bigodot y_n$.*

Denote the incremental hash function as:

$$y(M) = \mathrm{HASH}^h_{\langle G \rangle}(M_1||M_2||\ldots||M_n) = \bigodot_{i=1}^{n} h(\langle i \rangle || M_i) \tag{1}$$

Initially the authors of [3] concluded that in order to have a cryptographically secure hash construction, the balance problem for the group $(G, \bigodot)$ should be hard, and they concluded that a size of the hash value in the range of 1024 bits would suffice for security levels of 2^{80}. However, later on, Wagner in [7] showed that using a generalized birthday attack, these parameters are breakable, implying that the size of the hash values should be much bigger (for standard security levels, even up to tens of thousands of bits). Basically those findings killed the attractiveness of the concept of incremental hashing.

Proposition 1. *Let $\mathrm{HASH}^h_{\langle G \rangle}$ be an incremental hash function defined by Definition 1. For any $Y \in \{0,1\}^k$ the complexity of finding a preimage message $M = M_1||M_2||\ldots||M_K$ of length $K \leqslant N$ blocks such that $Y = \mathrm{HASH}^h_{\langle G \rangle}(M)$ is:*

$$\min_{K \leqslant N} O(K \cdot 2^{\frac{k}{1+\lg\lfloor K \rfloor}}) \tag{2}$$

If the length of the messages is not restricted, then the minimum in equation (2) is achieved for messages of $K = 2^{\sqrt{k}-1}$ blocks.

Proof. Just adopt the notation from [7, Sec. 2, *Summary*] to match the notation of variables in Definition 1.

Observation 3. *For the compression functions $h : \{0,1\}^b \to \{0,1\}^k$ where k is multiple of 64 bits i.e. $k = 64 \cdot L$, on modern 64-bit CPUs, instead of modular operations with k bit prime numbers in the group $(G, \bigodot)$, much more efficient operations would be word wise operations of addition in the group $((\mathbb{Z}_{64})^L, \boxplus_{64})$.*

3 Definition of *i*SHAKE

Recently, NIST has proposed the *DRAFT SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions* [6]. In that draft there are definitions for two Extendable-Output Functions named SHAKE128 and SHAKE256. We just briefly mention their definition:

$$\text{SHAKE128}(M, d) = \text{RawSHAKE128}(M||11, d),$$

where

$$\text{RawSHAKE128}(M, d) = \textsc{Keccak}[256](M||11, d),$$

and

$$\text{SHAKE256}(M, d) = \text{RawSHAKE256}(M||11, d),$$

where

$$\text{RawSHAKE256}(M, d) = \textsc{Keccak}[512](M||11, d).$$

3.1 *i*SHAKE128

We give the following proposal for an incremental hash function *i*SHAKE128 with a security of 128 bits:

Definition 2.

1. *Let $h : \{0,1\}^{1344} \to \{0,1\}^{2688}$ be defined as the function $h(m) = \text{SHAKE128}(m, 2688)$ where $|m| = 1344$.*
2. *Let the message $M = M_1||M_2||\ldots||M_n$ be represented as a concatenation of n blocks, where $n < N$, and $N = 2^{25}$ is the largest number of blocks in any message we plan to hash.*
3. *The size of each block M_i in bits is determined by the following relation: $|M_i| = 1344 - 64 = 1280$.*
4. *For each block M_i, $i = 1, \ldots, n$, prepend a 64-bit binary encoding $\langle i \rangle$ of the block index i to the block content M_i to get an augmented block $\overline{M_i} = \langle i \rangle || M_i$.*
5. *For each $i = 1, \ldots, n$, apply h to $\overline{M_i}$ to get a hash value $y_i = h(\overline{M_i}) = \text{SHAKE128}(\overline{M_i}, 2688)$.*
6. *Let $((\mathbb{Z}_{64})^{42}, \boxplus_{64})$ be a commutative group with the operation $\boxplus_{64}$ that represents a 64-bit word wise addition of 42 words.*
7. *Combine $y_1, \ldots, y_n$ via a combining group operation $\boxplus_{64}$ to get the final hash value*

$$y = y_1 \boxplus_{64} y_2 \boxplus_{64} \cdots \boxplus_{64} y_n.$$

Denote the incremental hash function *i*SHAKE128 *as:*

$$i\text{SHAKE128}(M) = \bigboxplus_{i=1}^{N}{}_{64} \; SHAKE128(\overline{M_i}, 2688). \tag{3}$$

Using appropriate values in expression (2) we have the following:

Corollary 1. *Let $b = 1280$, $k = 2688$. and let the maximal allowed number of blocks be $N = 2^{25}$. Then*

$$\min_{K \quad N} O(K \cdot 2^{\frac{k}{1+\lg[K]}}) = 2^{128.385} \tag{4}$$

3.2 *i*SHAKE256

We give the following proposal for an incremental hash function *i*SHAKE256 with a security of 256 bits:

Definition 3.

1. *Let $h : \{0,1\}^{1088} \to \{0,1\}^{6528}$ be defined as the function $h(m) = \text{SHAKE256}(m, 6528)$ where $|m| = 1088$.*
2. *Let the message $M = M_1||M_2||\ldots||M_n$ be represented as a concatenation of n blocks, where $n < N$, and $N = 2^{28}$ is the largest number of blocks in any message we plan to hash.*
3. *The size of each block M_i in bits is determined by the following relation: $|M_i| = 1088 - 64 = 1024$.*
4. *For each block M_i, $i = 1, \ldots, n$, prepend a 64-bit binary encoding $\langle i \rangle$ of the block index i to the block content M_i to get an augmented block $\overline{M_i} = \langle i \rangle || M_i$.*
5. *For each $i = 1, \ldots, n$, apply h to $\overline{M_i}$ to get a hash value $y_i = h(\overline{M_i}) = \text{SHAKE256}(\overline{M_i}, 6528)$.*
6. *Let $((\mathbb{Z}_{64})^{102}, \boxplus_{64})$ be a commutative group with the operation $\boxplus_{64}$ that represents a 64-bit word wise addition of 102 words.*
7. *Combine $y_1, \ldots, y_n$ via a combining group operation $\boxplus_{64}$ to get the final hash value*

$$y = y_1 \boxplus_{64} y_2 \boxplus_{64} \cdots \boxplus_{64} y_n.$$

Denote the incremental hash function *i*SHAKE256 *as:*

$$i\text{SHAKE256}(M) = \boxplus_{64,\, i=1}^{N} SHAKE256(\overline{M_i}, 6528). \tag{5}$$

Using appropriate values in expression (2) we have the following:

Corollary 2. *Let $b = 1024$, $k = 6528$. and let the maximal allowed number of blocks be $N = 2^{28}$. Then*

$$\min_{K \quad N} O(K \cdot 2^{\frac{k}{1+\lg[K]}}) = 2^{253.103} \tag{6}$$

4 Conclusions

Based on the observation that in the Zettabyte era the need for incremental hashing will be very big, and based on the property that the functions SHAKE128 and SHAKE256 (recently proposed by NIST in the SHA-3 Draft FIPS 202 proposal [6]) can output data with arbitrary long length, we have proposed two incremental hash functions *i*SHAKE128 and *i*SHAKE256 with corresponding security levels of 128 and 256 bits.

References

1. Historical cost of computer memory and storage. *hblok.net • Freedom, Electronics and Tech*, February 2013. http://hblok.net/blog/storage/.
2. Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. Incremental cryptography: The case of hashing and signing. In Yvo Desmedt, editor, *CRYPTO*, volume 839 of *Lecture Notes in Computer Science*, pages 216–233. Springer, 1994.
3. Mihir Bellare and Daniele Micciancio. A new paradigm for collision-free hashing: Incrementality at reduced cost. In Walter Fumy, editor, *EUROCRYPT*, volume 1233 of *Lecture Notes in Computer Science*, pages 163–192. Springer, 1997.
4. Cisco. Cisco visual networking index: Forecast and methodology, 2012-2017. *White Paper*, May 2013. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/VNI_Hyperconnectivity_WP.pdf.
5. EMC. The emc digital universe study with research and analysis by idc. *Open Report*, April 2014. http://www.emc.com/leadership/digital-universe/index.htm?pid=home-dig-uni-090414.
6. NIST. Draft sha-3 standard: Permutation-based hash and extendable-output functions. *FIPS 202*, April 2014. http://csrc.nist.gov/publications/PubsDrafts.html#FIPS-202.
7. David Wagner. A generalized birthday problem. In Moti Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 288–303. Springer, 2002.