# A New Proof for the Security of the Keyed Sponge Construction in the Ideal Compression Function Model

**Tuan Anh Nguyen, Bui Cuong Nguyen**

*Abstract*— **In this paper, we present a new proof for the security of keyed Sponge. Our method is built on the previous result about the indistinguishability of the Sponge construction. Following this approach, we can see the strong relationship between the security of keyed Sponge and its original version.**

*Tóm tắt*— **Trong bài báo này, chúng tôi đưa ra một chứng minh mới cho độ an toàn của cấu trúc Sponge có khóa. Phương pháp của chúng tôi sử dụng kết quả trước đó về tính không phân biệt được của cấu trúc Sponge. Theo cách tiếp cận này, chúng ta có thể thấy mối liên hệ chặt chẽ về độ an toàn của cấu trúc Sponge có khóa và phiên bản nguyên thủy của nó.**

***Keywords***— ***Sponge construction, keyed Sponge construction, ideal compression function model, PRF security.***

***Từ khóa***— ***Cấu trúc Sponge, Cấu trúc Sponge có khóa, mô hình hàm nén lý tưởng, độ an toàn PRF.***

## I. INTRODUCTION

Since its introduction, the Sponge construction [1] has been attracting research attention in the cryptography community. It is the fundamental of the SHA-3 standard Keccak [2]. In the security model of Maurer [3], Bertoni at el. [4] proved that the advantage in differentiating the sponge construction from a random oracle is upper bounded by $O(N^2/2^c)$, with N the number of calls to the underlying function *f* and *c* the capacity.

Inspired by the introduction of keys into hash functions as before and the beautiful theoretical results of the sponge construction, designers presented the keyed version for it: Sponge(K||M), we denote by KeyedSponge. This version was proposed to build a wide spectrum of symmetric-key primitive: Reseedable pseudorandom number generator [5], pseudorandom function and message authentication codes (PRFs/MAC) [6, 7], extendable-output functions [8] and authenticated encryption modes [9]. Because of its wide application, the security of the KeyedSponge construction has been evaluated in many documents and now it still attracts interest from the cryptography community.

*Previous results*

The KeyedSponge construction has been previously studied by two main approaches: the H-coefficient technique and the Sponge graph. For the first method there are two outstanding papers: Elena Andreeva [10] and Peter Gaži [11]. Specifically, Elena Andreeva evaluated the security of keyed Sponge when the key length k is a multiple of r. The distinguishing advantage of the KeyedSponge construction under any adversary who makes at most $q$ construction queries, and at most $Q$ primitive queries is upper bounded by

$$\frac{N_0^2+2\mu Q}{2^c} + \lambda(Q) + \frac{2\left(\frac{k}{r}\right)Q}{2^b},$$

where $N_0$ is the number of fresh calls to the underlying function $f$ when he makes q construction queries (here we let *f(x)* is not fresh if it has already been made due to a prior query to the construction) and $\mu$ is the total maximum multiplicity (see [10]) and

$$\lambda(Q) \le \begin{cases} \dfrac{Q}{2^k} & \text{if } k = r \\ \min\left\{\dfrac{Q^2}{2^{c+1}} + \dfrac{Q}{2^k}, \dfrac{1}{2^b} + \dfrac{Q}{2^{\left(\frac{1}{2} - \frac{\log_2(3b)}{2r} - \frac{1}{r}\right)k}}\right\} & \text{otherwise.} \end{cases}$$

This manuscript is received July 10, 2019. It is commented on August 16, 2019 and is accepted on August 23, 2019 by the first reviewer. It is commented on September 30, 2019 and is accepted on October 6, 2019 by the second reviewer.

The distinguishing advantage of the KeyedSponge was estimated by Peter Gaži has upper bound:

$$O\left(\frac{q^2+qQ+lq}{2^{b-z}}\right)+\frac{\binom{k}{r}N_1}{2^b}$$
$$+\min\left\{\frac{N_1}{2^{\left(\frac{1}{2}-\frac{\log_2(3b)}{2r}-\frac{1}{r}\right)k}},\frac{N_1}{2^k}+\frac{N_1^2}{2^c}\right\},$$

where $l$ is the maximum number of blocks in each construction query, z (z<r) is the fixed length of output and $N_1=ql+Q$.

For the second method, Guido Bertoni [6] uses the Sponge graph to directly evaluate the security of the KeyedSponge construction. The distinguishing advantage is upper bounded by:

$$O\left(\frac{N_0^2}{2^c}+\frac{N_0Q}{2^c}+\frac{Q}{2^k}\right).$$

In our knowledge, it is the best result ever.

**Our contributions**. In this paper, we give a new proof for the security of the KeyedSponge construction according to the second approach. We evaluate it by using the security of the Sponge construction. Our bound are:

$$\frac{Q}{2^k}+\frac{Q^2}{2^{c+1}}+\frac{N^2}{2^{c+1}},$$

where $N$ is the number of fresh calls to the underlying function $f$ when adversary makes both construction queries and primitive queries. Although this result is not as tight as Guido Bertoni's, it shows a close relationship between the security of the KeyedSponge and Sponge construction.

**Organization of the paper**. After the introduction, in Section 2 we present some preliminaries. In Section 3, we recall the security of the Sponge construction. In Section 4, we evaluate the indistinguishability of the KeyedSponge construction. Finally, some conclusions are given in Section 5.

## II. PRELIMINARIES

We denote the set of all finite strings of arbitrary length by $\mathbb{Z}_2^*$ , the set of infinite-length bit strings is denoted by $\mathbb{Z}_2^\infty$, the concatenation of two strings $x$ and $y$ is denoted as $x||y$. We let $\mathrm{left}_z(x)$ denote the $z$ leftmost bits of $x$. We denote the length in bits of a message $x$ by $|x|$. The number of r-bit blocks of $x$ is denoted by $|x|_r$.

*Random oracle.*

Let $\mathcal{RO}:\mathbb{Z}_2^* \rightarrow \mathbb{Z}_2^\infty$ be a random oracle which takes inputs of arbitrary but finite length and returns random infinite strings, where each output bit is selected uniformly and independently for every input $M$.

We denote a call to $\mathcal{RO}$ where the output is truncated to its $z$ first bits by $Z=\mathcal{RO}(M,z)$.

*Sponge construction.*

The Sponge construction determines the Sponge function $\mathrm{Sponge}[f,pad,r]$with domain $\mathbb{Z}_2^*$ and codomain $\mathbb{Z}_2^\infty$ by using a fixed length underlying function $f$ (transformation or permutation), a sponge-compliant padding rule (see [12], Definition 1) and a parameter bitrate $r$. It can return a finite-length output by taking its $z$ first bits.

The underlying function $f$ operates on a fix number of bit, the width $b$. The state of the Sponge construction is $b$ bit. First, we initialize the b bits of the Sponge state to zero. The input is padded and cut into $r$-bits blocks. Then it is processed by an absorbing phase followed by a squeezing phase (see Fig.1). In these phase, the first $r$ bits and the remaining b-r bits of the state are treated differently. We let the first $r$ bits of the state as the outer part $\bar{s}$, and the last $c=b-r$ bits as its inner part $\hat{s}$ ($c$ is called the capacity). The Sponge construction is presented in Fig.1.

The Sponge construction gets as input a message $M$, a natural number $z$, and it outputs a string $Z\in\{0,1\}^z$:

$$\mathrm{Sponge}^f(M,z)=\mathrm{Sponge}[f,pad,r](M,z),$$

where $\mathrm{Sponge}[f,pad,r]$ is defined in Algorithm 1.

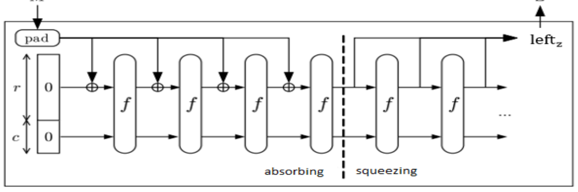


Fig.1. The Sponge construction

**Algorithm 1. Sponge$[f, pad, r]$**

```
Require: $r < b$
Interface: $Z = sponge(M, z)$ with $M \in \mathbb{Z}_2^*$, $z > 0$ and $Z \in \mathbb{Z}_2^z$.
    $P = M||pad[r](|M|)$
    $s = 0^b$
    for $i = 0$ to $|P|_r - 1$ do
        $s = s \oplus (P_i||0^{b-r})$
        $s = f(s)$
    end for
    $T = \lfloor s \rfloor_r$
    while $|T| < z$ do
        $s = f(s)$
        $T = T||\text{left}_z(s)$
    end while
    return $Z = \text{left}_z(T)$
```

*The absorb function.* The absorb function. The function $\text{absorb}(\cdot)$ takes as input a message $P$ with $|P|$ multiple of $r$ and it outputs the state $s \in \mathbb{Z}_2^b$

```
    $s = 0^b$
    For $i = 0$ to $|P|_r - 1$ do
      $s = s \oplus (P_i||0^{b-r})$
      $s = f(s)$
    End for
    Return $s$
```

*Path.* $P$ is a path to the state $s$ if $s = \text{absorb}(P)$.

The Sponge graph. The Sponge function can be represented by the Sponge graph with $2^b = 2^{r+c}$ nodes and $2^b$ edges. The nodes are the state values and for every couple $(s, t)$ with $t = f(s)$ there is a directed edge from $s$ to $t$. From each node, there is only one outgoing edge. If $f$ is a permutation, in each node, there is only one incoming edge.

These nodes can be divided by the value of the inner state. We call the set of all nodes that same inner state by a *supernode*. Edges between nodes are therefore also edges between supernodes. There are $2^c$ supernodes, one supernode corresponds to one inner state value. Each supernode has $2^r$ nodes which defined by the outer part $\bar{s}$ of their state.

*The keyed Sponge.*

The keyed Sponge is denoted as KeyedSponge which gets as input a key $K \in \{0,1\}^k$, a message $M \in \{0,1\}^*$, and a natural number $z$. Then, it returns a string $Z \in \{0,1\}^z$:

$$\text{KeyedSponge}_K^f(M, z) \coloneqq \text{Sponge}^f(K\|M, z) = Z.$$

*The security model.* An adversary $\mathcal{A}$ is an algorithm that is given query access to one or more oracle $\mathcal{O}$, denoted $\mathcal{A}^{\mathcal{O}}$. Let $\mathcal{A}^{\mathcal{O}} = 1$ be the event that $\mathcal{A}$ returns 1 after $\mathcal{A}'s$ interaction. In the security model of this paper, we consider the KeyedSponge construction is built on a random permutation $f$. The PRF-security of Keyed Sponge is the indistinguishability between the real world and the ideal world. Let $\mathcal{O}_R = \text{KeyedSponge}_K^f(\cdot)$ with $K \xleftarrow{\$} \{0,1\}^k$ be the oracle in the real world, and $\mathcal{O}_I = \mathcal{RO}$ be the oracle in the ideal world. The indistinguishability considers the case where an adversary $\mathcal{A}$ has query access to $(\mathcal{O}_R, f, f^{-1})$ in the real world and $(\mathcal{O}_I, f, f^{-1})$ in the ideal world, and after $\mathcal{A}$'s interaction, it outputs a result $y \in \{0,1\}$. We call queries to $\mathcal{O}_R/\mathcal{O}_I$ "construction queries" and queries to $(f, f^{-1})$ "primitive queries". We define the advantage function as

$$\text{Adv}_{\text{KeyedSponge}}^{\text{prf}}(\mathcal{A}) \coloneqq \left|\Pr\left[\mathcal{A}^{\mathcal{O}_R, f, f^{-1}} = 1\right] - \Pr\left[\mathcal{A}^{\mathcal{O}_I, f, f^{-1}} = 1\right]\right|.$$

We denote $q$ and $Q$ respectively as the number of construction and primitive queries.

## III. MODEL THE SECURITY OF THE SPONGE CONSTRUCION

In [12], the authors evaluated the indistinguishability of the Sponge construction when an adversary was able to query the Sponge construction and the underlying function $f$. Then, in the ideal world besides the oracle $\mathcal{RO}$, [12] built another component with the same interface as $f$. For the components to be hard to distinguish, it should simulate the behavior of a random permutation of the same width as $f$. For this reason it is called a *simulator*, denoted $\mathcal{P}$.

In addition, we have an additional constraint when an adversary queries to the real world (the Sponge construction and its underlying function $f$), he the adversary can verify whether the responses to the queries are *Sponge-consistent* or not. The Sponge-consistent means that: the result for a construction query will be the same as the answer when we simulate the Sponge construction by querying directly to $f$. In particular, let $M$ be a message, if we make a construction query $(M, z)$ then we receive $Z$. In the other hand, the message $M$ after padding will have the form: $P = M\|\text{pad}[r](|M|)$. Then, we can compute the $j$th block of $Z$ by querying to the function $f$: $Z_j = \overline{\text{absorb}}(P\|0^{rj})$. For the ideal world to be hard to distinguish from the real world it shall also behave sponge-consistent. For that reason, the simulator may have query access to the random oracle $\mathcal{RO}$ for satisfying sponge-consistency.

Before going into the specific definition of the simulator $\mathcal{P}$ we recall some symbols. In the Sponge graph, we define the set of *rooted* supernodes $\mathcal{R}$ as the subset of $\mathbb{Z}_2^c$ containing $0^c$ and all the supernodes accessible from it through the supernode graph. We say that a node $s = (\bar{s}, \hat{s})$ is rooted if $\hat{s} \in \mathcal{R}$. Let $O$ be the set of supernodes with an outgoing edge.

**Algorithm 2 (see algorithm 9, [12])**: The simulator $\mathcal{P}$

```
Interface $\mathcal{P}^1$, taking node $s$ as input.
If node $s$ has no outgoing edge then
  If node $s$ is rooted AND $\mathcal{R} \cup O \neq \mathbb{Z}_2^c$ (no saturation) then
    Construct path to $t$: find path to $s$, append $\bar{s}$ and call the result $P$
    Write $P$ as $P = P'0^{rj}$ where $P'$ does not end with $0^r$
    if $P'$ can be unpadded into $M$ then
      Assign to $\bar{t}$ the value $Z_j$ with $Z = \mathcal{RO}(M, jr)$
    Else
      Choose $\bar{t}$ randomly and uniformly
    end if
    Choose $\hat{t}$ randomly and uniformly form $\mathbb{Z}_2^c\backslash(\mathcal{R} \cup O)$ and such that $\bar{t}\|\hat{t}$ has no incoming edge yet
    Let $t = \bar{t}\|\hat{t}$
  Else
    Choose $t$ randomly and uniformly from all nodes that have no incoming edge yet
  End if
  Add an edge from $s$ to $t$
End if
Return the node $t$ at the end of the outgoing edge from $s$

Interface $\mathcal{P}^{-1}$, taking node $s$ as input
If node $s$ has no incoming edge then
  Choose $\bar{t}$ randomly and uniformly
  Choose $\hat{t}$ randomly and uniformly from $\mathbb{Z}_2^c\backslash\mathcal{R}$ and such that $\bar{t}\|\hat{t}$ has no outgoing edge yet
  Let $t = \bar{t}\|\hat{t}$
  Add an edge from $t$ to $s$
End if
Return the node $t$ at the beginning of the incoming edge into $s$
```

Then, [12] considered the advantage of an adversary when distinguishing the two following world.

*Real world.* It contains the Sponge construction and the underlying random permutation $f$. The adversary can make queries to the Sponge construction, the permutation $f$ and $f^{-1}$.

*Ideal world.* It contains the random oracle $\mathcal{RO}$ and the simulator $\mathcal{P}$. The adversary can make queries to $\mathcal{P}$ and $\mathcal{P}^{-1}$.

We define the $\mathcal{RO}$ differentiating advantage as

$$\text{Adv}_{\text{Sponge}}^{\text{ind}}(\mathcal{A}) := \left|\Pr\left[\mathcal{A}^{\text{Sponge}^f, f, f^{-1}}\right] - \Pr\left[\mathcal{A}^{\mathcal{RO}, \mathcal{P}, \mathcal{P}^{-1}} = 1\right]\right|$$

*Theorem 1 (Theorem 9, [12]). The $\mathcal{RO}$ differentiating advantage of the Sponge construction calling the random permutation $f$ is upper bound by $1 - \prod_{i=0}^{N-1}\left(\frac{1-\frac{i+1}{2^c}}{1-\frac{i}{2^{r+c}}}\right)$ with $N$ is the number of fresh calls to $f$.*

In the paper [12], $N$ was denoted by the cost of the queries. However, in our security model, it is the number of fresh calls to $f$.

## IV. OUR EVALUATION FOR THE SECURITY OF THE KEYSPONGE

In this section, we will evaluate the PRF-security of the KeyedSponge construction by using Theorem 1 which states about the $\mathcal{RO}$ differentiating advantage of the Sponge construction.

*Proposition 1. Let $\mathcal{A}$ be an adversary making at most $q$ construction queries and at* *most $Q$ primitive queries. Then, the PRF-security of the KeyedSponge construction calling the random permutation $f$ is upper bound by:*

$$\mathrm{Adv}^{\mathrm{prf}}_{\mathcal{F}_K}(\mathcal{A}) \leq \frac{Q}{2^k} + \frac{Q^2}{2^{c+1}} + \frac{N^2}{2^{c+1}},$$

*where $N$ is the number of fresh calls to $f$ in both query types.*

*Proof.* This result will be proved by reduction. It means that we will prove the security of the KeyedSponge construction through the security of the Sponge construction by constructing an adversary $\mathcal{B}$ which against the Sponge construction from the adversary $\mathcal{A}$. First, $\mathcal{B}$ chooses a key $K$ randomly and uniformly from $\{0,1\}^k$, and it remains the same throughout the process ($\mathcal{A}$ does no $K$). If $\mathcal{A}$ makes a construction query $(M, z)$ then $\mathcal{B}$ makes the construction query $(K\|M, z)$ to its oracle. If $\mathcal{A}$ makes the primitive query $X$ then $\mathcal{B}$ also makes the primitive query $X$. The adversary $\mathcal{B}$ returns to $\mathcal{A}$ the value that he receives. Final, after making the queries, if $\mathcal{A}$ returns a bit $y \in \{0,1\}$ then $\mathcal{B}$ also returns the bit $y$. This means that, if $\mathcal{A}$ thinks that the oracles, which he interacted, is the real world, then B also thinks that the oracles, which he interacted, is the real world, and vice versa.

Let $\Pr[\mathcal{A}^{\mathrm{real}} \Rightarrow 1]$ or $\Pr[\mathcal{A}^{\mathrm{ideal}} \Rightarrow 1]$ be the probability that $\mathcal{A}$ returns 1 when he is used as a subroutine of $\mathcal{B}$, where the oracle that $\mathcal{B}$ is interacted is real or ideal world. We have:

$$\Pr[\mathcal{A}^{\mathrm{real}} \Rightarrow 1] = \Pr\left[\mathcal{B}^{\mathrm{Sponge}^f, f, f^{-1}} \Rightarrow 1\right]$$

and

$$\Pr[\mathcal{A}^{\mathrm{ideal}} \Rightarrow 1] = \Pr[\mathcal{B}^{\mathcal{RO}, \mathcal{P}, \mathcal{P}^{-1}} \Rightarrow 1].$$

In the other hand, in the real world, the result that $\mathcal{A}$ receives for the construction query $(M, z)$ is $\mathrm{Sponge}^f(K\|M, z) = \mathrm{KeyedSponge}^f_K(M, z)$. This means that the view that $\mathcal{A}$ runs as a subroutine of $\mathcal{B}$ same the view that $\mathcal{A}$ runs against the KeyedSponge construction. We have,

$$\Pr[\mathcal{A}^{\mathrm{real}} \Rightarrow 1] = \Pr[\mathcal{A}^{\mathcal{O}_R, f, f^{-1}} = 1].$$

Now, we consider when $\mathcal{B}$ accesses into the ideal model. The result that $\mathcal{A}$ receives for the construction query $(M, z)$ is $\mathcal{RO}(K\|M, z)$. It is a $z$-bit randomly and uniformly string. This is identical when $\mathcal{A}$ runs against the KeyedSponge construction. For primitive queries $X$, the result that $\mathcal{A}$ receives from $\mathcal{B}$ is $\mathcal{P}(X)$ or $\mathcal{P}^{-1}(X)$. So, we need evaluate the difference between a random permutation $f$ and the simulator $\mathcal{P}$.

*Lemma 1 (Lemma 5, [12]). The advantage of an adversary in distinguishing $f$ and $\mathcal{P}$ with the number of queries $Q < 2^c$ is upper bounded by:*

$$1 - \prod_{i=0}^{Q-1} \left( \frac{1 - \frac{i+1}{2^c}}{1 - \frac{i}{2^{r+c}}} \right).$$

(The proof of this lemma is presented in [12]).

When $Q$ is significantly smaller than $2^c$, the above bound can be simplified to $Q^2/2^{c+1}$. Indeed, using the $1 - x \approx e^{-x}$ approximation, we have

$$\log \prod_{i=0}^{Q-1} \left( \frac{1 - \frac{i+1}{2^c}}{1 - \frac{i}{2^{r+c}}} \right) = \sum_{i=0}^{Q-1} \left[ \log\left(1 - \frac{i+1}{2^c}\right) - \log\left(1 - \frac{i}{2^{r+c}}\right) \right]$$

$$\approx \sum_{i=0}^{Q-1} \left[ -\frac{i+1}{2^c} - \left(-\frac{i}{2^{r+c}}\right) \right]$$

$$= \sum_{i=0}^{Q-1} \left[ -\frac{i+1}{2^c} + \frac{i}{2^{r+c}} \right]$$

$$= -\frac{Q(Q+1)}{2^{c+1}} + \frac{(Q-1)Q}{2^{r+c+1}}.$$

Then

$$\prod_{i=0}^{Q-1}\left(\frac{1-\frac{i+1}{2^c}}{1-\frac{i}{2^{r+c}}}\right) \approx e^{-\frac{Q(Q+1)}{2^{c+1}}+\frac{(Q-1)Q}{2^{r+c+1}}}.$$

So, we have

$$1-\prod_{i=0}^{Q-1}\left(\frac{1-\frac{i+1}{2^c}}{1-\frac{i}{2^{r+c}}}\right) \approx 1-e^{-\frac{Q(Q+1)}{2^{c+1}}+\frac{(Q-1)Q}{2^{r+c+1}}}$$

$$\approx \frac{Q(Q+1)}{2^{c+1}}-\frac{(Q-1)Q}{2^{r+c+1}} = O\left(\frac{Q^2}{2^{c+1}}\right).$$

Continue with the case that $\mathcal{B}$ accesses into the ideal model. We can see that, if $\mathcal{A}$ runs against the KeyedSponge construction then $\mathcal{RO}(M,z)$ and $f(X)$ (or $f^{-1}(X)$) do not have any relationship. When $\mathcal{A}$ runs as a subroutine of $\mathcal{B}$, the values that he receives for the construction queries and the primitive queries are $\mathcal{RO}(K\|M,z)$ and $\mathcal{P}(X)$ (or $\mathcal{P}^{-1}(X)$). Note that the simulator $\mathcal{P}$ satisfies Sponge-consistent: the result for a construction query to $\mathcal{RO}$ will be the same as the answer when we simulate by querying directly to $\mathcal{P}$. However, this only happens when the adversary $\mathcal{A}$ guesses the key $K$ among primitive queries. The probability of it is $Q/2^k$.

Thus, in the case that $\mathcal{B}$ accesses into the ideal model, we have

$$\Pr\left[\mathcal{B}^{\mathcal{RO},\mathcal{P},\mathcal{P}^{-1}} \Rightarrow 1\right]-\Pr\left[\mathcal{A}^{\mathcal{O}_I,f,f^{-1}}=1\right] \le \frac{Q}{2^k}+\frac{Q^2}{2^{c+1}}.$$

From above arguments we have

$$\begin{aligned}\mathrm{Adv}^{\mathrm{prf}}_{\mathrm{KeySponge}}(\mathcal{A}) &= \Pr\left[\mathcal{A}^{\mathcal{O}_R,f,f^{-1}}=1\right] - \Pr\left[\mathcal{A}^{\mathcal{O}_I,f,f^{-1}}=1\right]\\ &\le \Pr\left[\mathcal{B}^{\mathrm{Sponge}^f,f,f^{-1}} \Rightarrow 1\right] - \Pr\left[\mathcal{B}^{\mathcal{RO},\mathcal{P},\mathcal{P}^{-1}} \Rightarrow 1\right]+\frac{Q}{2^k}+\frac{Q^2}{2^{c+1}}\\ &\le \mathrm{Adv}^{\mathrm{ind}}_{\mathrm{Sponge}}(\mathcal{B})+\frac{Q}{2^k}+\frac{Q^2}{2^{c+1}}\\ &\le \frac{Q}{2^k}+\frac{Q^2}{2^{c+1}}+\frac{N^2}{2^{c+1}},\end{aligned}$$

where $N$ is the number of fresh call to $f$ when $\mathcal{B}$ making the construction and primitive queries. However, $N$ is also the number of fresh call to $f$ when $\mathcal{A}$ making the construction and primitive queries. Indeed, for the construction query $M$ of $\mathcal{A}$ or $K\|M$ of $\mathcal{B}$, the oracle of $\mathcal{A}$ and $\mathcal{B}$ both compute $\mathrm{Sponge}^f(K\|M,z)$; for the primitive query $X$, both of them compute $f(X)$.■

## V. CONCLUSION

In this paper, the security of the KeyedSponge construction has been evaluated by a new way. We have proved the security of the KeyedSponge construction based on the security of the Sponge construction by reduction method. However, our indirect proof lead to the security bound is not as good as the result in the direct way of Guido Bertoni. Therefore, closing this gap will still be an open problem in the future.

## REFERENCES

[1]. Bertoni, G., et al. Sponge functions. in ECRYPT hash workshop. 2007. Citeseer.

[2]. Bertoni, G., et al., Keccak specifications. Submission to NIST (round 2), 2009: p. 320-337.

[3]. Maurer, U., R. Renner, and C. Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. in Theory of cryptography conference. 2004. Springer.

[4]. Bertoni, G., et al. On the indifferentiability of the sponge construction. in Annual International Conference on the Theory and Applications of Cryptographic Techniques. 2008. Springer.

[5]. Bertoni, G., et al. Sponge-based pseudo-random number generators. in International Workshop on Cryptographic Hardware and Embedded Systems. 2010. Springer.

[6]. Bertoni, G., et al. On the security of the keyed sponge construction. in Symmetric Key Encryption Workshop. 2011.

[7]. Bertoni, G., et al., Permutation-based encryption, authentication and authenticated encryption. Directions in Authenticated Ciphers, 2012.

[8]. Dworkin, M.J., SHA-3 standard: Permutation-based hash and extendable-output functions. 2015.

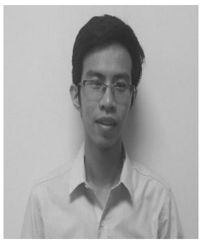
[9]. Bertoni, G., et al. Duplexing the sponge: single-pass authenticated encryption and other applications. in International Workshop on Selected Areas in Cryptography. 2011. Springer.

[10]. Andreeva, E., et al. Security of keyed sponge constructions using a modular proof approach. in International Workshop on Fast Software Encryption. 2015. Springer.

[11]. Gaži, P., K. Pietrzak, and S. Tessaro. The exact PRF security of truncation: tight bounds for keyed sponges and truncated CBC. in Annual Cryptology Conference. 2015. Springer.

[12]. Guido, B., et al., Cryptographic sponge functions. 2011.

## ABOUT THE AUTHORS



**B.S. Tuan Anh Nguyen**

Email: tuananhnghixuan@gmail. com

The Workplace: Institute of Cryptography Science and Technology, Government Information Security Committee.

The education process: Graduated in Mathematic, VNU University of Science, 2016.

Subjects: block cipher, hash function, message authentication code, tweakable block cipher

**PhD. Bui Cuong Nguyen**

Email:nguyenbuicuong@gmail. com

The Workplace: Institute of Cryptography Science and Technology, Government Information Security Committee.

The education process: Graduated in Mathematic, Hanoi National University of Education, 2004. Graduated Master in Mathematics, VNU University of Science, 2007. Graduated PhD in Cryptography, Academy of military science and technology, 2018.

Subjects: block cipher, hash function, message authentication code, tweakable block cipher