The road from Panama to Keccak via RadioGatún

Guido Bertoni[1], Joan Daemen[1], Michaël Peeters[2] and Gilles Van Assche[1]

[1] STMicroelectronics
[2] NXP Semiconductors

Abstract. In this paper, we explain the design choices of Panama [8] and RadioGatún [1], which lead to Keccak [3]. After a brief recall of Panama, RadioGatún and the trail backtracking cost, we focus on three important aspects. First, we explain the role of the belt in the light of differential trails. Second, we discuss the relative advantages of a block mode hash function compared to a stream mode one. Finally, we point out why Panama and RadioGatún are not sponge functions [2] and why their design philosophy differs from that of Keccak.

1 Introduction

After the cryptanalysis of SHA-1 by Wang et al. in 2005 [11], we decided to continue the exploration of the design of hash functions along the lines of Panama and its predecessors. We start by describing Panama as the starting point of our study. We then recall the trail backtracking cost and the design decisions behind RadioGatún.

1.1 Panama

Panama is a cryptographic hash function designed by Joan Daemen and Craig Clapp in 1998 [8]. As depicted on Figure 1, the state of Panama is composed of two parts:

- the *mill* (originally called state), which is composed of 17 words (a_i, $i = 0 \dots 16$) of 32 bits each, and
- the *belt* (originally called buffer), which is composed of 8×32 words.

Between each round, 8 words of input are XORed into the mill and into the belt, as indicated on Figure 1. After all the input blocks are processed, 33 blank rounds (i.e., without any input) are performed. Part of the state can then be used as output. Any output length can be generated by further iterating the round function and extracting part of the state at each iteration.

The mill and the belt are processed differently by the round function. The belt undergoes a simple linear transformation, making it operate as a linear feedback shift register (LFSR). In contrast, the mill undergoes a non-linear function $\iota \circ \theta \circ \pi \circ \gamma$ made out of the following components:

Dagstuhl Seminar Proceedings 09031
Symmetric Cryptography
http://drops.dagstuhl.de/opus/volltexte/2009/1958

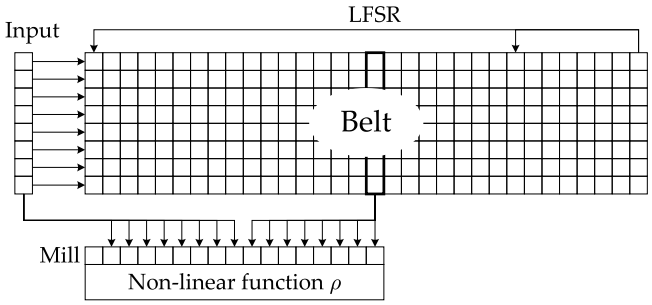

Fig. 1. The structure of PANAMA

- γ is the non-linear step and processes the bits of the words as $a_i \leftarrow a_i + (a_{i+1} + 1)a_{i+2} + 1$;
- π provides dispersion by displacing the words and rotating the bits within each word: $a_i \leftarrow a_{7i} \ggg i(i+1)/2$;
- θ diffuses linearly the words in the mill: $a_i \leftarrow a_i + a_{i+1} + a_{i+4}$;
- ι provides asymmetry among the words.

At each round, 8 words of the belt are XORed into words of the mill. Note that there is no feedback from the mill to the belt in PANAMA.

The belt is designed such that any differential trail is at least 33 rounds long. However, this did not prevent from breaking PANAMA using differential trails [10,7]. The trails used in these attacks have a low backtracking cost, which is defined below.

1.2 The trail backtracking cost

As depicted in Figure 2, a differential trail is a sequence of differences in the state (t') and in the message blocks (p'). It fully defines the differences before and after each round. In order to find a message pair that follows a given trail, the bit values at each round must satisfy conditions that derive from the trail. To satisfy these conditions, the attacker has degrees of freedom coming from the absolute value of the message blocks.

The trail backtracking cost [1] expresses the maximum number of conditions that must be satisfied at a given point in time. We here give a definition that looks different from the original one, but which is nevertheless equivalent.

Let ℓ be the number of message bits at each round; it is also the number of degrees of freedom per round. The trail spans n rounds. Let W_i be the number of conditions at round i, for $0 \leq i \leq n-1$; this is the weight profile of the trail. We then define the following recursion:

- $H_n = 0$,
- $H_i = \max(H_{i+1} + W_i - \ell, 0)$.

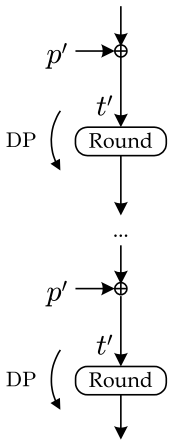

Fig. 2. A differential trail

The value H_i gives the number of conditions at the beginning of round i that cannot be resolved by the degrees of freedom after round i. The backtracking cost is defined as

$$C = \ell + \max_i H_i.$$

i	ℓ	W_i	H_i
0	8	11	5
1	8	10	2
2	8	2	0
3	8	9	**6**
4	8	13	5
5			0

Fig. 3. Example of trail weight profile

In Figure 3, we give the weight profile of an imaginary trail and use it as an example to compute the trail backtracking cost. In this example, the attacker can choose $\ell = 8$ bits at each round. In the end, she just needs to find one pair, so there are no conditions or degrees of freedom left, and we set $H_n = H_5 = 0$. At round 4, there are 13 conditions, 8 of which can be solved using the available degrees of freedom, hence $H_4 = 13 - 8 = 5$. At round 3, 9-8=1 condition that cannot be solved immediately is added, and thus $H_3 = 6$. The backtracking cost $C = H_3 + \ell = 14$ indicates that 14 conditions have to be solved in round 3, so we need 8 degrees of freedom at round 3 plus 6 degrees of freedom coming from rounds 0-2 to expect to be able to satisfy them.

1.3 RadioGatún

RadioGatún is a cryptographic hash function that we presented at the NIST Hash Workshop in 2006 [1]. As depicted in Figure 4, the state of RadioGatún is also composed of a belt and of a mill:

- the *mill* is composed of 19 words of w bits each, and
- the *belt* is composed of 3×13 words.

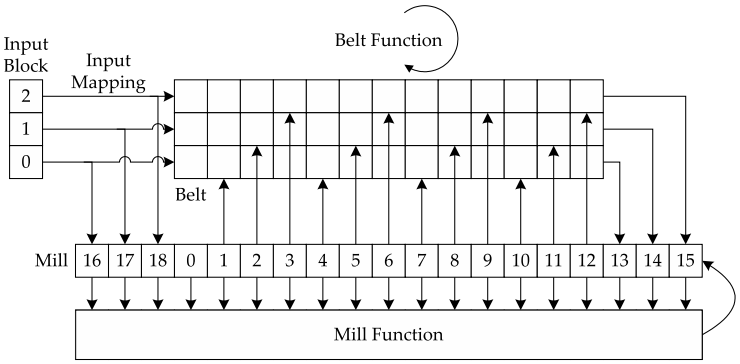

Fig. 4. The structure of RadioGatún

The word size is a parameter w, which can be set to any value between 1 and 64. By default, $w = 64$.

The round function in the mill is very similar to that of Panama, except that the indexes are taken modulo 19 (instead of modulo 17).

RadioGatún was designed as a way to solve the problems on Panama [10,7]. The differences between the two hash functions are the following.

First, RadioGatún modifies the belt and mill structure by adding feedback from the mill to the belt. In Panama, the belt is evolving linearly and independently of the mill (in the absorbing phase); collisions can thus easily be produced in the belt. In RadioGatún, this is no longer possible as the belt receives differences coming from the mill.

Second, the smaller belt size decreases the memory footprint of RadioGatún.

Third, only 3/19 words in the mill can be controlled through the message blocks (instead of 8/17 in Panama). Given a round function, this has a negative impact on performance but a positive one on security as it has a threefold impact on differential trails.

- It gives less freedom for trail construction and will typically increase the minimum weight of trails.

– For the same trail, it increases the value $W_i - \ell$ at each round and hence the backtracking cost of the trail.
– For the same backtracking cost, it increases the number of rounds to which conditions must be transferred.

2 Beyond RadioGatún

In late 2006, we created Gnoblio as a test bench for different configurations. The mill of Gnoblio is composed of 11 words (instead of 17 or 19), which helps obtain results quickly. By default, the round function operating on the mill is identical to that of Panama or RadioGatún, except for its size. The belt is unspecified to allow different belt sizes and feedback configurations. The message blocks contain 1 or 2 words of input, which allows to bracket the input to mill size ratio of RadioGatún: $\frac{1}{11} < \frac{3}{19} < \frac{2}{11}$.

Among other things, the purpose of Gnoblio was to gain more insight in some aspects of belt-and-mill hash functions. We now detail two of these aspects.

2.1 The role of the belt

The role of the belt in Panama is to guarantee a minimal length of trails. Any differential trail is at least 33-round long, due to the linear expansion of the belt and the fact that it evolves independently of the mill.

The role of the belt in RadioGatún is to provide long-term diffusion. A difference in the mill can propagate to the belt and then come back later in the mill through the belt-to-mill feedback.

The impact of the belt on differential trails is more subtle than in Panama. To illustrate this, consider a differential trail as in Figure 5 with differences p'_i in the message blocks, a'_i in the mill and b'_i in the belt. The belt evolves linearly and so do the differences. Hence, the difference in the belt b'_i at round i depends linearly on the message difference and on the differences at round $i-1$: $b'_i = \lambda(a'_{i-1}, b'_{i-1}, p'_i)$.

i	Δ input	Δ mill	Δ belt
0	p'_0	a'_0	$b'_0 = 0$
1	p'_1	a'_1	$b'_1 = \lambda(a'_0, b'_0, p'_1)$
...			
$n-1$	p'_{n-1}	a'_{n-1}	$b'_{n-1} = \lambda(a'_{n-2}, b'_{n-2}, p'_{n-1})$
n	p'_n	$a'_n = 0$	$b'_n = \lambda(a'_{n-1}, b'_{n-1}, p'_n) = 0$

Fig. 5. General structure of a differential trail in a belt and mill hash function

To obtain an internal collision, the last round must specify that $a'_n = 0$ and $b'_n = 0$. The last condition can be replaced by a linear condition on a'_{n-1}, b'_{n-1}

and p'_n. Iteratively, all the linear conditions on the belt differences can be replaced by linear conditions on the message block differences and mill differences. In other words, the trail seen as a sequence of p'_i and a'_i must be compatible with the belt-and-mill structure. Also, the size of the belt determines the number of conditions that $b'_n = 0$ impose on the trail. This somehow imposes a minimum trail length, as a trail must be long enough to satisfy all the conditions.

The number of conditions stemming from $b'_n = 0$ does not depend on n, the length of the trail. For long trails, these conditions are spread among all n steps. In other words, this effect does not scale with the trail length.

While the belt appears to bring added value at a reasonable cost, the restrictions that the belt imposes on differential trails do not scale with the trail length. For Keccak, we have therefore decided to remove the belt and instead increase the number of words in the mill.

2.2 From stream to blocks

With Panama and RadioGatún, one round is processed between two message block insertions. We call this a *stream mode*, as the message blocks are inserted (more or less) continuously into the state.

If instead we would insert two message blocks every two rounds, the performance would essentially be the same. What would be the impact on security?

The number of words inserted has an impact on generic attacks. For a constant state size, increasing the input length means that the state part that is not controlled directly (i.e., the "capacity" in the sponge terminology [2]) shrinks. Depending on the security claim, this may or may not have an impact. For RadioGatún, for instance, this effect would be beyond the claimed security level.

Let us discuss the impact on the trail backtracking cost. For simplicity, we compare the input of ℓ words every round with the input of 2ℓ words every two rounds. We consider in both cases the values of H_{2j} and H_{2j+1}.

In the first case, we get

$$\begin{aligned} H_{2j} &= \max(H_{2j+1} + W_{2j} - \ell, 0) \\ H_{2j+1} &= \max(H_{2j+2} + W_{2j+1} - \ell, 0) \\ &\rightarrow \max(H_{2j+2} + W_{2j+1} + W_{2j} - 2\ell, W_{2j} - \ell, 0), \end{aligned}$$

while in the second case

$$\begin{aligned} H_{2j} &= \max(H_{2j+1} + W_{2j}, 0) \\ H_{2j+1} &= \max(H_{2j+2} + W_{2j+1} - 2\ell, 0) \\ &\rightarrow \max(H_{2j+2} + W_{2j+1} + W_{2j} - 2\ell, W_{2j}, 0). \end{aligned}$$

It is clear that, for the same weight profile, the backtracking cost cannot decrease when increasing the number of rounds between two message block insertions. This was experimentally verified on Gnoblio.

If we extend this reasoning to a larger number of rounds, we tend to a *block mode*, where (relatively) larger message blocks are inserted between the application of a sequences of a (relatively) large number of rounds. The evaluation of the trail backtracking cost works as follows. Consider a sequence of r rounds between two message block insertions. Then $H_r = 0$ and $H_i = \max(H_{i+1}+W_i, 0) = H_{i+1} + W_i$. The trail backtracking cost becomes $C = \sum_i W_i$, where the sum of the weights in the trail becomes a relevant quantity.

In this respect, the advantages of the block mode are the following. First, the entire weight profile counts, as we the backtracking cost essentially depends on the sum of the weights. Second, the attacker can control part of the state only every r rounds. However, a drawback is that trail clustering becomes possible, as different trails can contribute to the same differential.

Another advantage of the block mode has to do with the algebraic degree of the round function. Panama and RadioGatún both use quadratic round functions. In [6], C. Bouillaguet and P.-A. Fouque have shown how to transfer conditions originating from a differential trail to previous rounds using Gröbner bases. This was possible due to the quadratic round function between message block insertion. In this respect, the transformation between two message insertions in a block mode has a higher degree and this in general makes condition transfer harder.

3 Sponge functions?

Sometimes, Panama, RadioGatún or Grindahl [9] are called sponge functions. While these hash functions are similar to sponge functions, we argue here why calling them sponge functions is inappropriate.

Fig. 6. This is not a sponge

As a first example, RadioGatún does not make use of the sponge construction [2], and the differences between the two are the following:

– In the sponge construction, the input is applied to and the output is extracted from the same part of the state. This is in fact essential for the simplicity of the indifferentiability proof of the sponge construction [4]. In RadioGatún this is not the case.

– In RadioGatún, there are a number of blank rounds between the application of the input and the extraction of the output. There are no blank rounds in a sponge function.

Furthermore, RadioGatún predates the sponge construction. In the RadioGatún paper [1] the security claim was expressed with respect to something called an ideal mangling function, which is different from a sponge function. The security claim is now expressed as a flat sponge claim [2], but this does not imply that the function has to follow the sponge construction.

As another example, Grindahl is not a sponge function because the input words overwrite part of the state, it has blank rounds and there is no squeezing defined.

Yet, there are indeed similarities between these hash functions and sponge functions, and it would be easy to slightly modify, e.g., RadioGatún in order to follow the sponge definition. This would be feasible without altering the ideas behind its design. But what would be the point of such a change?

Making RadioGatún fit in the sponge construction would allow to use results on generic attacks against sponge functions. At a rate of $3w$ bits, there remain $55w$ bits that cannot be directly controlled from the attacker, which correspond to the c_{sponge} bits of capacity in a sponge function. Thus one can claim a resistance of the sponge function against generic attacks at $2^{c_{\text{sponge}}/2} = 2^{27.5w}$.

The claimed security of RadioGatún is $2^{c_{\text{claim}}/2} = 2^{9.5w}$, much below the complexity of generic attacks. The gap between c_{sponge} and c_{claim} accounts for the fact that the round function is not designed to be strong by itself and thus the best attacks are clearly non-generic. The design philosophy of RadioGatún is to avoid internal collisions in the absorbing phase and to decorrelate the input blocks with the output blocks using blank rounds. Trying to fit RadioGatún in the sponge framework does not bring any added value, as its security relies globally on the iteration of its round function, not on the strength of the round function alone.

In this respect, Keccak has a fairly different design philosophy and can readily be called a sponge function. It is based on the sponge construction from the start and uses the *hermetic sponge strategy* [5]. The design philosophy consists in instantiating a sponge function by designing the permutation Keccak-f so as to avoid any structural properties with the exception of a compact description. By structural properties we mean properties that a typical random permutation does not have. The sponge construction then provides provable security against all generic attacks.

Thanks to this strategy, there is no gap between the claimed security level and the capacity of the sponge construction used by Keccak, namely, $c_{\text{sponge}} = c_{\text{claim}}$.

4 Conclusion

As a conclusion, we wish to point out a few trends in the evolution of the hash functions considered in the scope of this paper. As illustrated in Figure 7, the

belt has decreased in size in favor of the mill (except for Gnoblio, for which the mill was chosen to be small). Also, the number of input words per round has decreased from Panama to Keccak. Finally, the block mode of Keccak contrasts with the stream modes used by Panama, RadioGatún and Gnoblio.

	Mill size	Belt size	Input/round
Panama	17	256	8/1
RadioGatún	19	39	3/1
Gnoblio	11	variable	$p/1$
Keccak$[r = 1024, c = 576]$	25	0	16/18

Fig. 7. Evolution of the parameters

References

1. G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche, RadioGatún, *a belt-and-mill hash function*, Second Cryptographic Hash Workshop, Santa Barbara, August 2006, `http://radiogatun.noekeon.org/`.
2. ———, *Sponge functions*, Ecrypt Hash Workshop 2007, May 2007, also available as public comment to NIST from `http://www.csrc.nist.gov/pki/HashWorkshop/Public_Comments/2007_May.html`.
3. ———, Keccak *specifications*, NIST SHA-3 Submission, October 2008, `http://keccak.noekeon.org/`.
4. ———, *On the indifferentiability of the sponge construction*, Advances in Cryptology – Eurocrypt 2008 (N. P. Smart, ed.), Lecture Notes in Computer Science, vol. 4965, Springer, 2008, `http://sponge.noekeon.org/`, pp. 181–197.
5. ———, Keccak *sponge function family main document*, NIST SHA-3 Submission (updated), January 2009, `http://keccak.noekeon.org/`.
6. C. Bouillaguet and P.-A. Fouque, *Analysis of the collision resistance of RadioGatún using algebraic techniques*, Selected Areas in Cryptography, Lecture Notes in Computer Science, vol. 4876, Springer, 2008.
7. J. Daemen and G. Van Assche, *Producing collisions for PANAMA, instantaneously*, Fast Software Encryption 2007 (A. Biryukov, ed.), LNCS, Springer-Verlag, 2007, pp. 1–18.
8. J. Daemen and C. S. K. Clapp, *Fast hashing and stream encryption with PANAMA*, Fast Software Encryption 1998 (S. Vaudenay, ed.), LNCS, no. 1372, Springer-Verlag, 1998, pp. 60–74.
9. L. Knudsen, C. Rechberger, and S. Thomsen, *Grindahl - a family of hash functions*, Fast Software Encryption 2007 (A. Biryukov, ed.), LNCS, Springer-Verlag, 2007, pp. 39–47.
10. V. Rijmen, B. Van Rompay, B. Preneel, and J. Vandewalle, *Producing collisions for PANAMA*, Fast Software Encryption 2001 (M. Matsui, ed.), LNCS, no. 2355, Springer-Verlag, 2002, pp. 37–51.
11. X. Wang, Y. L. Yin, and H. Yu, *Collision search attacks on SHA-1*, Research summary, 2005.