Bachelor's Thesis Computing Science

Leakage Resilience of the Suffix Keyed Sponge

Henk Berendsen
s1054308

January 7, 2023

First supervisor/assessor:
Dr. Ir. Bart Mennink

Second assessor:
Prof. Dr. Joan Daemen

Radboud University

Abstract

In this thesis, we analyse the tightness of the non-adaptive leakage resilient pseudorandom function (NALR-PRF) security bound on the Suffix Keyed Sponge formalised by Dobraunig and Mennink (ToSC 2019/4). We show that this bound is not tight due to an unnoticed effect of the leakage function on the size of the largest multicollision an attacker can use to attack the Suffix Keyed Sponge. Furthermore, we show how the bound can be tightened when assuming that the leakage function is defined such that it leaks a secret part of the Suffix Keyed Sponge's state, or the Hamming weight of this secret part.

Contents

Chapter 1

Introduction

Background. The National Institute of Standards and Technology (NIST) has initiated the Lightweight Cryptography Standardization Process [2], often abbreviated as NIST LWC. The goal of this process is to standardise lightweight cryptographic algorithms which can run on devices with low computational power. During NIST LWC, many algorithms were submitted as candidates to be standardised.

One of the criteria NIST uses to evaluate the candidates is resistance against side channel attacks [3]: attacks on a cryptographic scheme which exploit physical attributes of the machine the algorithm is running on, such as power consumption or execution time, to learn sensitive information used in the algorithm, like the secret key. Side channel attacks cannot be considered in the classical security model, called black-box security, in which it is assumed that the attacker can only observe the input and output of a cryptographic algorithm. To account for side channel attacks, the leakage resilience security model can be used. In this model, it is assumed that apart from the input and output of the algorithm, the attacker also learns information leaked during execution of the algorithm.

NIST has selected 10 of the candidates of NIST LWC as finalists. One of these finalists is ISAP, a family of authenticated encryption algorithms which is designed to be robust against side channel attacks [1]. The building block in ISAP used for authentication is called the Suffix Keyed Sponge, and it is the focus of this thesis.

The Suffix Keyed Sponge, abbreviated as SuKS, is a MAC function formalised by Dobraunig and Mennink. They have given a bound on the security of the SuKS in both the black-box and leakage resilience security model [7]. Furthermore, Dobraunig and Mennink have shown that the bound they gave on the black-box security of the SuKS is tight [8]. This means that, in the black-box security model, the complexity of the best possible attacks on the SuKS match the bound. However, it has yet to be shown whether the bound on the leakage resilience security of the SuKS is tight.

Contribution of this thesis. In this thesis, the tightness of the Suffix Keyed Sponge in the leakage resilience security model is analysed from the perspective of an attacker. The most important result of this analysis is that the leakage resilience security bound on the SuKS is not tight. Specifically, one of the terms in this bound is supposed to match the complexity of a key recovery attack, but the complexity of this attack is actually greater than the aforementioned term. This discrepancy between the complexity of the term and the attack is caused by an effect of the leakage on the size of the largest multicollision for the SuKS (the largest set of distinct inputs to the SuKS which result in the same output) found by the attacker.

In the bound, the size of the largest multicollision is estimated by the multicollision limit function, defined by Daemen et al. [6]. In an effort towards tightening the bound, we analyse how the effect of leakage on the size of the largest multicollision can be incorporated into the multicollision limit function for a specific type of leakage. The type of leakage we consider is the leakage of the Hamming weight of a value, which for binary values is equivalent to the amount of 1's in the value.

Related Work. To the best of our knowledge, there exists no tightness analysis of the leakage resilience security bound on the SuKS. Our analysis is based on the bound given by Dobraunig and Mennink, and the attacks used in our analysis are derived from the attacks used in their tightness analysis of the black-box security bound.

We estimate the size of the largest multicollision when considering leakage by using a modified version of the multicollision limit function, defined in the aforementioned article by Daemen et al. In this article, an upper bound on the multicollision limit function which can be computed more efficiently is also given. In order to be able to use this upper bound, we give an upper bound on the modified multicollision limit function in terms of the original multicollision limit function, in which the attacker can use more distinct inputs to the SuKS to find collisions. The method we use to get this upper bound is derived from a method used in the same article by Daemen et al.

This modified version of the multicollision limit function considers leakage of the Hamming weight of a value. We have chosen this type of leakage because it has been shown that leaking the Hamming weight of a value is a realistic modelling of leakage [9, 12, 13].

Overview. First, the notation and concepts necessary to understand the thesis are explained in Chapter 2. Then, the tightness analysis of the leakage resilience security bound on the SuKS is described in Chapter 3. Afterwards, the analysis of the multicollision limit function considering Hamming weight leakage is given in Chapter 4. Finally, the results of these two analyses are discussed in Chapter 5.

Chapter 2

Preliminaries

2.1 Notation

In this section, notation used in the rest of the thesis is defined.

Definition 2.1 (Left-most and right-most bits of a bit string)**.** $\mathrm{left}_k(S)$ and $\mathrm{right}_k(S)$ denote respectively the k left-most and k right-most bits of the bit string S.

Definition 2.2 (Bit string concatenation)**.** Let S_1 and S_2 be two bit strings. $S_1||S_2$ is the bit string formed by concatenating S_1 and S_2.

Definition 2.3 (Sets of bit strings)**.** Let $\{0,1\}^k$ denote the set of bit strings of length k for any $k \in \mathbb{N}$ and $\{0,1\}^*$ the set of bit strings of any length.

Definition 2.4 (Probability)**.** Let $\mathbf{Pr}(E)$ denote the probability that event E occurs, given as a number in the interval $[0,1]$.

Definition 2.5 (Hamming weight)**.** Let $\mathrm{HW}(S)$ denote the Hamming weight of bit string S, i.e. the amount of bits in S equal to 1.

Definition 2.6 (Falling factorial)**.** The nth falling factorial of x, denoted by $(x)_n$, is defined as $\frac{x!}{(x-n)!}$.

2.2 Mathematical Bounds

Mathematical bounds can be used to simplify inequalities. In this section, we list the mathematical bounds used in this thesis, along with a correctness proof for each bound,

Lemma 2.1. *Let $A, B, n \in \mathbb{N}$. If $A \geq B$, then $\frac{(A)_n}{(B)_n} \geq \left(\frac{A}{B}\right)^n$ holds for any n such that $n \leq B$.*

Proof. We prove by induction on n that Lemma 2.1 holds for all $n \in \mathbb{N}$ with $n \leq B$:

Base case.

$\frac{(A)_0}{(B)_0} \geq \left(\frac{A}{B}\right)^0$ because $\frac{(A)_0}{(B)_0} = \frac{\frac{A!}{(A-0)!}}{\frac{B!}{(B-0)!}} = \frac{1}{1} = 1 = \left(\frac{A}{B}\right)^0$.

Inductive case.

For all $k \in \mathbb{N}$ such that $k+1 \leq B$, we have to prove that:

$$\frac{(A)_{k+1}}{(B)_{k+1}} \geq \left(\frac{A}{B}\right)^{k+1} \quad \text{if } A \geq B.$$

We take as Induction Hypothesis (IH):

$$\frac{(A)_{k}}{(B)_{k}} \geq \left(\frac{A}{B}\right)^{k} \quad \text{if } A \geq B.$$

Proof:

$$\begin{aligned}
\frac{(A)_{k+1}}{(B)_{k+1}} &= \frac{\frac{A!}{(A-(k+1))!}}{\frac{B!}{(B-(k+1))!}} \\
&= \frac{\frac{A!}{(A-k-1)!} \cdot \frac{A-k}{A-k}}{\frac{B!}{(B-k-1)!} \cdot \frac{B-k}{B-k}} \\
&= \frac{\frac{A!(A-k)}{(A-k)!}}{\frac{B!(B-k)}{(B-k)!}} \\
&= \frac{(A)_k\,(A-k)}{(B)_k\,(B-k)} \\
&= \frac{(A)_k}{(B)_k} \cdot \frac{A-k}{B-k} \\
&\overset{\text{IH}}{\geq} \left(\frac{A}{B}\right)^k \cdot \frac{A-k}{B-k} \\
&\overset{*}{\geq} \left(\frac{A}{B}\right)^k \cdot \frac{A}{B} \\
&= \left(\frac{A}{B}\right)^{k+1}.
\end{aligned}$$

In the step labelled with the * symbol, we use that $\frac{A-k}{B-k} \geq \frac{A}{B}$. This holds if $A \geq B$, because the inequality can be simplified as follows:

$$\begin{gathered}
\frac{A-k}{B-k} \geq \frac{A}{B}, \\
AB - kB \geq AB - kA, \\
-kB \geq -kA, \\
B \leq A.
\end{gathered}$$

Therefore, we have proven by mathematical induction that Lemma 2.1 holds for all $n \in \mathbb{N}$ with $n \leq B$. □

Lemma 2.2. *$1 + x \leq e^x$ for all $x \in \mathbb{R}$.*

Proof. Let $f(x) = e^x - (1 + x)$. If $f(x) \geq 0$ for all $x \in \mathbb{R}$, then Lemma 2.2 holds. First, we find the first and second derivative of $f(x)$:

$$\begin{aligned} f'(x) &= \frac{\mathrm{d}}{\mathrm{d}x}\left[e^x - (1 + x)\right] = \frac{\mathrm{d}}{\mathrm{d}x}\left[e^x\right] - \frac{\mathrm{d}}{\mathrm{d}x}\left[1 + x\right] = e^x - 1. \\ f''(x) &= \frac{\mathrm{d}}{\mathrm{d}x}\left[e^x - 1\right] = \frac{\mathrm{d}}{\mathrm{d}x}\left[e^x\right] - \frac{\mathrm{d}}{\mathrm{d}x}\left[1\right] = e^x. \end{aligned}$$

Then, we find the critical points of $f(x)$, which are the values x where $f'(x) = 0$. Since $f'(x) = e^x - 1$, we know that $f'(x) = 0$ if and only if $x = 0$. Using the second derivative test, we find that f has a local minimum at $x = 0$, because $f''(0) = e^0 = 1 > 0$. Because $x = 0$ is the only critical point of f, it follows that $x = 0$ is the global minimum of f.

$f(0) = e^0 - (1 + 0) = 1 - 1 = 0$, therefore $f(0) \geq 0$. Since $f(x) \geq 0$ for the value x where $f(x)$ is minimal, it certainly holds that $f(x) \geq 0$ for all $x \in \mathbb{R}$. Therefore, we have proven that Lemma 2.2 holds. □

Lemma 2.3. *$\left(1 - \frac{1}{x}\right)^x \geq e^{-\frac{x}{x-1}}$ for all $x \in \mathbb{R} \setminus \{0, 1\}$.*

Proof. We first prove that $1 - \frac{1}{x} \geq e^{-\frac{1}{x-1}}$ for all $x \in \mathbb{R} \setminus \{0, 1\}$. Let $y = \frac{1}{x}$. We have for all $y \in \mathbb{R} \setminus \{1\}$ that:

$$1 - y = \frac{1}{\frac{1}{1-y}} = \frac{1}{\frac{1-y+y}{1-y}} = \frac{1}{\frac{1-y}{1-y} + \frac{y}{1-y}} = \frac{1}{1 + \frac{y}{1-y}}. \tag{1}$$

Let $z = \frac{y}{1-y}$. Using that $1 + z \leq e^z$ for all $z \in \mathbb{R}$ (see Lemma 2.2), we get:

$$\frac{1}{1 + \frac{y}{1-y}} = \frac{1}{1 + z} \geq \frac{1}{e^z} = e^{-z} = e^{-\frac{y}{1-y}}. \tag{2}$$

It follows from (1) and (2) that:

$$1 - y \geq e^{-\frac{y}{1-y}}. \tag{3}$$

Substituting $\frac{1}{x}$ for y in (3) gives:

$$1 - \frac{1}{x} \geq e^{-\frac{\frac{1}{x}}{1-\frac{1}{x}}} = e^{-\frac{1}{x\left(1-\frac{1}{x}\right)}} = e^{-\frac{1}{x-1}}. \tag{4}$$

With (4), we have proven that $1 - \frac{1}{x} \geq e^{-\frac{1}{x-1}}$ holds for all $x \in \mathbb{R} \setminus \{0, 1\}$. Now, raising both sides to the power of x gives:

$$\left(1 - \frac{1}{x}\right)^x \geq \left(e^{-\frac{1}{x-1}}\right)^x = e^{-\frac{x}{x-1}}. \tag{5}$$

It follows from (5) that Lemma 2.3 is proven. □

Lemma 2.4. $e^{\frac{2^x}{2^x-1}} \leq e^2$ *on the interval* $[1, \infty)$.

Proof. Let $h(x) = e^{\frac{2^x}{2^x-1}}$. To prove the Lemma, we first show that $h(x)$ is decreasing for all $x \in [1, \infty)$, which is the case if the derivative of $h(x)$ is less than 0 for all $x \in [1, \infty)$.

We start by finding the derivative of $h(x)$. Let $f(x) = e^x$ and $g(x) = \frac{2^x}{2^x-1}$. Then $h(x) = f(g(x))$. Using the chain rule, we get:

$$\begin{aligned} h'(x) &= f'(g(x)) \cdot g'(x) \\ &= \frac{\mathrm{d}}{\mathrm{d}x}\left[e^{g(x)}\right] \cdot g'(x) \\ &= e^{g(x)} \cdot g'(x) \\ &= e^{\frac{2^x}{2^x-1}} \cdot \frac{\mathrm{d}}{\mathrm{d}x}\left[\frac{2^x}{2^x-1}\right]. \end{aligned}$$

Using the quotient rule and that $\frac{\mathrm{d}}{\mathrm{d}x}[2^x] = 2^x \ln(2)$, we get:

$$\begin{aligned} h'(x) &= e^{\frac{2^x}{2^x-1}} \cdot \frac{(2^x-1)\frac{\mathrm{d}}{\mathrm{d}x}[2^x] - 2^x \frac{\mathrm{d}}{\mathrm{d}x}[2^x-1]}{(2^x-1)^2} \\ &= e^{\frac{2^x}{2^x-1}} \cdot \frac{(2^x-1)\frac{\mathrm{d}}{\mathrm{d}x}[2^x] - 2^x\left(\frac{\mathrm{d}}{\mathrm{d}x}[2^x] - \frac{\mathrm{d}}{\mathrm{d}x}[1]\right)}{(2^x-1)^2} \\ &= e^{\frac{2^x}{2^x-1}} \cdot \frac{(2^x-1)(2^x\ln(2)) - 2^x(2^x\ln(2))}{(2^x-1)^2} \\ &= e^{\frac{2^x}{2^x-1}} \cdot \frac{(2^x-1-2^x)(2^x\ln(2))}{(2^x-1)^2} \\ &= e^{\frac{2^x}{2^x-1}} \cdot -\frac{2^x\ln(2)}{(2^x-1)^2}. \end{aligned}$$

Next, we show that $h'(x) < 0$:

$$e^{\frac{2^x}{2^x-1}} \cdot -\frac{2^x\ln(2)}{(2^x-1)^2} \overset{?}{<} 0.$$

We know that $\frac{2^x\ln(2)}{(2^x-1)^2}$ is positive for all $x \in [1, \infty)$. Hence, if we divide by $-\frac{2^x\ln(2)}{(2^x-1)^2}$ we are dividing by a negative number and the inequality sign flips:

$$e^{\frac{2^x}{2^x-1}} \overset{?}{>} 0.$$

Because e raised to the power of any real number is positive, the above inequality holds for any real number x on the interval $[1, \infty)$, and we find that $h(x)$ is decreasing on the interval $[1, \infty)$.

Because $h(x)$ is decreasing on this interval, it follows that $h(x)$ is maximal for $x = 1$ on this interval. Therefore, if $h(1) \leq e^2$, then certainly $h(x) \leq e^2$ on the interval $[1, \infty)$.

It holds that $h(1) \leq e^2$ because:

$$h(1) = e^{\frac{2^1}{2^1-1}} = e^{\frac{2}{1}} = e^2.$$

Therefore, $h(x) \leq e^2$ on the interval $[1, \infty)$ and we have proven that Lemma 2.4 holds. $\square$

2.3 Suffix Keyed Sponge

The Suffix Keyed Sponge is a MAC function formalised by Dobraunig and Mennink [7]. Before going into detail about the SuKS, we first explain the sponge construction defined by Bertoni et al. [5], which the SuKS is partially based on.

2.3.1 Explanation of the sponge construction

The sponge construction (see Figure 1) has a b-bit state consisting of a rate of size r (the outer part of the state) and a capacity of size c (the inner part of the state), with $b = r + c$. The initial value of the state is 0^b. The input to the sponge function P is 'absorbed' by splitting it into r-bit blocks $P_1, P_2, \ldots, P_\ell$ and adding these blocks bitwise to the outer part of the state. The output Z is then 'squeezed' by concatenating the r-bit blocks $Z_1, Z_2, \ldots$ until the desired output length has been reached. Each time after a block has been absorbed or squeezed, a random permutation p is evaluated on the state.

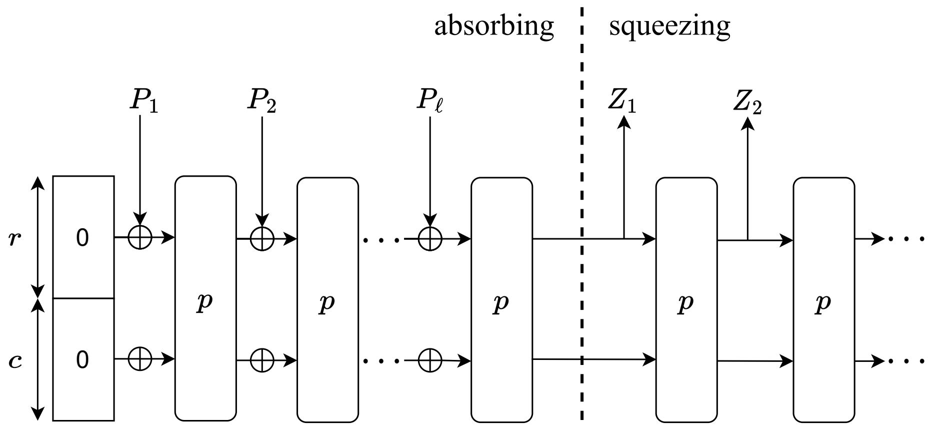

Figure 1: Sponge Construction

2.3.2 Explanation of the Suffix Keyed Sponge

The Suffix Keyed Sponge is depicted in Figure 2. The first part of the SuKS (the part until the state U) is identical to the absorbing phase of the sponge construction. After this, the state is changed by evaluating a function G with as input the secret key K on the outer s bits of the state. The random permutation p is evaluated one final time on the state, and the outer t bits of the resulting output form the tag T.

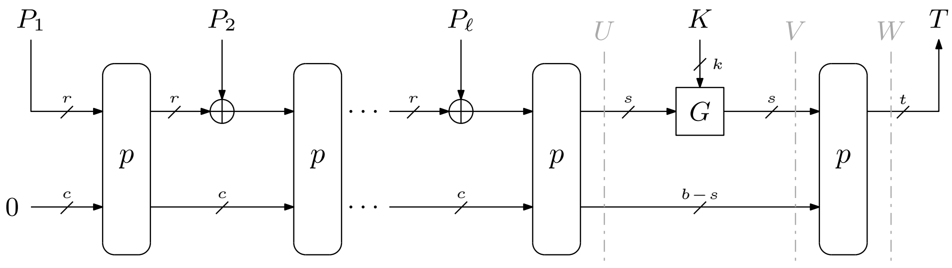

Figure 2: Suffix Keyed Sponge

The intuition behind the SuKS is that by only using K at the end of the scheme, just a small part of the scheme is vulnerable to leakage: only the evaluation of G and the last evaluation of p can leak secret information. All other evaluations of p are not vulnerable to leakage because their input and output are not secret.

2.4 PRF security and NALR-PRF security

2.4.1 PRF security

Let $b, k, t \in \mathbb{N}$ and $m \in \mathbb{N} \cup \{*\}$. PRF (pseudorandom function) security is a well-established method to measure the security of a function $F : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^t$, which takes as input a k-bit key K and m-bit message M, produces a t-bit tag T, and uses a b-bit permutation $p : \{0,1\}^b \to \{0,1\}^b$ as a primitive.

The PRF security of F against an adversary $\mathcal{A}$, denoted by $\mathbf{Adv}_F^{\mathrm{prf}}(\mathcal{A})$, is defined as the distinguishing advantage of the adversary when distinguishing the function F instantiated with some key K and permutation p, denoted by F_K^p, from the random oracle $\mathcal{RO}$. This distinguishing advantage describes how difficult it is for the adversary to distinguish F_K^p from $\mathcal{RO}$: the lower this advantage, the harder the two are to distinguish.

To distinguish F_K^p from $\mathcal{RO}$, the adversary can query both F_K^p and $\mathcal{RO}$ with m-bit messages M to receive t-bit tags T, but does not know which of the two functions it is querying. The adversary can also query the permutation p. $\mathbf{Adv}_F^{\mathrm{prf}}(\mathcal{A})$ is often expressed in terms of construction queries q and primitive queries N: construction queries are made to F_K^p or $\mathcal{RO}$, while primitive queries are made to p. Construction queries are also called online queries, because the attacker cannot compute the result of the query on their own machine (due to secret information in the construction such as K in F_K^p, or due to the construction being an ideal counterpart such as $\mathcal{RO}$); primitive queries are also called offline queries, since the attacker can compute the result of the query on their own machine because p is public.

A function F is considered to be a PRF if $\mathbf{Adv}_F^{\text{prf}}(\mathcal{A})$ is very low, meaning it is very difficult for an adversary $\mathcal{A}$ to distinguish F_K^p from $\mathcal{RO}$. Furthermore, a PRF is a secure MAC function in the black-box security model [4, 10, 11], which makes the notion of PRF security very useful in this model.

2.4.2 NALR-PRF security

Although the PRF security of a function F against an adversary $\mathcal{A}$ is a useful notion in the black-box security model, it does not take into account that the adversary could exploit leakage in F, and can therefore not be used in the leakage resilience security model. To describe the security of the SuKS in this model, Dobraunig and Mennink transformed the definition of PRF security to NALR-PRF (non-adaptive leakage resilient pseudorandom function) security [7].

In NALR-PRF security, denoted by $\mathbf{Adv}_F^{\text{nalr-prf}}(\mathcal{A})$, the adversary again has to distinguish F_k^p and $\mathcal{RO}$, with access to both functions and p. Additionally, the attacker has access to a leaky version of F_k^p which leaks, for every evaluation of every primitive used in F, at most λ bits of information about the input or output to that primitive, for some $\lambda \in \mathbb{N}$. This leakage is represented by mathematical functions, and there is a separate leakage function for each type of primitive. The leakage function for a primitive takes as input both the input and output of that primitive; the output of the leakage function is the leaked information of at most λ bits. $\mathbf{Adv}_F^{\text{nalr-prf}}(\mathcal{A})$ is defined as the maximum distinguishing advantage of an adversary $\mathcal{A}$ distinguishing between F_k^p and $\mathcal{RO}$, over all possible combinations of leakage functions for the primitives in F.

To illustrate how these leakage functions work, we give an example of leakage functions for the SuKS. The SuKS contains two primitives: the b-bit permutation p and function $G : \{0,1\}^k \times \{0,1\}^s \to \{0,1\}^s$. Let L_p and L_G be the leakage functions for the primitives p and G respectively. The input and output of p are both b bits long; for G, the inputs are a k-bit key and s-bit value, and the output is also an s-bit value. Some example definitions for L_p and L_G are:

$$L_p : \{0,1\}^b \times \{0,1\}^b \to \{0,1\}^\lambda$$
$$L_p(V, W) = \text{right}_\lambda(W),$$

$$L_G : \{0,1\}^k \times \{0,1\}^s \times \{0,1\}^s \to \{0,1\}^\lambda$$

$$L_G(K, U, V) = \mathrm{left}_\lambda(K).$$

In the case of the SuKS, we do not have to consider the leakage of each evaluation of this primitive, but only of the single evaluation of G and the last evaluation of p. This is because those evaluations are the only ones where secret information can be leaked, as explained in Section 2.3, and the attacker does not learn anything from the leakage of public information.

2.5 Uniform and Universal Functions

The notion of uniformity of a function describes how likely it is that a certain input maps to a certain output; the notion of universality of a function describes how likely it is that this function gives the same output for distinct inputs. Dobraunig and Mennink have given definitions to quantify how uniform and universal a function is [7]:

Definition 2.7 ($2^{-\delta}$-uniformity)**.** A function $G : \{0,1\}^k \times \{0,1\}^s \to \{0,1\}^s$ is $2^{-\delta}$-uniform if, for any $K \in \{0,1\}^k$ and $X, Y \in \{0,1\}^s$, δ is the largest real number such that

$$\mathbf{Pr}(G(K, X) = Y) \leq 2^{-\delta}.$$

Definition 2.8 ($2^{-\varepsilon}$-universality)**.** A function $G : \{0,1\}^k \times \{0,1\}^s \to \{0,1\}^s$ is $2^{-\varepsilon}$-universal if, for any $K \in \{0,1\}^k$ and $X, X' \in \{0,1\}^s$, ε is the largest real number such that

$$\mathbf{Pr}(G(K, X) = G(K, X')) \leq 2^{-\varepsilon}.$$

With these definitions, we can prove the following theorem:

Theorem 2.1. *For any function $G : \{0,1\}^k \times \{0,1\}^s \to \{0,1\}^s$ which is $2^{-\delta}$-uniform and $2^{-\varepsilon}$-universal, it holds that $\varepsilon \geq \delta$.*

Proof. Because G is $2^{-\varepsilon}$-universal, we know that $\mathbf{Pr}(G(K, X) = G(K, X')) \leq 2^{-\varepsilon}$. We show that it also holds that $\mathbf{Pr}(G(K, X) = G(K, X')) \leq 2^{-\delta}$:

$$\begin{aligned}
&\mathbf{Pr}(G(K, X) = G(K, X')) \\
&\quad = \sum_Y \left[\mathbf{Pr}(G(K, X) = G(K, X') \mid G(K, X') = Y) \cdot \mathbf{Pr}(G(K, X') = Y)\right] \\
&\quad = \sum_Y \left[\mathbf{Pr}(G(K, X) = Y) \cdot \mathbf{Pr}(G(K, X') = Y)\right] \\
&\quad \leq \sum_Y \left[2^{-\delta} \cdot \mathbf{Pr}(G(K, X') = Y)\right] \\
&\quad = 2^{-\delta} \cdot \sum_Y \left[\mathbf{Pr}(G(K, X') = Y)\right]
\end{aligned}$$

$$\begin{aligned} &= 2^{-\delta} \cdot 1 \\ &= 2^{-\delta}. \end{aligned}$$

Now, we know that:

$$\mathbf{Pr}(G(K,X) = G(K,X')) \leq 2^{-\varepsilon}, \tag{6}$$

$$\mathbf{Pr}(G(K,X) = G(K,X')) \leq 2^{-\delta}. \tag{7}$$

Furthermore, we know that ε is the largest real number such that (6) holds, and hence $2^{-\varepsilon}$ is the smallest real number such that (6) holds. Therefore, it must be the case that $2^{-\varepsilon} \leq 2^{-\delta}$, which can only be the case if $\varepsilon \geq \delta$. Hence, we have proven Theorem 2.1. □

2.6 Multicollision limit function

A multicollision for a function F is a set of one or more distinct inputs which are mapped to the same output in F. To estimate the maximum size of a multicollision, the multicollision limit function defined by Daemen et al. [6] can be used. The multicollision limit function uses the problem of throwing balls into bins to model the collisions. The function has also been used by Dobraunig and Mennink [7] to give bounds on the security of the SuKS. Their definition of the multicollision limit function is repeated here:

Definition 2.9 (Multicollision limit function)**.** Let $q, b, s \in \mathbb{N}$ such that $s \leq b$. Consider the experiment of throwing q balls uniformly at random in 2^{b-s} bins, and denote by μ the maximum number of balls in any single bin. The multicollision limit function $\mu^{q}_{b-s,s}$ is defined as the smallest natural number x that satisfies

$$\mathbf{Pr}(\mu > x) \leq \frac{x}{2^s}.$$

The experiment described in this definition is illustrated in Figure 3 with a toy example where $q = 4$ and $b - s = 2$. In other words, 4 balls are thrown in 4 bins. In this example, two balls are thrown in the first bin, one ball is thrown in the second and fourth bin and none are thrown in the third bin. In this toy example, $\mu = 2$ because the single bin with the maximum number of balls is bin 1 with two balls.

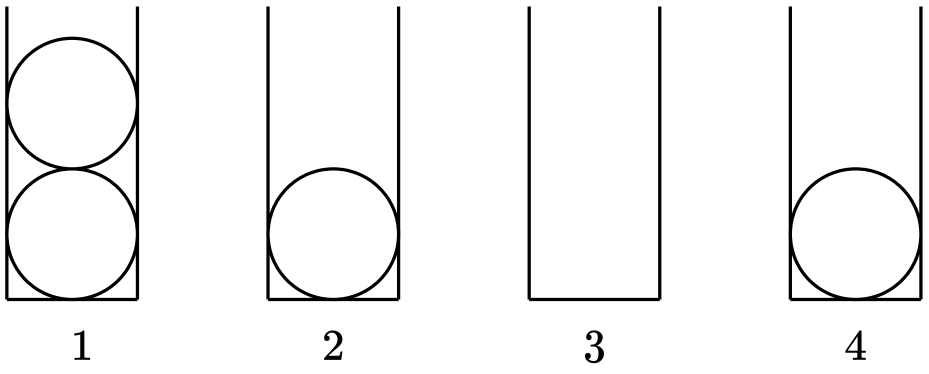

Figure 3: Toy example of a balls-and-bins experiment

The intuition behind the multicollision limit function is that multicollisions can be compared to a balls-and-bins experiment: the q balls represent q distinct input values to some function F; throwing a ball into the bins is an analogy to evaluating the function F on an input value; the bin in which the ball is thrown represents the output of F for the given input value.

In this analogy, having multiple balls in one bin means that the input values which these balls represent form a multicollision for the function F. Furthermore, the value μ represents the size of the largest multicollision.

In the definition of the multicollision limit function, it is assumed that the balls are thrown with replacement, which in the analogy means it is possible the same input value is evaluated multiple times. From an attacker's perspective, it is not useful to evaluate the same input value multiple times; an attacker with full control over the input to F would try different inputs each time, which is analogous to the balls being thrown without replacement. Daemen et al. have shown in their article that for a balls-and-bins problem with q balls and $b - s$ bins where the balls are thrown uniformly without replacement, the maximum amount of balls in any single bin can be estimated by $\mu^{2q}_{b-s,s}$.

The multicollision limit function allows us to ensure that when considering an attacker with a bounded amount of resources, the probability of this attacker achieving a multicollision larger than a certain value is very unlikely. This notion is useful for finding the security bound on a cryptographic scheme.

2.7 Security Bounds on the Suffix Keyed Sponge

Dobraunig and Mennink have given a PRF security and NALR-PRF security bound on the SuKS in Section 5 and 6 of their article formalising the SuKS

[7], respectively. These security bounds are given in (8) and (9):

$$\mathbf{Adv}_F^{\mathrm{prf}}(\mathcal{A}) \leq \frac{2N^2}{2^c} + \frac{\mu_{b-s,s}^{2(N-q)} \cdot N}{2^{\min\{\delta,\varepsilon\}}} + \frac{\mu_{t,b-t}^{q} \cdot N}{2^{b-t}}, \tag{8}$$

$$\mathbf{Adv}_F^{\mathrm{nalr\text{-}prf}}(\mathcal{A}) \leq \frac{2N^2}{2^c} + \frac{\mu_{s,b-s}^{2(N-q)}}{2^{b-s}} + \frac{\mu_{b-s,s}^{2(N-q)} \cdot N}{2^{\min\{\delta,\varepsilon\}-\mu_{s,b-s}^{2(N-q)}\lambda}} + \frac{\mu_{t,b-t}^{2q} \cdot N}{2^{b-t-\lambda}}. \tag{9}$$

Chapter 3

Tightness of Suffix Keyed Sponge NALR-PRF security bound

In this chapter, we analyse the tightness of the NALR-PRF security bound on the SuKS given in (9) on page 15. We will frequently refer to primitives and states in the SuKS which are depicted in Figure 2 on page 10.

To analyse the tightness of the NALR-PRF security bound, the tightness of each term in the bound must be analysed separately, since each term serves to describe the success probability of a different attack. Because the first term in the NALR-PRF security bound equals the first term in the PRF security bound, and Dobraunig and Mennink have already proven that the PRF security bound is tight [8], we will leave this term out of our analysis.

In the following three sections, we will analyse the tightness of the latter three terms in the NALR-PRF security bound given in (9). The tightness of the second term will be analysed using probability theory. The tightness of the last two terms will be analysed by attempting to construct a matching attack: an attack on the SuKS which matches the success probability given in the term. In each attack, we assume that the attacker can make N primitive queries and q construction queries. As mentioned in the introduction of the thesis, the matching attacks used in this analysis are based on the matching attacks given in the PRF security tightness analysis by Dobraunig and Mennink.

Because the second term of the leakage bound originates from the matching attack for the third term, as will be explained in the section about the second term, we discuss the third term before the second.

3.1 Matching attacks on term 3 of the NALR-PRF bound

Term 3 of the NALR-PRF security bound is similar to term 2 of the PRF security bound. There are 2 matching attacks on this latter term, which will be discussed in the next two sections, before we attempt to construct matching attacks on the former term.

3.1.1 First attack

In the first attack, it is assumed that G is the XOR function, and therefore $s = k$. The main idea of the attack is to recover the state V for a given construction query, so that the key K can be retrieved by computing $K = \text{left}_s(U) \oplus \text{left}_s(V)$. U can be computed offline because K only comes into play after the SuKS has reached the state U.

The attacker makes one construction query for some message M to get the corresponding tag T. Then, the attacker guesses the value of $\text{left}_k(V)$ belonging to this query. Because $\text{right}_{b-k}(V)$ is equal to $\text{right}_{b-k}(U)$, and U can be computed offline, the attacker knows $\text{right}_{b-k}(V)$.

Let the attacker's guess of $\text{left}_k(V)$ be called Z. To verify if Z is a correct guess, the attacker can check whether $\text{left}_t(p(Z||\text{right}_{b-k}(V))) = T$. Because the attacker can make N primitive queries, and must verify each guess with one evaluation of p, the attacker can make at most N guesses. This way, the attack has a success probability of $\frac{N}{2^s}$, because N different guesses are made and the probability that any guess is correct is 2^{-s} because G being the XOR is 2^{-s}-uniform.

The attack can be sped up by finding a multicollision of μ plaintexts which result in the same value for $\text{right}_{b-k}(U) = \text{right}_{b-k}(V)$, and computing the tag of each plaintext. This way, the guess for $\text{left}_k(V)$ could be a match for any of the μ plaintexts.

The amount of plaintexts μ can be estimated using the multicollision limit function (see Section 2.6). Because the attacker tries to find collisions for $\text{right}_{b-k}(U)$, a value of size $b - k = b - s$, the amount of bins is 2^{b-s}. The attacker can find collisions by making offline queries to p. The attacker can make a total of N queries to p, but the attacker also uses queries to p in the construction queries to the SuKS. The amount of queries the attacker can use to find collisions is approximately $N - q$. Finally, because the attacker can try a different input for p in every primitive query, we consider a balls-and-bins problem where the balls are thrown without replacement. Therefore, the multicollision limit function which estimates μ is $\mu_{b-s,s}^{2(N-q)}$. Taking the multicollision into account, the success probability of the attack becomes $\frac{\mu_{b-s,s}^{2(N-q)} \cdot N}{2^s}$.

3.1.2 Second attack

In the second attack, it is assumed that G is a PRF which is hard to invert, making it not as easy to recover the key as with an XOR. Instead, this attack mounts a forgery, meaning that the attacker finds the tag corresponding to a message without making a construction query for the message.

The main idea of the attack is to find 2 plaintexts P_1 and P_2 with a collision for $\text{left}_s(U)$. For the plaintexts, the respective states U_1 and U_2 are computed offline, and a construction query is made to get the tag T_1 for P_1. The forgery can be made by finding $\text{left}_s(V)$, which has the same value for both plaintexts, since the input to G is the same for both due to the collision in $\text{left}_s(U)$. $\text{left}_s(V)$ is found by making N guesses and verifying for each guess Z whether $\text{left}_t(p(Z||\text{right}_{b-s}(U_1))) = T_1$. If the guess Z is correct, the attacker has a forgery for P_2 because they can compute the corresponding tag $T_2 = \text{left}_t(p(Z||\text{left}_{b-s}(U_2))$. This attack has success probability $\frac{N}{2^\delta}$, because G is $2^{-\delta}$-uniform.

The attack can be sped up by finding a multicollision of μ plaintexts $P_1^1, P_1^2, \ldots, P_1^\mu$ with the same value for $\text{right}_{b-s}(U)$. The attacker can then find for each plaintext P_1^i another plaintext P_2^i with the same value for $\text{left}_s(U)$. The attacker also computes the corresponding states U and tags as described above, and can then make guesses Z to find the state V of one of the plaintext pairs (P_1^i, P_2^i). If the guess for Z is correct for any of these pairs, the attacker can create a forgery. Because μ is estimated by $\mu_{b-s,s}^{2(N-q)}$, just like in the first attack, this results in the success probability $\frac{\mu_{b-s,s}^{2(N-q)} \cdot N}{2^\delta}$.

3.1.3 Deriving the bound

The only difference between the success probabilities of the attacks is the denominator: in the first attack, the denominator is 2^s, while in the second attack, the denominator is 2^δ. The difference between the denominators originated from the fact that in the first attack, G was the 2^{-s}-uniform XOR, whereas in the second attack, G was a 2^{-s}-uniform PRF. Because G has an output of s bits, it can be at best 2^{-s}-uniform, so we know that $2^{-\delta} \geq 2^{-s}$. Therefore, the first attack has at least the complexity of the second attack. The complexity of the second attack still differs from term 2 of the PRF security bound in the denominator, because the denominator of this term is $2^{\min\{\delta,\varepsilon\}}$. However, since we have shown in Theorem 2.1 on page 12 that $\varepsilon \geq \delta$, $2^{\min\{\delta,\varepsilon\}}$ can be simplified to δ and therefore the term does exactly match the second attack.

3.1.4 Attack in the NALR-PRF security model

For both attacks on term 2 of the black-box bound, a similar attack can be performed in the leaky setting by using a leakage function for the permutation

$p : \{0,1\}^b \to \{0,1\}^b$ such that $p(V) = W$.

For both attacks, the main challenge was guessing $\text{left}_s(V)$. This part of the attacks is what causes the denominator of the term to be $2^{\min\{\delta,\varepsilon\}}$. However, the denominator can become less strong due to leakage.

Suppose we define the leakage function for p such that it is always the same λ bits of V which leak, for example the left-most λ bits:

$$L_p : \{0,1\}^b \times \{0,1\}^b \to \{0,1\}^\lambda$$
$$L_p(V,W) = \text{left}_\lambda(V)$$

This way, the amount of bits of V which need to be guessed is reduced by λ, which would result in a security bound of $\frac{\mu_{b-s,s}^{2(N-q)} \cdot N}{2^{\min\{\delta,\varepsilon\}-\lambda}}$.

However, the security bound can be reduced even more by taking into account that there could also be plaintexts forming a multicollision for $\text{left}_s(U)$. These plaintexts would then also form a multicollision for $\text{left}_s(V)$, which is the value that needs to be guessed. The amount of plaintexts forming a multicollision for $\text{left}_s(V)$ is estimated by $\mu_{s,b-s}^{2(N-q)}$; this multicollision limit function can be explained similarly to the multicollision limit function $\mu_{b-s,s}^{2(N-q)}$ described in Section 3.1.1. We assume that all $\mu_{s,b-s}^{2(N-q)}$ plaintexts have a distinct value for $\text{right}_{b-s}(V)$. We can then define L_p such that a different combination of λ bits leaks for each different value of $\text{right}_{b-s}(V)$ in the $\mu_{s,b-s}^{2(N-q)}$ plaintexts forming the multicollision. Therefore, $\mu_{s,b-s}^{2(N-q)}\lambda$ bits of $\text{left}_s(V)$ leak, and the attack results in the bound $\frac{\mu_{b-s,s}^{2(N-q)} \cdot N}{2^{\min\{\delta,\varepsilon\}-\mu_{s,b-s}^{2(N-q)}\lambda}}$.

3.2 Matching attack on term 2 of the NALR-PRF bound

In the third term of the NALR-PRF security bound analysed in Section 3.1, it was assumed that the size of the multicollision for $\text{left}_s(U)$ is less than or equal to $\mu_{s,b-s}^{2(N-q)}$. However, the NALR-PRF security bound must also take the possibility into account that the size of the multicollision is larger than $\mu_{s,b-s}^{2(N-q)}$, because then the amount of leakage, and consequently the success probability of the attacker, would increase as well.

Let $\mathsf{success}$ describe the event that the attacker successfully breaks the scheme, i.e. mounts a forgery or retrieves the key. Let good describe the event that the size of the multicollision is less than or equal to $\mu_{s,b-s}^{2(N-q)}$. Let bad describe the event that the size of the multicollision is larger than $\mu_{s,b-s}^{2(N-q)}$.

Now, using probability theory, we can derive:

$$\mathbf{Pr}(\mathsf{success}) = \mathbf{Pr}(\mathsf{success} \cup \mathsf{good}) + \mathbf{Pr}(\mathsf{success} \cup \neg\mathsf{good})$$

$$= \mathbf{Pr}(\mathsf{success} \cup \mathsf{good}) + \mathbf{Pr}(\mathsf{success} \cup \mathsf{bad})$$
$$= \mathbf{Pr}(\mathsf{success} \mid \mathsf{good}) \cdot \mathbf{Pr}(\mathsf{good}) + \mathbf{Pr}(\mathsf{success} \mid \mathsf{bad}) \cdot \mathbf{Pr}(\mathsf{bad})$$
$$\leq \mathbf{Pr}(\mathsf{success} \mid \mathsf{good}) + \mathbf{Pr}(\mathsf{bad}).$$

Here $\mathbf{Pr}(\mathsf{good})$ and $\mathbf{Pr}(\mathsf{success} \mid \mathsf{bad})$ are less than or equal to 1, because they are probabilities. Because they are multiplied with another term, leaving them out of the equation results in an equation which is at least as large. Also, note that both these probabilities are usually very close to 1: $\mathbf{Pr}(\mathsf{good})$ is very close to the value 1 because the probability of a large multicollision is small, and $\mathbf{Pr}(\mathsf{success} \mid \mathsf{bad})$ is close to 1 because the attacker's success probability is very high for a large multicollision. Therefore, ignoring these two terms does not change the estimated value of $\mathbf{Pr}(\mathsf{success})$ significantly.

$\mathbf{Pr}(\mathsf{success} \mid \mathsf{good})$ is described by term 3 of the NALR-PRF security bound. $\mathbf{Pr}(\mathsf{bad})$ is described by term 2. This is because, according to the definition of the multicollision limit function, $\mu_{s,b-s}^{2(N-q)}$ is defined to be the smallest natural number x such that the probability that there is a multicollision with a size larger than x is less than or equal to $\frac{x}{2^{b-s}} = \frac{\mu_{s,b-s}^{2(N-q)}}{2^{b-s}}$, which equals term 2 of the bound. Note that for typical parameters of the SuKS, e.g. $s = t = k$, this term is negligible.

3.3 Matching attack on term 4 of the NALR-PRF bound

This term is very similar to term 3 of the PRF security bound, which we will discuss first. In the matching attack for term 3 of the PRF security bound, it is assumed that the function G is again an XOR, and the main idea of the attack is the same as the first matching attack on term 3: recovering the state V in order to find the key. However, the way V is recovered is different.

V is recovered by calling the inverse primitive p^{-1} on a tag T which was computed with a construction query beforehand, and a guess for the remaining $b - t$ bits. For a random permutation, the probability of this guess being correct is $\frac{N}{2^{b-t}}$, with N being the amount of times p^{-1} was called (with different guesses for the rightmost $b - t$ bits).

However, the probability can be improved if there would be multiple plaintexts, and consequently multiple states V which are correct for the given tag, which is the case if these plaintexts form a multicollision for T.

The amount of plaintexts forming a multicollision equals $\mu_{t,b-t}^{2q}$, because the attacker tries to find collisions for the t-bit tag T, which they can only compute using the q construction queries. The superscript of the multicollision limit function is $2q$ because we consider a balls-and-bins problem where the balls are sampled without replacement. Since T is a part of the state W, and W is the output of the last evaluation of p, the balls are analogous to the inputs to this last evaluation of p. Since we assumed G is an XOR, both primitives used in the SuKS are bijective. Hence, the attacker can make sure that the input to the last evaluation of p is different in each construction query, and therefore the balls are sampled without replacement. When taking the multicollision into account, the success probability of the attack becomes $\frac{\mu^{2q}_{t,b-t} \cdot N}{2^{b-t}}$.

This success probability is greater than term 3 of the PRF security bound, $\frac{\mu^{q}_{t,b-t} \cdot N}{2^{b-t}}$, due to the superscript of the multicollision limit function being $2q$ instead of q. It appears this is due to a mistake in the proof of the PRF security bound, but this mistake is not present in the NALR-PRF security bound.

3.3.1 Attack in the NALR-PRF security model

A similar attack can be performed in the leaky setting, where we use a leakage function for the primitive $p : \{0,1\}^b \to \{0,1\}^b$, with $p(V) = W$. Let this leakage function be defined as:

$$
\begin{aligned}
&L_p : \{0,1\}^b \times \{0,1\}^b \to \{0,1\}^\lambda \\
&L_p(V, W) = \mathrm{right}_\lambda(W).
\end{aligned}
$$

This way, the attacker learns an additional λ bits of W apart from the t-bit tag, and therefore only needs to guess $b - t - \lambda$ bits.

The fourth term, $\frac{\mu^{2q}_{t,b-t} \cdot N}{2^{b-t-\lambda}}$, originates from the idea that the matching attack and leakage function described above could be combined in order to guess $b - t - \lambda$ bits with N trials, with the goal of finding a match with one of $\mu^{2q}_{t,b-t}$ different values of V. However, with this bound it is wrongly assumed that each of the $\mu^{2q}_{t,b-t}$ plaintexts in the multicollision will have the same value of $\mathrm{right}_\lambda(W)$: these plaintexts form a multicollision for T, but the rest of the state W may be different.

Because the state W may be different, the guess of $b - t - \lambda$ bits of W made by the attacker can only match the plaintexts in the collision, for which these λ bits in their corresponding state W are equal. We can estimate the amount of such plaintexts with the multicollision limit function $\mu^{2q}_{t+\lambda,b-t-\lambda}$, because this multicollision limit function estimates the size of the largest multicollision for the t-bit tag T, as well as the λ leaked bits.

Therefore, when assuming a leakage function which leaks λ unknown bits of W, in this case the right-most bits, the fourth term in the SuKS NALR-PRF security bound becomes $\frac{\mu^{2q}_{t+\lambda,b-t-\lambda} \cdot N}{2^{b-t-\lambda}}$.

With this assumed leakage function, the amount of multicollisions the attacker can use, estimated by $\mu^{2q}_{t,b-t}$, is independent of the leakage value the attacker learns. This can be seen from the fact that only the amount of bits of leakage plays a role in this expression, and not the actual value. The reason for this independency is that each leakage value is equally likely to occur. Furthermore, each leakage value also gives the same amount of information about the state W, namely λ bits.

However, when assuming a different leakage function, it is possible that the amount of multicollisions available to the attacker does depend on the leakage value the attacker learns. Take for example a leakage function which leaks the Hamming weight of $\text{right}_{b-t}(W)$, defined as follows:

$$L_p : \{0,1\}^b \times \{0,1\}^b \to \{0,1\}^\lambda$$
$$L_p(V,W) = \text{HW}(\text{right}_{b-t}(W)).$$

With this leakage function, leakage values have different probabilities of occurring: for example, the leakage value 0 only occurs when $\text{right}_{b-t}(W) = 0^{b-t}$, while the leakage value 1 occurs for each value of $\text{right}_{b-t}(W)$ which contains exactly one bit equal to 1 (there are $b-t$ such values, since the bit equal to 1 can be placed in $b-t$ different positions). Hence, it is more likely to find collisions if the leaked value is 1, than if the leaked value is 0.

Furthermore, with this leakage function, the amount of information learned about $\text{right}_{b-t}(W)$ depends on the leakage value: for example, the leakage value 0 reveals that $\text{right}_{b-t}(W) = 0^{b-t}$, whereas the leakage value 1 reveals that $\text{right}_{b-t}(W)$ can be one of $b-t$ different values.

3.4 Conclusions of the tightness analysis

From the tightness analysis, we can conclude that the fourth term in the NALR-PRF security bound on the SuKS is not tight, because in the matching attack for this term it was not taken into account that the size of the largest multicollision may change if the attacker also uses leakage. Because this single term in the bound is not tight, it follows that the whole bound is also not tight.

In Section 3.3.1, we have shown that we can obtain a tight bound on the NALR-PRF security of the SuKS, if we only consider leakage functions which simply leak λ bits of $\text{right}_{b-t}(W)$ in the matching attack on the fourth term. This bound is given in (10):

$$\mathbf{Adv}_F^{\text{nalr-prf}}(\mathcal{A}) \le \frac{2N^2}{2^c} + \frac{\mu_{s,b-s}^{2(N-q)}}{2^{b-s}} + \frac{\mu_{b-s,s}^{2(N-q)} \cdot N}{2^{\min\{\delta,\varepsilon\} - \mu_{s,b-s}^{2(N-q)}\lambda}} + \frac{\mu_{t+\lambda,b-t-\lambda}^{2q} \cdot N}{2^{b-t-\lambda}}. \quad (10)$$

It is more difficult to give a tight bound on the NALR-PRF security of the SuKS when considering a leakage function where the size of the multicollision the attacker can use depends on the leaked value, such as a leakage function which leaks the Hamming weight of a value.

In Chapter 4, we will study how the size of the largest multicollision available to the attacker can be estimated when considering leakage functions which leak the Hamming weight of a value. We do so by incorporating the Hamming weight leakage into the multicollision limit function.

Chapter 4

Multicollision limit function with Hamming weight leakage

As mentioned at the end of Section 3.4, the goal of this chapter is to estimate the size of the largest multicollision available to an attacker performing the matching attack on term 4 of the SuKS NALR-PRF security bound, when considering leakage functions which leak the Hamming weight of a value. There are two reasons why we choose this specific type of leakage functions. Firstly, the Hamming weight leakage functions are interesting because the amount of information the attacker learns from the leakage, and the amount of multicollisions the attacker can use, depends on its output, as shown in Section 3.3.1. Secondly, leaking the Hamming Weight of a value is a realistic modelling of leakage [9, 12, 13].

In an intermediary step of this matching attack, the attacker tries to find the b-bit state W of the SuKS. The attacker knows $T = \text{left}_t(W)$, hence they only need to find $\text{right}_{b-t}(W)$. In this chapter, we will generalize this to an attacker who knows $\text{left}_r(W)$ and thus only needs to find $\text{right}_{b-r}(W) = \text{right}_c(W)$, since t and $b-t$ are just two possible values r, c such that $b = r+c$.

Besides knowing $\text{left}_r(W)$, the attacker also learns leaked information from a leakage function L_p^{HW}. This leakage function leaks for the last permutation p used in the SuKS the Hamming weight of the w right-most bits of the output of p, with $1 \leq w \leq c$. It is formally defined as follows, with $\lambda = \lceil \log(w+1) \rceil$:

$$\begin{aligned} L_p^{\text{HW}} &: \{0,1\}^b \times \{0,1\}^b \to \{0,1\}^\lambda \\ L_p^{\text{HW}}(V, W) &= \text{HW}(\text{right}_w(W)). \end{aligned} \tag{11}$$

We define $\lambda = \lceil \log(w+1) \rceil$, because the Hamming weight of a w-bit value can attain $w+1$ possible values (namely 0 up until and including w), and can hence be encoded in $\lceil \log(w+1) \rceil$ bits.

The distribution of outputs of L_p^{HW} is non-uniform, unlike a leakage function which simply leaks a certain amount of bits of the input. This non-uniformity exists because there are different amounts of bit strings of a fixed length for different values of the Hamming weight. For example, for w-bit strings such as $\mathrm{right}_w(W)$, there is one bit string with a Hamming weight of 0 (0^w), and there are w bit strings with a Hamming weight of 1 (all w-bit strings with a single 1, since this single 1 can be in w different positions).

The amount of bit strings of length w with Hamming weight n is $\binom{w}{n}$. This is because the amount of w-bit strings s with $\mathrm{HW}(s) = n$ equals the amount of bit strings with n bits equal to 1. In turn, this amount is equal to the number of ways to choose n out of w bits which are equal to 1, while the other bits are equal to 0, which is described by the binomial coefficient $\binom{w}{n}$.

To analyse how the leakage function L_p^{HW} influences the amount of multicollisions, we will incorporate this leakage function into the multicollision limit function. In order to do this, we first define a balls-and-bins problem which takes the non-uniform distribution of the Hamming weight into account.

4.1 Balls-and-bins problem

To model the Hamming weight leakage in a balls-and-bins problem, the balls and bins have to be redefined:

- We consider the balls to be b-bit values, because the attacker is trying to find collisions for a part of the W, which is the output of an evaluation of the b-bit permutation p.
- The attacker now learns the value of $\mathrm{left}_r(W)$ as well as $\mathrm{HW}(\mathrm{right}_w(W))$. Hence, we consider a bin for each possible combination of these values. Because the Hamming weight leakage function's codomain contains $w + 1$ different values, there are $2^r \cdot (w + 1)$ bins.

In the original balls-and-bins problem of the multicollision limit function, the balls are thrown with replacement according to a uniform distribution. In our redefined balls-and bins problem, the balls are thrown according to a non-uniform distribution denoted by $D_{\mathsf{HW\text{-}nr}}$. The distribution is non-uniform due to the balls being thrown without replacement (since this was the case in the matching attack on term 4 of the SuKS NALR-PRF security bound, as described in Section 3.3), and due to the non-uniform distribution of the Hamming weight itself, which was explained at the start of this section.

4.2 Bounding the multicollision limit function

The maximum amount of balls in a single bin μ for the balls-and-bins problem defined in the previous section can be described by

$$\mu_{r',c'}^{q,D_{\mathsf{HW\text{-}nr}}}$$

with $r' = r + \log(w+1)$ and $c' = c - \log(w+1)$.

The superscript in this multicollision limit function denotes that q balls are thrown in the bins according to the distribution $D_{\mathsf{HW\text{-}nr}}$.

According to the definition of the multicollision limit function in Section 2.6, we get from the subscript parameter r' that there are $2^{r'} = 2^{r+\log(w+1)} = 2^r \cdot 2^{\log(w+1)} = 2^r \cdot (w+1)$ bins, and from the subscript parameter c' that we define this multicollision limit function as the smallest x satisfying:

$$\mathbf{Pr}(\mu > x) \leq \frac{x}{2^{c'}} = \frac{x}{2^{c-\log(w+1)}} = \frac{x}{\frac{2^c}{2^{\log(w+1)}}} = \frac{x}{\frac{2^c}{w+1}} = (w+1) \cdot \frac{x}{2^c}.$$

with μ the maximum amount of balls in any single bin.

Because this multicollision limit function is hard to compute due to the non-uniformity of the Hamming weight leakage function, we will prove that it is upper bounded by another multicollision limit function which uniformly distributes the balls over the bins:

Theorem 4.1. *Let $r, c, q, w \in \mathbb{N}$ such that $1 \leq w \leq c$ and $r \geq 1$. Let $r' = r + \log(w+1)$ and $c' = c - \log(w+1)$. Then $\mu_{r',c'}^{q,D_{\mathsf{HW\text{-}nr}}} \leq \mu_{r',c'}^{\alpha q}$ for $\alpha = \binom{w}{\lfloor \frac{w}{2} \rfloor} \frac{e^2(w+1)}{2^w}$.*

4.3 Proof of Theorem 4.1

Consider two ball-and-bins experiments:

1. We throw αq balls into $2^{r'}$ bins according to a uniform distribution with replacement.
2. We throw q balls into $2^{r'}$ bins according to the distribution $D_{\mathsf{HW\text{-}nr}}$ without replacement.

Note that the maximum number of balls in any single bin in Experiment 1 and Experiment 2 are bounded by $\mu_{r',c'}^{\alpha q}$, and $\mu_{r',c'}^{q,D_{\mathsf{HW\text{-}nr}}}$ respectively. Let X_i^{exp1} and X_i^{exp2} denote the number of balls in bin i in the respective experiments, for $1 \leq i \leq 2^{r'}$. Let μ^{exp1} and μ^{exp2} denote the highest number of balls in any single bin in the respective experiments.

To prove the Lemma, we will apply the same strategy as used by Daemen et al. in Section 6.6 of their article [6]. First, we prove that $\mu_{r',c'}^{\alpha q}$ has some threshold t as lower bound. By the pigeonhole principle, there must be a bin in Experiment 1 containing at least $\left\lceil \frac{\alpha q}{2^{r'}} \right\rceil$ balls. Hence, for $t = \left\lceil \frac{\alpha q}{2^{r'}} \right\rceil$, $\mu^{\alpha q}_{r',c'} \geq t$ holds.

Then, we prove that, for all $y \geq t$,

$$\mathbf{Pr}(\mu^{\text{exp1}} > y) \geq \mathbf{Pr}(\mu^{\text{exp2}} > y). \tag{12}$$

This is a useful result because of the following Lemma:

Lemma 4.1. *If* $\boldsymbol{Pr}(\mu^{exp1} > y) \geq \boldsymbol{Pr}(\mu^{exp2} > y)$ *for all* $y \geq t$*, then* $\mu^{q,D_{\mathsf{HW\text{-}nr}}}_{r',c'} \leq \mu^{\alpha q}_{r',c'}$.

Proof.

(1) Assume that $\mathbf{Pr}(\mu^{\text{exp1}} > y) \geq \mathbf{Pr}(\mu^{\text{exp2}} > y)$ for all $y \geq t$.

(2) From (1) it follows that $\mathbf{Pr}(\mu^{\text{exp1}} > y) \geq \mathbf{Pr}(\mu^{\text{exp2}} > y)$ holds for $y = \mu^{\alpha q}_{r',c'}$.

(3) By definition (see Section 2.6), $\mu^{\alpha q}_{r',c'}$ is the smallest number x such that $\mathbf{Pr}(\mu^{\text{exp1}} > x) < \frac{x}{2^{c'}}$.

(4) From (2) and (3) it follows that $\mathbf{Pr}(\mu^{\text{exp1}} > x) \geq \mathbf{Pr}(\mu^{\text{exp2}} > x)$.

(5) From (3) and (4) it follows that $\mathbf{Pr}(\mu^{\text{exp2}} > x) < \frac{x}{2^{c'}}$.

(6) By definition, $\mu^{q,D_{\mathsf{HW\text{-}nr}}}_{r',c'}$ is the smallest number x' such that $\mathbf{Pr}(\mu^{\text{exp2}} > x') < \frac{x'}{2^{c'}}$.

(7) From (5) and (6) it follows that $x' \leq x$.

(8) From (3), (6) and (7) we can conclude that $\mu^{q,D_{\mathsf{HW\text{-}nr}}}_{r',c'} \leq \mu^{\alpha q}_{r',c'}$. □

By Lemma 4.1, Theorem 4.1 holds under the condition that $\mathbf{Pr}(\mu^{\text{exp1}} > y) \geq \mathbf{Pr}(\mu^{\text{exp2}} > y)$ for all $y \geq t$. This condition is satisfied if $\mathbf{Pr}(X_i^{\text{exp1}} > y) \geq \mathbf{Pr}(X_i^{\text{exp2}} > y)$ for all $y \geq t$ and for all bins i; if each bin i in Experiment 1 is at least as likely to contain more than y balls as each bin i in Experiment 2, then certainly the single bin with the most balls in Experiment 1 is at least as likely to contain more than y balls as the single bin with the most balls in Experiment 2.

In turn, this new condition is satisfied if $\mathbf{Pr}(X_i^{\text{exp1}} = y) \geq \mathbf{Pr}(X_i^{\text{exp2}} = y)$ for all $y \geq t$ and for all bins i; if each bin i in Experiment 1 is at least as likely to contain y balls as each bin i in Experiment 2, for all $y \in t$, then each bin i in Experiment 2 is also at least as likely to contain more than y balls as each bin in Experiment 2.

Therefore, to show that the theorem holds, it remains to be proven that, for all $y \geq t$ and for all bins i, $\mathbf{Pr}(X_i^{\text{exp1}} = y) \geq \mathbf{Pr}(X_i^{\text{exp2}} = y)$.

To show that this holds, we first give definitions for $\mathbf{Pr}(X_i^{\text{exp1}} = y)$ and $\mathbf{Pr}(X_i^{\text{exp2}} = y)$.

Let $j = i \bmod (w+1)$. Then $\mathbf{Pr}(X_i^{\text{exp1}} = y)$ and $\mathbf{Pr}(X_i^{\text{exp2}} = y)$ are defined as follows:

$$\mathbf{Pr}(X_i^{\text{exp1}} = y) = \binom{\alpha q}{y}(2^{-r'})^y(1-2^{-r'})^{\alpha q - y},$$

$$\mathbf{Pr}(X_i^{\text{exp2}} = y) = \begin{cases} 0 & \text{if } y > \binom{w}{j}2^{c-w} \\ \binom{q}{y}\frac{\left(\binom{w}{j}2^{c-w}\right)_y\left(2^b-\binom{w}{j}2^{c-w}\right)_{q-y}}{\left(2^b\right)_q} & \text{if } y \le \binom{w}{j}2^{c-w} \end{cases}.$$

Now, we explain the reason why these probabilities correctly model the experiments, before showing that the inequality given in (12) holds.

4.3.1 Probability in Experiment 1

For Experiment 1, the probability of a single ball falling in bin i is $2^{-r'}$, and y balls need to fall in this bin. The probability of a single ball ending up in any bin except bin i is $1 - 2^{-r'}$, and this needs to occur for the remaining $\alpha q - y$ balls. Finally, there are $\binom{\alpha q}{y}$ ways to choose the y balls which fall in bin i.

4.3.2 Probability in Experiment 2

For Experiment 2, since there are $(w+1)\cdot 2^r$ bins in total, there are 2^r bins per Hamming weight value $0, \ldots, w$. To model this, we let the Hamming weight value of bin i be $j = i \bmod (w+1)$. Note that each b-bit value only belongs in one of the bins, and each of these values represents a ball. For each bin i, there are $\binom{w}{j}$ w-bit values with the correct Hamming weight, the r left-most bits are fixed to one value and there are 2^{b-r-w} possible values for the remaining $b-r-w$ bits. Therefore, there are a total of $\binom{w}{j}2^{b-r-w} = \binom{w}{j}2^{c-w}$ balls which belong to bin i and $2^b - \binom{w}{j}2^{c-w}$ balls which do not belong to bin i.

Firstly, we consider the case that y is greater than $\binom{w}{j}2^{c-w}$, the amount of balls that fit in bin i. In this case, the probability of bin i containing y balls is zero.

Secondly, we consider the case that y is less than or equal to the amount of balls that fit in bin i. In this case, y balls need to fall in bin i and $q-y$ in the other bins, and there are $\binom{q}{y}$ ways to choose the y balls which fall in bin i.

Because the balls are sampled without replacement, the probabilities change with each ball thrown, since the total amount of balls to sample from and the amount of space left in one of the bins decrease with each ball thrown. To account for this, we can use the falling factorial.

The term $\left(\binom{w}{j}2^{c-w}\right)_y$ models the fact that before any balls are thrown in i, there are $\binom{w}{j}2^{c-w}$ balls which can still fall in it; each time a ball is thrown in this bin, the amount of balls which can still fall in it decreases by 1, until all y balls which are supposed to fall in it are thrown.

Similarly, the term $\left(2^b - \binom{w}{j}2^{c-w}\right)_{q-y}$ models the fact that before any balls are thrown in bins other than bin i, there are $2^b - \binom{w}{j}2^{c-w}$ balls which can still fall in them; each time a ball is thrown in one of these bins, the amount of balls which can still fall in them is decreased by 1, until all $q-y$ which are supposed to fall in bins other than bin i are thrown.

Finally, the term $\left(2^b\right)_q$ models the fact that before any balls are thrown in the bins, there are 2^b balls to choose from, and each time a ball is thrown the amount of balls to choose from decreases by 1, until all q balls are thrown.

4.3.3 Proving the inequality between the probabilities

Now, we will show that (12) holds, i.e. that $\mathbf{Pr}(X_i^{\text{exp1}} = y) \geq \mathbf{Pr}(X_i^{\text{exp2}} = y)$ for all bins i. Note that this inequality would trivially hold in the case that y is greater than the capacity of the bin in Experiment 2. In this case $\mathbf{Pr}(X_i^{\text{exp2}} = y)$ is equal to 0, while $\mathbf{Pr}(X_i^{\text{exp1}} = y)$ is always at least 0. Therefore, it only remains to prove the inequality in the case that y is at most the capacity of the bin. By substituting the definitions given above in (12), we get the following inequality:

$$\binom{\alpha q}{y}(2^{-r'})^y(1-2^{-r'})^{\alpha q - y} \overset{?}{\geq} \binom{q}{y}\frac{\left(\binom{w}{j}2^{c-w}\right)_y \left(2^b - \binom{w}{j}2^{c-w}\right)_{q-y}}{\left(2^b\right)_q}. \quad (13)$$

Now, we will prove that (13) holds, using the assumptions in Theorem 4.1 ($1 \leq w \leq c$ and $r \geq 1$).

Proof. Using that $\binom{n}{r} = \frac{n!}{r!(n-r)!} = \frac{(n)_r}{r!}$, we get:

$$\frac{(\alpha q)_y}{y!}(2^{-r'})^y(1-2^{-r'})^{\alpha q - y} \overset{?}{\geq} \frac{(q)_y}{y!}\frac{\left(\binom{w}{j}2^{c-w}\right)_y \left(2^b - \binom{w}{j}2^{c-w}\right)_{q-y}}{\left(2^b\right)_q},$$

$$\frac{(\alpha q)_y}{(q)_y}(2^{-r'})^y(1-2^{-r'})^{\alpha q - y} \overset{?}{\geq} \frac{\left(\binom{w}{j}2^{c-w}\right)_y \left(2^b - \binom{w}{j}2^{c-w}\right)_{q-y}}{\left(2^b\right)_q}.$$

Using that $b = r' + c' \implies -r' = c' - b$, we get:

$$\frac{(\alpha q)_y}{(q)_y}(2^{c'-b})^y(1-2^{c'-b})^{\alpha q - y} \overset{?}{\geq} \frac{\left(\binom{w}{j}2^{c-w}\right)_y \left(2^b - \binom{w}{j}2^{c-w}\right)_{q-y}}{\left(2^b\right)_q},$$

$$\frac{(\alpha q)_y}{(q)_y}\left(\frac{2^{c'}}{2^b}\right)^y\left(1-\frac{2^{c'}}{2^b}\right)^{\alpha q-y} \overset{?}{\geq} \frac{\left(\binom{w}{j}2^{c-w}\right)_y\left(2^b-\binom{w}{j}2^{c-w}\right)_{q-y}}{\left(2^b\right)_y\left(2^b-y\right)_{q-y}},$$

$$\frac{(\alpha q)_y}{(q)_y}\left(\frac{2^{c'}}{2^b}\right)^y\left(\frac{2^b}{2^b}-\frac{2^{c'}}{2^b}\right)^{\alpha q-y} \overset{?}{\geq} \frac{\left(\binom{w}{j}2^{c-w}\right)_y}{\left(2^b\right)_y}\frac{\left(2^b-\binom{w}{j}2^{c-w}\right)_{q-y}}{\left(2^b-y\right)_{q-y}},$$

$$\frac{(\alpha q)_y}{(q)_y}\left(\frac{2^{c'}}{2^b}\right)^y\left(\frac{2^b-2^{c'}}{2^b}\right)^{\alpha q-y} \overset{?}{\geq} \frac{\left(\binom{w}{j}2^{c-w}\right)_y}{\left(2^b\right)_y}\frac{\left(2^b-\binom{w}{j}2^{c-w}\right)_{q-y}}{\left(2^b-y\right)_{q-y}},$$

$$\frac{(\alpha q)_y}{(q)_y}\left(\frac{2^{c'}}{2^b}\right)^y\left(\frac{2^b-2^{c'}}{2^b}\right)^{\alpha q}\left(\frac{2^b}{2^b-2^{c'}}\right)^y \overset{?}{\geq} \frac{\left(\binom{w}{j}2^{c-w}\right)_y}{\left(2^b\right)_y}\frac{\left(2^b-\binom{w}{j}2^{c-w}\right)_{q-y}}{\left(2^b-y\right)_{q-y}},$$

$$\frac{(\alpha q)_y}{(q)_y}\left(\frac{2^{c'}}{2^b-2^{c'}}\right)^y\frac{\left(2^b\right)_y}{\left(\binom{w}{j}2^{c-w}\right)_y}\frac{\left(2^b-y\right)_{q-y}}{\left(2^b-\binom{w}{j}2^{c-w}\right)_{q-y}} \overset{?}{\geq} \left(\frac{2^b}{2^b-2^{c'}}\right)^{\alpha q}.$$

It holds that:

1. $2^b \geq \binom{w}{j}2^{c-w}$ since $2^b > 2^c = 2^w \cdot 2^{c-w} = \left(\sum_{k=0}^{w}\binom{w}{k}\right)\cdot 2^{c-w} > \binom{w}{j}2^{c-w}$,
2. $2^b - y \geq 2^b - \binom{w}{j}2^{c-w}$ because $y \leq \binom{w}{j}2^{c-w}$

Hence, we can use that $\frac{(A)_x}{(B)_x} \geq \left(\frac{A}{B}\right)^x$ if $A \geq B$ (see Lemma 2.1) to get:

$$\left(\frac{\alpha q}{q}\right)^y\left(\frac{2^{c'}}{2^b-2^{c'}}\right)^y\left(\frac{2^b}{\binom{w}{j}2^{c-w}}\right)^y\left(\frac{2^b-y}{2^b-\binom{w}{j}2^{c-w}}\right)^{q-y} \overset{?}{\geq} \left(\frac{2^b}{2^b-2^{c'}}\right)^{\alpha q},$$

$$\alpha^y\left(\frac{2^{c'}}{2^b-2^{c'}}\right)^y\left(\frac{2^b}{\binom{w}{j}2^{c-w}}\right)^y\left(\frac{2^b-\binom{w}{j}2^{c-w}}{2^b-y}\right)^y \overset{?}{\geq} \left(\frac{2^b}{2^b-2^{c'}}\right)^{\alpha q}\left(\frac{2^b-\binom{w}{j}2^{c-w}}{2^b-y}\right)^q,$$

$$\left(\alpha\frac{2^{c'}}{2^b-2^{c'}}\frac{2^b}{\binom{w}{j}2^{c-w}}\frac{2^b-\binom{w}{j}2^{c-w}}{2^b-y}\right)^y \overset{?}{\geq} \left(\frac{2^b}{2^b-2^{c'}}\right)^{\alpha q}\left(\frac{2^b-\binom{w}{j}2^{c-w}}{2^b-y}\right)^q.$$

Using that $b = r' + c'$ and $b = r + c$, we get:

$$\left(\alpha\frac{1}{2^{r'}-1}\frac{1}{\binom{w}{j}2^{-r-w}}\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}}\right)^y \overset{?}{\geq} \left(\frac{1}{1-2^{-r'}}\right)^{\alpha q}\left(\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}}\right)^q,$$

$$\left(\frac{\alpha}{2^{r'}-1}\frac{1}{\binom{w}{j}2^{-(r+w)}}\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}}\right)^y \overset{?}{\geq} \left(\frac{1^\alpha}{(1-2^{-r'})^\alpha}\right)^q\left(\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}}\right)^q,$$

$$\left(\frac{\alpha}{2^{r'}-1}\frac{2^{r+w}}{\binom{w}{j}}\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}}\right)^y \overset{?}{\geq} \left(\frac{1-\binom{w}{j}2^{-r-w}}{(1-2^{-r'})^\alpha\left(1-\frac{y}{2^b}\right)}\right)^q. \tag{14}$$

To further simplify the above inequality (14), we first need to show that the base of the left-hand side is at least 1:

$$\frac{\alpha}{2^{r'}-1}\frac{2^{r+w}}{\binom{w}{j}}\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}} \overset{?}{\geq} 1. \tag{15}$$

Recall that y is the amount of bins in bin i, so $y \geq 0$. We know that the left-hand side of (15) is minimal when $y = 0$. Therefore, if we can prove that (15) holds for $y = 0$, we know that it must hold for every possible value of y. We substitute $y = 0$ in (15):

$$\begin{aligned}
\frac{\alpha}{2^{r'}-1}\frac{2^{r+w}}{\binom{w}{j}}\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{0}{2^b}} &\overset{?}{\geq} 1,\\
\frac{\alpha}{2^{r'}-1}\frac{2^{r+w}}{\binom{w}{j}}\left(1-\binom{w}{j}2^{-r-w}\right) &\overset{?}{\geq} 1,\\
\frac{\alpha}{2^{r'}-1}\left(\frac{2^{r+w}}{\binom{w}{j}}-\frac{\binom{w}{j}}{\binom{w}{j}}\right) &\overset{?}{\geq} 1,\\
\frac{\alpha}{2^{r'}-1}\frac{2^{r+w}-\binom{w}{j}}{\binom{w}{j}} &\overset{?}{\geq} 1,\\
\frac{\alpha}{2^{r'}-1} &\overset{?}{\geq} \frac{\binom{w}{j}}{2^{r+w}-\binom{w}{j}},\\
\alpha &\overset{?}{\geq} \frac{2^{r'}\binom{w}{j}-\binom{w}{j}}{2^{r+w}-\binom{w}{j}},\\
\alpha &\overset{?}{\geq} \frac{2^{r+\log(w+1)}\binom{w}{j}-\binom{w}{j}}{2^{r+w}-\binom{w}{j}},\\
\alpha &\overset{?}{\geq} \frac{2^{r}(w+1)\binom{w}{j}-\binom{w}{j}}{2^{r+w}-\binom{w}{j}}.
\end{aligned}$$

Therefore, the base of the left-hand side of (14) is at least 1 under the condition that:

$$\alpha \geq \frac{2^{r}(w+1)\binom{w}{j}-\binom{w}{j}}{2^{r+w}-\binom{w}{j}}. \tag{16}$$

We show that this condition holds later in the proof.

Now, it suffices to show that (14) holds for $y = t = \left\lceil \frac{\alpha q}{2^{r'}} \right\rceil$ as left-hand side exponent instead of for all $y \geq t$. This is because the left-hand side of (14) is minimal for $y = t$ if the base of the left-hand side is at least 1, and we have just shown that this is the case if $\alpha \geq \frac{2^r(w+1)\binom{w}{j}-\binom{w}{j}}{2^r 2^w-\binom{w}{j}}$.

The demonstration of (14) is quite elaborate, and has been placed in Appendix A to improve the readability of the thesis. The result of this demonstration is that the following condition on α must be satisfied for (14) to hold:

$$\alpha \geq \binom{w}{j}\frac{e^2(w+1)}{2^w}. \tag{17}$$

Now, to show that (13) holds, it remains to be proven that α satisfies the two conditions given in (16) and (17). We do this by first showing that the first condition is satisfied if the second condition is satisfied, and then showing that the second condition is satisfied.

If the second condition is satisfied, the first condition must also be satisfied because the lower bound on α in the first condition is smaller than or equal to the lower bound on α in the second condition:

$$\begin{aligned}
\frac{2^r(w+1)\binom{w}{j} - \binom{w}{j}}{2^{r+w} - \binom{w}{j}} &\overset{?}{\leq} \binom{w}{j}\frac{e^2(w+1)}{2^w}, \\
\frac{2^r(w+1) - 1}{2^{r+w} - \binom{w}{j}} &\overset{?}{\leq} \frac{e^2(w+1)}{2^w}, \\
2^{r+w}(w+1) - 2^w &\overset{?}{\leq} 2^{r+w}e^2(w+1) - \binom{w}{j}e^2(w+1), \\
-2^w &\overset{?}{\leq} 2^{r+w}e^2(w+1) - \binom{w}{j}e^2(w+1) - 2^{r+w}(w+1), \\
2^w &\overset{?}{\geq} \binom{w}{j}e^2(w+1) + 2^{r+w}(w+1) - 2^{r+w}e^2(w+1), \\
2^w &\overset{?}{\geq} (w+1)\left(\binom{w}{j}e^2 + 2^{r+w} - 2^{r+w}e^2\right), \\
\frac{2^w}{w+1} &\overset{?}{\geq} \binom{w}{j}e^2 + 2^{r+w}(1-e^2). \qquad (18)
\end{aligned}$$

Because we assumed that $r \geq 1$ and $w \geq 1$, we know that the term $2^{r+w}(1-e^2)$ is negative since 2^{r+w} is positive and $1-e^2$ is negative. Therefore, the right-hand side of the above inequality (18) is largest when $r = 1$. It follows that we can substitute $r = 1$ in (18), because if it holds for $r = 1$, then it must certainly hold for all $r \geq 1$:

$$\begin{aligned}
\frac{2^w}{w+1} &\overset{?}{\geq} \binom{w}{j}e^2 + 2^{1+w}(1-e^2), \\
\frac{2^w}{w+1} &\overset{?}{\geq} \binom{w}{j}e^2 + 2^w(2(1-e^2)), \\
\frac{2^w}{w+1} &\overset{?}{\geq} \binom{w}{j}e^2 + 2^w(2-2e^2).
\end{aligned}$$

We use that $\binom{w}{j} < \sum_{k=0}^{w}\binom{w}{k} = 2^w$ to replace $\binom{w}{j}$ with 2^w:

$$\begin{aligned}
\frac{2^w}{w+1} &\overset{?}{\geq} 2^w e^2 + 2^w(2-2e^2), \\
\frac{2^w}{w+1} &\overset{?}{\geq} 2^w(2-e^2).
\end{aligned}$$

Because $w \geq 1$, the left-hand side of this inequality is positive. Since $2 \leq e^2$, the right-hand side of this inequality is negative. Therefore, the inequality holds, and we have shown that, if the second condition on α given in (17) is satisfied, then the first condition on α given in (16) must also be satisfied.

Now, it only remains to be shown that the second condition on α is satisfied. Hence, we must show that α is greater than or equal to $\binom{w}{j}\frac{e^2(w+1)}{2^w}$. Because $\binom{w}{j}$ is maximal for $j = \lfloor \frac{w}{2} \rfloor$, it suffices to show that α is greater than or equal to $\binom{w}{\lfloor \frac{w}{2} \rfloor}\frac{e^2(w+1)}{2^w}$, which is the case because $\alpha = \binom{w}{\lfloor \frac{w}{2} \rfloor}\frac{e^2(w+1)}{2^w}$.

We have now shown that the second condition on α is satisfied. It follows that the first condition on α is also satisfied. Because both conditions on α are satisfied, we have proven that (13) holds under the assumptions that $1 \leq w \leq c$ and $r \geq 1$. ☐

Because we have now proven that (13) holds, assuming that $1 \leq w \leq c$ and $r \geq 1$, it follows from Lemma 4.1 that Theorem 4.1 is proven.

4.4 Applying Theorem 4.1 to the SuKS bound

We can use Theorem 4.1 to tighten the NALR-PRF security bound on the SuKS (see (9) on page 15) when only considering Hamming weight leakage:

Corollary 4.1. *The tight NALR-PRF security bound on the SuKS when only considering the leakage function L_p^{HW} defined in* (11) *on page 24 is*

$$\boldsymbol{Adv}_F^{\text{nalr-prf}}(\mathcal{A}) \leq \frac{2N^2}{2^c} + \frac{\mu_{s,b-s}^{2(N-q)}}{2^{b-s}} + \frac{\mu_{b-s,s}^{2(N-q)} \cdot N}{2^{\min\{\delta,\varepsilon\} - \mu_{s,b-s}^{2(N-q)}\lambda}} + \frac{\mu_{t+\lambda,b-t-\lambda}^{\alpha q} \cdot N}{\binom{w}{\ell} 2^{b-t-w}} \quad (19)$$

with $\lambda = \lceil \log(w+1) \rceil$, $\alpha = \binom{w}{\lfloor \frac{w}{2} \rfloor}\frac{e^2(w+1)}{2^w}$ and ℓ the leakage value from L_p^{HW} which results in the largest multicollision estimated by $\mu_{t+\lambda,b-t-\lambda}^{\alpha q}$.

Proof. The only term in this bound which has changed with respect to the NALR-PRF security bound in (9) is the last term, since this the only term in the NALR-PRF security bound which is not tight. To prove the last term in (19) is tight, we give a matching attack for this term. This matching attack is similar to the attack given in Section 3.3. We assume that the function G in the SuKS is an XOR and the attacker tries to recover the state V in order to find the secret key K:

1. The attacker tries q distinct plaintexts to find multicollisions for T and $\mathrm{HW}(\mathrm{right}_w(W))$. It follows from Theorem 4.1 that the size of the largest multicollision found by the attacker is upper bounded by $\mu_{t+\lambda,b-t-\lambda}^{\alpha q}$. Let the leakage value $\mathrm{HW}(\mathrm{right}_w(W))$ of this multicollision be called ℓ.

2. In order to recover the state V, the attacker must first find the state W. The attacker knows $\text{left}_t(W) = T$ and has to guess $\text{right}_{b-t}(W)$. Due to the leakage, the attacker also learns $\text{HW}(\text{right}_w(W)) = \ell$, which means $\text{right}_{b-t}(W)$ can be any of $\binom{w}{\ell}$ values. The attacker has no information about the remaining $b - t - w$ bits, so these bits can be any of 2^{b-t-w} values. It follows that the attacker has to guess between $\binom{w}{\ell}2^{b-t-w}$ values.

   The probability that the attacker guesses correctly with one guess is $\frac{\mu^{\alpha q}_{t+\lambda,b-t-\lambda}}{\binom{w}{\ell}2^{b-t-w}}$, because each element in the multicollision has a different value for $\text{right}_{b-t}(W)$, and the attacker only needs to find one of these values. The probability that the attacker guesses correctly with N guesses is $\frac{\mu^{\alpha q}_{t+\lambda,b-t-\lambda}\cdot N}{\binom{w}{\ell}2^{b-t-w}}$.

3. If the attacker guesses $\text{right}_{b-t}(W)$ correctly, they can call the inverse primitive p^{-1} on W to find V, and compute $K = \text{left}_k(U) \oplus \text{left}_k(V)$ (U is known to the attacker because no secret information is involved in computing it). Therefore, the success probability of this attack is $\frac{\mu^{\alpha q}_{t+\lambda,b-t-\lambda}\cdot N}{\binom{w}{\ell}2^{b-t-w}}$.

In conclusion, we have proven that the last term in (19) is tight by giving a matching attack. Because the other terms were already proven to be tight, Corollary 4.1 is proven. □

4.4.1 Limitation of the tightened NALR-PRF security bound

One limitation of the bound given in (19) is that it uses ℓ, the leakage value which results in the largest multicollision, in the fourth term of the bound. The value of ℓ depends on what plaintexts the attacker queries to find the multicollisions, and the success probability of the attacker is dependent on the value of ℓ; the attacker's success probability is highest when $\ell = \lfloor \frac{w}{2} \rfloor$ or $\ell = \lceil \frac{w}{2} \rceil$, and it is lowest when $\ell = 0$ or $\ell = w$, because these values for ℓ result respectively in the maximal and minimal value for $\binom{w}{\ell}$.

To improve the bound, the fourth term could be split into two terms. In the first term, it would be assumed that ℓ can only attain values which are inconvenient for the attacker, meaning they result in a relatively low success probability, and then ℓ could be removed from the term by substituting the inconvenient value which results in the highest success probability. In the second term, it would be assumed that, if the value of ℓ is convenient for the attacker, the attacker's success probability is 1. Therefore, in this second term, the attacker's success probability can be described by the probability that ℓ has a convenient value for the attacker.

Chapter 5

Conclusions

In conclusion, we have shown that the NALR-PRF security bound on the Suffix Keyed Sponge is not tight. In Chapter 3, we have analysed the tightness of the bound and come to the conclusion that the bound can be tightened by taking into account that the leakage influences the size of the largest multicollision in one of the matching attacks for the SuKS. Furthermore, in this chapter we have tightened the bound on the NALR-PRF security when only considering leakage functions which leak λ bits of the SuKS state. In Chapter 4, we have given a tightened bound when only considering leakage functions which leak the Hamming weight of w bits of the SuKS state by incorporating this leakage into the multicollision limit function used to estimate the size of the largest multicollision.

There are still numerous opportunities for doing research on the tightness of the NALR-PRF security bound on the SuKS. For example, the tightened bound on the NALR-PRF security of the SuKS when considering Hamming weight leakage could be improved by removing the leakage value from the bound, as described in Section 4.4.1. Furthermore, tight bounds could be given when considering other types of leakage functions. Finally, the NALR-PRF security bound on the SuKS could be tightened such that the bound holds for all possible leakage functions.

Bibliography

[1] ISAP - Specification. `https://isap.iaik.tugraz.at/specification.html`. Accessed: 4-12-2022.

[2] Lightweight Cryptography - CSRC. `https://csrc.nist.gov/projects/lightweight-cryptography`. Accessed: 4-12-2022.

[3] Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process. `https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/final-lwc-submission-requirements-august2018.pdf`. Accessed: 4-12-2022.

[4] Mihir Bellare, Oded Goldreich, and Anton Mityagin. The Power of Verification Queries in Message Authentication and Authenticated Encryption. Cryptology ePrint Archive, Paper 2004/309, 2004. `https://eprint.iacr.org/2004/309`.

[5] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Sponge functions. *ECRYPT Hash Workshop 2007*, May 2007.

[6] Joan Daemen, Bart Mennink, and Gilles Van Assche. Full-State Keyed Duplex With Built-In Multi-User Support. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 606–637. Springer International Publishing, 2017.

[7] Christoph Dobraunig and Bart Mennink. Security of the Suffix Keyed Sponge. *IACR Transactions on Symmetric Cryptology*, 2019(4):223–248, January 2020.

[8] Christoph Dobraunig and Bart Mennink. Tightness of the Suffix Keyed Sponge. *IACR Transactions on Symmetric Cryptology*, 2020(4):195–212, December 2020.

[9] Sébastien Duval, Pierrick Méaux, Charles Momin, and François-Xavier Standaert. Exploring Crypto-Physical Dark Matter and Learning with Physical Rounding Towards Secure and Efficient Fresh Re-Keying.

IACR Transactions on Cryptographic Hardware and Embedded Systems, 2021(1):373–401, December 2020.

[10] Peter Gaži, Krzysztof Pietrzak, and Stefano Tessaro. The Exact PRF Security of Truncation: Tight Bounds for Keyed Sponges and Truncated CBC. In *Advances in Cryptology – CRYPTO 2015*, pages 368–387, August 2015.

[11] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. On the Cryptographic Applications of Random Functions. In *Advances in Cryptology – CRYPTO 1984*, pages 276–288, 08 1984.

[12] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards.* 2007.

[13] Rita Mayer-Sommer. Smartly Analyzing the Simplicity and the Power of Simple Power Analysis on Smartcards. In Çetin K. Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2000*, pages 78–92. Springer Berlin Heidelberg, 2000.

Appendix A

Appendix

In this appendix, we show in detail that (14) given on page 30 holds for $y = \left\lceil \frac{\alpha q}{2^{r'}} \right\rceil$ as left-hand side exponent. Therefore, we show that

$$\left(\frac{\alpha}{2^{r'} - 1} \frac{2^{r+w}}{\binom{w}{j}} \frac{1 - \binom{w}{j} 2^{-r-w}}{1 - \frac{y}{2^b}} \right)^{\left\lceil \frac{\alpha q}{2^{r'}} \right\rceil} \geq \left(\frac{1 - \binom{w}{j} 2^{-r-w}}{(1 - 2^{-r'})^{\alpha} \left(1 - \frac{y}{2^b}\right)} \right)^{q} .$$

Showing that this inequality holds is a step in the proof of (13) on page 29, which in turn is a step in the proof of Theorem 4.1 on page 26.

$$\left(\frac{\alpha}{2^{r'}-1}\frac{2^{r+w}}{\binom{w}{j}}\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}}\right)^{\left\lceil\frac{\alpha q}{2^{r'}}\right\rceil} \overset{?}{\geq} \left(\frac{1-\binom{w}{j}2^{-r-w}}{(1-2^{-r'})^{\alpha}\left(1-\frac{y}{2^b}\right)}\right)^{q},$$

$$\left(\frac{\alpha}{2^{r'}-1}\frac{2^{r+w}}{\binom{w}{j}}\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}}\right)^{\frac{\alpha q}{2^{r'}}} \overset{?}{\geq} \left(\frac{1-\binom{w}{j}2^{-r-w}}{(1-2^{-r'})^{\alpha}\left(1-\frac{y}{2^b}\right)}\right)^{q},$$

$$\left(\left(\frac{\alpha}{2^{r'}-1}\frac{2^{r+w}}{\binom{w}{j}}\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}}\right)^{\frac{\alpha q}{2^{r'}}}\right)^{\frac{2^{r'}}{\alpha q}} \overset{?}{\geq} \left(\left(\frac{1-\binom{w}{j}2^{-r-w}}{(1-2^{-r'})^{\alpha}\left(1-\frac{y}{2^b}\right)}\right)^{q}\right)^{\frac{2^{r'}}{\alpha q}},$$

$$\frac{\alpha}{2^{r'}-1}\frac{2^{r+w}}{\binom{w}{j}}\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}} \overset{?}{\geq} \left(\frac{1-\binom{w}{j}2^{-r-w}}{(1-2^{-r'})^{\alpha}\left(1-\frac{y}{2^b}\right)}\right)^{\frac{2^{r'}}{\alpha}},$$

$$\log\left(\frac{\alpha}{2^{r'}-1}\frac{2^{r+w}}{\binom{w}{j}}\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}}\right) \overset{?}{\geq} \log\left(\left(\frac{1-\binom{w}{j}2^{-r-w}}{(1-2^{-r'})^{\alpha}\left(1-\frac{y}{2^b}\right)}\right)^{\frac{2^{r'}}{\alpha}}\right),$$

$$\log\left(\frac{\alpha}{2^{r'}-1}\right)+\log\left(\frac{2^{r+w}}{\binom{w}{j}}\right)+\log\left(\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}}\right) \overset{?}{\geq} \frac{2^{r'}}{\alpha}\log\left(\frac{1-\binom{w}{j}2^{-r-w}}{(1-2^{-r'})^{\alpha}\left(1-\frac{y}{2^b}\right)}\right),$$

$$\log\left(\frac{\alpha}{2^{r'}-1}\right)+\log\left(\frac{2^{r+w}}{\binom{w}{j}}\right)+\log\left(\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}}\right) \overset{?}{\geq} \frac{2^{r'}}{\alpha}\log\left(1-\binom{w}{j}2^{-r-w}\right)-\frac{2^{r'}}{\alpha}\log\left((1-2^{-r'})^{\alpha}\left(1-\frac{y}{2^b}\right)\right),$$

$$\log\left(\frac{\alpha}{2^{r'}-1}\right)+\log\left(\frac{2^{r+w}}{\binom{w}{j}}\right)+\log\left(\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}}\right) \overset{?}{\geq} \frac{2^{r'}}{\alpha}\log\left(1-\binom{w}{j}2^{-r-w}\right)-2^{r'}\log\left(1-2^{-r'}\right)-\frac{2^{r'}}{\alpha}\log\left(1-\frac{y}{2^b}\right),$$

$$\log\left(\frac{\alpha}{2^{r'}-1}\right)+\log\left(\frac{2^{r+w}}{\binom{w}{j}}\right)+2^{r'}\log\left(1-2^{-r'}\right)\overset{?}{\geq}\left(\frac{2^{r'}}{\alpha}-1\right)\log\left(1-\binom{w}{j}2^{-r-w}\right)-\left(\frac{2^{r'}}{\alpha}-1\right)\log\left(1-\frac{y}{2^b}\right)$$
$$\log\left(\frac{\alpha}{2^{r'}-1}\right)+\log\left(\frac{2^{r+w}}{\binom{w}{j}}\right)+2^{r'}\log\left(1-2^{-r'}\right)\overset{?}{\geq}\left(\frac{2^{r'}}{\alpha}-1\right)\left(\log\left(1-\binom{w}{j}2^{-r-w}\right)-\log\left(1-\frac{y}{2^b}\right)\right)$$
$$\log\left(\frac{\alpha}{2^{r'}-1}\right)+\log\left(\frac{2^{r+w}}{\binom{w}{j}}\right)+2^{r'}\log\left(1-2^{-r'}\right)\overset{?}{\geq}\left(\frac{2^{r'}}{\alpha}-1\right)\left(\log\left(\frac{1-\binom{w}{j}2^{-r-w}}{1-\frac{y}{2^b}}\right)\right).$$

We know that $y \leq \binom{w}{j}2^{c-w}$, from which it follows that $1-\frac{y}{2^b} \geq 1-\binom{w}{j}2^{-r-w}$. Therefore, the second term on the right-hand side is at most $\log(1)$:

$$\log\left(\frac{\alpha}{2^{r'}-1}\right)+\log\left(\frac{2^{r+w}}{\binom{w}{j}}\right)+2^{r'}\log\left(1-2^{-r'}\right)\overset{?}{\geq}\left(\frac{2^{r'}}{\alpha}-1\right)\log(1),$$
$$\log\left(\frac{\alpha}{2^{r'}-1}\right)+\log\left(\frac{2^{r+w}}{\binom{w}{j}}\right)+2^{r'}\log\left(1-2^{-r'}\right)\overset{?}{\geq}0,$$
$$\log\left(\frac{\alpha}{2^{r'}-1}\right)+r+w-\log\left(\binom{w}{j}\right)+2^{r'}\log\left(1-2^{-r'}\right)\overset{?}{\geq}0,$$
$$\log(\alpha)\overset{?}{\geq}\log\left(2^{r'}-1\right)+\log\left(\binom{w}{j}\right)-2^{r'}\log\left(1-2^{-r'}\right)-r-w,$$
$$2^{\log(\alpha)}\overset{?}{\geq}2^{\log\left(2^{r'}-1\right)+\log\left(\binom{w}{j}\right)-2^{r'}\log\left(1-2^{-r'}\right)-r-w},$$
$$\alpha\overset{?}{\geq}\frac{(2^{r'}-1)\binom{w}{j}}{(1-2^{-r'})^{2^{r'}}2^{r+w}}.$$

Using that $2^{r'} - 1 < 2^{r'}$, we can replace $2^{r'} - 1$ by $2^{r'}$ on the right-hand side of the inequality:

$$\alpha \overset{?}{\geq} \frac{2^{r'}\binom{w}{j}}{(1-2^{-r'})^{2^{r'}}2^{r+w}},$$
$$\alpha \overset{?}{\geq} \frac{(w+1)2^{r}\binom{w}{j}}{(1-2^{-r'})^{2^{r'}}2^{r+w}},$$
$$\alpha \overset{?}{\geq} \binom{w}{j}\frac{w+1}{\left(1-\frac{1}{2^{r'}}\right)^{2^{r'}}2^{w}}.$$

We assumed that $r \geq 1$. Because $r' = r + \log(w+1) > r$, it follows that $r' > 1$ and that $2^{r'} > 2$. Therefore, we can use that $\left(1-\frac{1}{x}\right)^{x} \geq e^{-\frac{x}{x-1}}$ for all $x \in \mathbb{R} \setminus \{1\}$ (see Lemma 2.3 on page 7) by substituting $x = 2^{r'}$ to get:

$$\alpha \overset{?}{\geq} \binom{w}{j}\frac{w+1}{e^{-\frac{2^{r'}}{2^{r'}-1}}2^{w}},$$
$$\alpha \overset{?}{\geq} \binom{w}{j}\frac{e^{\frac{2^{r'}}{2^{r'}-1}}(w+1)}{2^{w}}.$$

Using that $e^{\frac{2^{x}}{2^{x}-1}} \leq e^{2}$ on the interval $[1,\infty)$ (see Lemma 2.4 on page 8), and that $r' > 1$, we can substitute $x = r'$ to get:

$$\alpha \overset{?}{\geq} \binom{w}{j}\frac{e^{2}(w+1)}{2^{w}}.$$

From this demonstration, we can conclude that (13) holds under the condition that α is greater than or equal to $\binom{w}{j}\frac{e^2(w+1)}{2^w}$. This condition is taken into account in the rest of the proof of Theorem 4.1.