Notes on the design and analysis of Simon and Speck

Ray Beaulieu
Douglas Shors
Jason Smith
Stefan Treatman-Clark
Bryan Weeks
Louis Wingers

June 8, 2017

Note. This document was prepared by the designers of Simon and Speck in order to address questions regarding the design rationale and analysis of the algorithms.

1 Introduction

In 2011, prompted by potential U.S. government requirements for lightweight ciphers (e.g., SCADA and logistics applications) and concerns that existing cryptographic solutions were unnecessarily restrictive, a team of cryptographic designers was formed within the National Information Assurance Research Laboratory of NSA's Research Directorate, with the goal of designing foundational lightweight cryptographic block ciphers. See [Age16]. Simon and Speck [BSS+13] emerged from that research effort in 2013.

Because our customers will rely on commercial devices, we determined that the only realistic way to make the algorithms available to them would be to put them in the public domain. Furthermore, because cost will be such an important driver in this arena—a fraction of a penny per device may make the difference between a cryptographic solution being viable or not—we were motivated to make Simon and Speck as simple, flexible, and lightweight as we could. Our hope was that their availability would make it possible to raise the security bar for IoT devices.

There has been a desire expressed by members of the academic community and others for more information about our design goals, methodology, and analytic results. We would like to address these understandable concerns.

We refer the reader to our DAC paper [BTCS+15] and our NIST Lightweight Cryptography Workshop paper [BSS+15] for some previous discussion of the reasons for our parameter choices. This paper expands significantly on those discussions.

2 Design goals

Our aim with the design of Simon and Speck was to enable security on highly constrained devices. We wanted each algorithm to have the full security possible for such a primitive (given the block and key size) against adversaries who can choose plaintext and ciphertext.

We also sought to provide resistance against related-key attacks. This was seen as less crucial for the applications we envisioned, especially since the algorithms were not intended to be used as hashes or in the TWEAKEY framework [JNP14]. However, we understood it would be preferable if the algorithms were immune to related-key attacks, and so we worked to prevent them.

3 Design methodology

Broadly speaking, we believe our design methodology aligns with that of other serious designers of cryptography. Briefly: Informed by intended uses, identify the sort of design you wish to create. This could be a Feistel-based ARX design, an SPN with 4-bit Sboxes, or something else. Then work to optimize choices for that sort of design with respect to security and efficiency.

Our primary aim with Simon and Speck was to facilitate implementations on a range of resource-constrained platforms, while not strongly favoring any particular one. This led us to consider very simple component functions. Because the choices one would make tend to diverge slightly for hardware vs. software platforms, and because of the expertise of various members of the design team, the effort proceeded in two directions. Simon and Speck were each meant to support the full range of hardware and software platforms, but Simon was tuned for slightly better hardware performance (basically, by forgoing modular addition in favor of the bitwise `AND`), and Speck for better software performance (using modular additions).

A cryptographer wants to design algorithms that he or she can understand. "Understand" means that cryptanalytic techniques should be available to the designer for the class of algorithms under consideration, to aid in making informed design choices. The sorts of algorithms we chose to develop were ones where we could carry out the cryptanalysis to our satisfaction. As we've said elsewhere, our aim was to develop the smallest, most flexible algorithms we could, subject to the constraint that we should be able to have high confidence in their security.

3.1 Simon design considerations

Our goal with Simon was to obtain the lightest possible round function; for this we employed the well-studied Feistel construction, which in particular allows decryption to be done easily once the encryption algorithm is implemented. We realized that we could strip the Feistel function down to the point of having a single (wordwise) quadratic term and a single linear term. Further reductions did not lead to viable designs. Of course the cost of such a simple round function is a fairly large number of rounds, in comparison to algorithms with complex round functions. But if the primary goal is to enable very compact implementations, then simple round functions are the way to go.

With one quadratic term and one linear term, there are three rotation parameters to be chosen. For simplicity, we wanted to pick a single set of values for all the sizes of Simon, and for this we were willing to forfeit the ability to individually optimize the choices for each block size.

How should one decide on the rotation values? Considering the case of a 64-bit block (i.e., a 32-bit word), there are $32^3 = 32768$ possible choices. Many of these are obviously poor (having repeated rotation amounts, common differences, a nontrivial gcd with a targeted word size, etc.). There are also symmetries that reduce the number of viable inequivalent candidates by a factor of about 16, and so at this point we had a fairly long list of parameter values, none *obviously* bad.

We considered efficiency issues in order to further whittle down the parameter choices. For 8-bit microcontrollers, in particular, we knew we wanted the rotation amounts to all be as close to multiples of 8 as possible, without sacrificing security. The reason for this is that on a typical 8-bit microcontroller (which we felt was an important platform to support), a rotation by $8k + 2$ is twice as expensive as a rotation by $8k + 1$, and a rotation by $8k + 3$ is three times as expensive. This imposes a pretty stringent constraint on the efficient parameter choices we wished to carry forward for further consideration.

Then the question is this: Given a number of seemingly viable parameter choices that have comparable efficiency, how should one choose between them? For us, this is where the cryptanalytic properties were considered. It was crucial that we optimize with respect to the *relevant* cryptanalytic features: there are many such features to look at, and they're not equally important.

The real issue was to decide which of the viable choices would yield the best algorithm. Since we've stipulated that they all have comparable efficiency (as round functions), the problem was to determine which would minimize the number of rounds required to obtain a secure algorithm. It was therefore necessary to understand what the *limiting* attack methods were, i.e., the ones that yield attacks that made it through the most rounds. We aimed to optimize the algorithms with respect to these cryptanalytic features. We will discuss attacks a little more in Section 4, but the design team's early analytic efforts led us to conclude that the limiting cryptanalytic features for Simon and Speck-type block ciphers would be of the linear and differential sort. So we began by focusing our analysis on those properties. It turns out that this was the right thing to do: in multiple papers published over the last four years, the best reduced-round attacks on Simon have consistently been linear and differential attacks.

We realize that there is an art involved in making parameter choices, as the objective function one is minimizing is not particularly precise, and varies according to implementation goals. Thus it's possible to argue intelligently for different choices than the ones we made.

For example, the Simon variant Simeck [YZS+15] achieves about a 2% hardware area reduction for three of the versions of Simon with four words of key (Simon 32/64, Simon 48/96, Simon 64/128) by modifying the rotation amounts and by replacing the Simon key schedule with a Speck-style key schedule (i.e., a key schedule that is based on the round function).[1] The Simeck parameters, $(\{0,5\},1)$ rather than $(\{1,8\},2)$, are a little worse for 8-bit microcontroller applications, requiring a total of four 1-bit rotations rather than three, but yield slightly better (non-bit-sliced) 64-bit implementations, because one of the rotations is eliminated. But the limiting cryptanalytic features are affected by the new parameters: for example, it takes 19 rounds for the probability of the strongest single difference path through SIMON 64 to fall below 2^{-64}, whereas 25 rounds (31% more) are required to achieve this for Simeck 64 [LLW17]. In the end, we favor our parameter choice because for a tiny cost in hardware, we realize pretty significant security improvements.

In addition to the Simeck work, there have been various claims in the literature (see [KLT15] and [KSI16]) that the SIMON parameter choices are not optimal. Typically this means they are not optimal with respect to some non-limiting cryptanalytic feature of the design[2] and/or because the authors allowed the consideration of parameters which were considerably less efficient than ours. For example, in [KLT15] and [KSI16], $(\{5,12\},3)$ is suggested as an improvement to SIMON's $(\{1,8\},2)$, because certain non-limiting cryptanalytic features—like Boolean degrees and impossible differentials—may be slightly better. (See Section 4 for more on the various attacks.) To shed some light on our design considerations, and to illustrate why in our view this is not a better choice, we would like to examine this choice in a little more detail.

The rotation amounts $(\{5,12\},3)$ are amongst the worst possible choices for performance on 8-bit microcontrollers, because the values are all far from multiples of 8. Therefore this was not an option we considered. On an 8-bit AVR microcontroller, by our count, the modified SIMON 64 round function using these parameters requires more than 60 cycles, whereas SIMON 64 requires 40. If, say, you could reduce the number of rounds by 10%,[3] but each round costs 50%

[1] This may not be the best choice: Zhang and Wu [ZW16] "conclude that it is not advisable for SIMON-like ciphers to re-use the round function in the key schedule." Also, we note that a similar reduction in area is not obtained by this method for the five versions of SIMON with two or three words of key.

[2] An example of a non-limiting cryptanalytic feature for SIMON is the Boolean degree. Compare [ZWW16] and [CW15]: the best reduced-round attack based on degrees for SIMON 64/128, for example, is an integral attack on 24 of 44 rounds, whereas the best linear attack works for 31 of 44 rounds. (We note that the 24-round attack is based on a 17-round distinguisher, and Xiang et al. [XZL16] give an 18-round distinguisher, so the 24-round attack likely extends to a 25-round attack—which is still not competitive with the linear attack.)

[3] 10% is surely an overestimate. Although we have not done the analysis, we suspect that the stepping would not be reduced at all.

more, is that a win? Certainly not in terms of throughput on this device, since $0.9 \cdot 1.5 = 1.35$, corresponding to a 35% hit on throughput.

So our point is this: we believe that to obtain the best possible design, one should not optimize with respect to a cryptanalytic feature without regard to how that feature affects the number of rounds that a secure algorithm would require (which may well be independent of that feature), and without regard for the cost of each one of those rounds.

While the design team did not analyze every possible parameter choice for every instance of Simon and Speck, we did do enough to convince ourselves that we were arriving at good parameter choices for efficient implementations on a broad range of constrained applications. In light of all our analysis, together with four years of academic analysis, we remain convinced that we made sound parameter choices.

For Simon, the key schedule posed another design challenge. We decided to make it linear, because that aided in our ability to understand it, and allowed us to perform some analysis. In order to block related keys, we wanted it to mix bits as quickly as possible, subject to our constraint that it be extremely lightweight. While a full analysis of the related-key paths is clearly much more computationally intensive than the corresponding analysis for ordinary differential paths, it is possible to use Matsui-like techniques to obtain path bounds that ensure that nothing very bad can happen. (And of course security does not rely on the ability to obtain *tight* bounds, just sufficient bounds.)

The issue of related-key attacks is interesting. Some designs (Even-Mansour) accept related-key attacks as intrinsic to the design, and that's typically viewed as acceptable. We did our best to avoid the sort of interactions that lead to related keys, and we believe that related-key attacks do not exist for our algorithms—and none have been found after four years of public scrutiny.

3.2 Speck design considerations

Speck is an ARX ("add, rotate, `XOR`") design—its nonlinearity comes from a modular addition, and it uses `XOR` and rotation for linear mixing. Modular addition is a natural choice over Simon's bitwise `AND` for software performance: at the same computational cost, it's stronger cryptographically. Indeed the ARX construction tends to yield the best performing software algorithms [Ber], [MMH+14].

On an ASIC, modular addition can be done serially using a single full adder, which takes three bits of input (the two bits to be added and a carry bit) and outputs two bits (the sum and the next carry bit). While this fact alone does not guarantee that an ARX design will have compact ASIC implementations, enabling such implementations was a design goal, and they can in fact be realized for Speck [BSS+14]. Even though computation of the addition carry chain means that latency can be relatively high, this is not an issue for ASICs with low clock-speed requirements, where even a 64-bit addition can be executed in a single clock cycle. Furthermore, latency can be reduced at the expense of area by a variety of well-developed techniques (carry-lookahead adders, carry-select adders, etc.).

FPGAs tend to include highly optimized circuitry for modular additions, which means ARX designs can have very high performance on those platforms as well.

As with Simon, we aimed to use a Feistel-like construction for Speck. If modular addition is to be the source of nonlinearity, it's necessary to add two things together, and a natural option is to add the two Feistel words x and y. Note that this choice moves us out of the world of pure Feistel constructions, but the alternative—doing an addition such as $S^a(y) + S^b(y)$—appeared to lead to analytic difficulties. In addition there is a software performance penalty incurred (in the form of move operations) when performing multiple operations on a single word. Speck avoids these penalties in the design of its round function, while Simon accepts them to achieve encrypt/decrypt symmetry, a divergence that is a consequence of the slightly different design goals for the two algorithms.

These considerations led us to the Feistel-like map $(x, y) \mapsto (y, (x + y) \oplus k)$ as the starting point for the Speck round function. On its own this map is cryptographically weak, but it can be strengthened by including some rotations: $(x, y) \mapsto (S^b y, (S^a x + y) \oplus k)$. Operations can still be done in-place for this sort of round function, which is good for software performance. To further strengthen the round function, we composed this map with the computationally inexpensive map $(x, y) \mapsto (y, x \oplus y)$. Note that we did not include another round key here: we could have, but it didn't seem to help much cryptographically and would have doubled the number of round keys required. This would have increased storage requirements for microcontroller implementations. The final version of the round function retains the ability to do operations in-place.

As with Simon, our initial analytic efforts focused on linear and differential properties. Again, subsequent external analysis confirms that this was the correct thing to consider: in the academic literature the best reduced-round attacks on Speck have been seen to be linear and differential attacks. Speck has just two rotation constants, so there are not too many parameter choices (1024 for Speck 64), and it was possible to find all choices which were near optimal with respect to resistance against 8-round differential and linear attacks. As was the case for Simon, many of these led to poor performance on software devices (especially 8-bit microcontrollers) and so were rejected.

For the sake of uniformity, we wanted to use the same parameters for all versions of Speck. The original choice of parameters for Speck was $(7, 2)$, and the resulting algorithm looked to be strong with respect to linear and differential attacks, and also had good performance (see previous discussion regarding 8-bit microcontrollers). Later, we changed the parameters to $(8, 3)$, except in the case of Speck 32: This version appeared to have comparable security and similar performance on 8-bit microcontrollers, but better performance on x86 processors, because an 8-bit rotation can be done using a SIMD byte-shuffle operation. Speck 32 retained the $(7, 2)$ choice because a rotation by 8 on a 16-bit word (i.e., a byte swap) doesn't mix particularly well, and we didn't want Speck 32/64 to require more rounds than Speck 48/72 (both are set at 22 rounds).

As noted in the Sparx design paper [DPU+16], for Speck 32 the parameter options $(9, 2)$, $(9, 5)$, $(11, 7)$, and $(7, 11)$ have slightly improved linear and differential properties over Speck's $(7, 2)$: the optimal differential and linear paths are a factor of two lower in probability and correlation, respectively. With this small improvement, however, it is unlikely that the number of rounds could be reduced. And, as the authors correctly point out, performance on 8-bit microcontrollers is degraded by using rotation amounts farther from multiples of 8 than Speck's $(7, 2)$. This eliminates most of the options, but leaves $(9, 2)$ as an interesting, and viable, choice. It would have comparable efficiency on 8-bit microcontrollers. But for bit-serial ASIC implementations, which are somewhat complicated to describe (see [BSS+14]), parameters (a, b) are worse for larger values of $|a - b|$, and so those implementations are not *quite* as good for $(9, 2)$ as they are for $(7, 2)$.

The next task was to supply the Speck round function with a key schedule. Given that small code size was a major goal of the design, it made sense to reuse the round function for round key generation. This approach enables on-the-fly round key generation for microcontroller implementations using just the round function code, very little ROM, and no RAM beyond what is required to hold the key and plaintext. Of course the round function had to be modified slightly to allow it to operate on 3- and 4-word keys.

We point out that SIMON does not follow the route of reusing the round function to generate round keys, because there is less of a reason to do this for hardware platforms, and because we were not satisfied with the security of such an approach for SIMON. This aligns with the observations made in [ZW16] (see above).

The SPECK key schedule includes a counter to block slide attacks and rotational attacks. Further security aspects of the key schedule are discussed in the next section.

4 Security Analysis

Since their publication in June 2013, a large amount of work has been performed in the academic community to understand the security of the algorithms, with around 70 cryptanalysis papers published so far. Much of this work has been done by leaders in the field, and to date no viable attacks have been found.

SIMON and SPECK were created to further our information assurance mission, and we firmly believe they are secure. The analysis done by the design team as part of the design process aligns with the academic analysis, and the public scrutiny the algorithms have received has only bolstered our confidence in their security.

A desire has been expressed that we publish our analysis of SIMON and SPECK, and we certainly understand the wish to have insight into our analysis. We would like to address that here. We will begin by discussing how we as the design team considered the standard block cipher attacks and their applicability to the security of the SIMON and SPECK designs.

Differential and linear attacks. As the limiting attacks on SIMON and SPECK have been observed to be linear and differential attacks, it is important to understand the linear and differential properties of the algorithms. Fortunately, this has been a focus of the academic research, and it was an area we paid considerable attention to in our design effort.

The design team used standard techniques (Matsui's algorithm, SAT/SMT solvers) to determine optimal differential and linear paths for SIMON and SPECK.

We agree with the results obtained by outside researchers.

Briefly, for Simon, there is a period-16 difference path and a closely related period-16 linear path, each with weight (meaning the negative $\log_2$ of the probability/squared correlation) equal to 60. The weight sequence for these paths, in both the linear and differential case, is `0224264668462422`, which agrees with what was found in [LLW17] and by a number of other authors. All the best paths for Simon 64, Simon 96, and Simon 128 beyond about 18 rounds are based on these paths or their reverses, with some possible twiddling at the ends. (The situation is a little different for Simon 32 and Simon 48, but here we focus on the 64-, 96-, and 128-bit block sizes.)

The results we obtain for Simon agree with those found in [LLW17]. In particular, the design team determined that the single path probabilities (and linear correlations) dip below $2^{-\text{block size}}$ for 12, 16, 20, 29, and 37 rounds for Simon 32, 48, 64, 96, and 128, respectively.

As has been noted by various authors [AAA+15], [AAA+14], [AL13], [CW15], [QHS15], [SHS+14], Simon has a strong multipath effect, largely because of the simplicity of its round function. The lightweight block cipher PRESENT exhibits a similar, and comparable, effect. This was taken into account when setting the number of rounds. We might very conservatively estimate that the number of rounds admitting detectable linear correlations (12, 16, 20, 29, and 37) increases by 50% or so, in the worst case. And then first/last round attack ideas must be factored in, along with a reasonable, but not excessive, security margin. This is the thinking that led us to set the rounds the way we did, as indicated in Table 1.

For Speck, the stepping was based on the best differential paths, which tend to be stronger than the best linear paths. See [FWG+16]. The single difference path probabilities dip below $2^{-\text{block size}}$ for 10, 12, 16, 18, and 21 rounds for Speck 32, 48, 64, 96, and 128, respectively. These results agree with those found by [FWG+16], [SHY16], and [Liu16].

Dinur [Din14] shows that an r-round differential distinguisher yields at least an $(r+m)$-round attack, where m is the number of words of key. For Speck, there is also a slight multipath effect for differences and so an additional round or two can be gained, as noted by Song et al. [SHY16].

Additional rounds were added to obtain a security buffer similar to that of AES-128.

block size	key size	rounds
32	**64**	32
48	**72**	36
	96	36
64	**96**	42
	128	44
96	**96**	52
	144	54
128	**128**	68
	192	69
	256	72

Table 1: SIMON rounds

The best linear paths for SPECK are notably weaker than the best difference paths, with squared correlations dropping below $2^{-\text{block size}}$ in fewer rounds than is necessary for the difference path probabilities.[4] This agrees with what was found (through non-exhaustive searches) in [FWG+16]. In [LWR16], it's proven that for SPECK 32, SPECK 48, and SPECK 64 the squared correlations fall below $2^{-\text{block size}}$ in 10, 11, and 14 rounds, respectively. The linear paths tend to exhibit a stronger multipath effect, but the best linear attacks for SPECK are still worse in every case than the best differential attacks.

The number of rounds for each version of SPECK is shown in Table 2.

The design team found the other sorts of block cipher attacks not to be competitive with the linear and differential attacks. We now consider these attacks; since they are not the limiting attacks, we believe they are of slightly less significance.

Impossible differential/zero correlation attacks. Typically these attacks are of the most concern when an algorithm has a small numbers of rounds. Standard constructions of such features use miss-in-the middle techniques to piece together forward and backward paths. So an algorithm with strong properties that persist (nearly) halfway through the full number of rounds can be susceptible to these methods.

[4] Except in the case of SPECK 32, where there's a tie in the number of rounds required.

block size	key size	rounds
32	**64**	22
48	**72**	22
	96	23
64	**96**	26
	128	27
96	**96**	28
	144	29
128	**128**	32
	192	33
	256	34

Table 2: Speck rounds

Simon and Speck have fairly large numbers of rounds (Simon more than Speck), and the analysis done by the design team indicated that Simon and Speck did not have strong enough paths through half of the rounds (or so) to make these attacks competitive with the best reduced-round attacks. This is supported by the analysis of [CWW15], [LKH+16], [WLV+14]. The best reduced-round impossible difference attack on Simon 64/128, for example, gets through 22 of 44 rounds [CWW15], while the best reduced-round attack overall on Simon 64/128 is a linear attack on 31 rounds [CW15]. And the gap between the best impossible differential/linear attack and the best attack overall widens as the block size increases, as can be seen in these references.

For Speck 64, 6-round impossible differentials have been found using MILP techniques [LKH+16]. Any resulting attack would not be competitive with the best current attack on Speck 64/128, for example, which is a differential attack on 20 of its 27 rounds [SHY16].

The story is very similar for zero-correlation attacks; they are also not competitive with other attacks: For example, 24 of 44 rounds of Simon 64/128 have been attacked by this method [SFW16]. Zero correlation distinguishers on Speck, like the impossible differentials, only appear to get through a handful of rounds.

Meet-in-the-middle attacks. A standard attack of this sort divides the key into three subsets. The first of these subsets constitutes guesses of outer round keys; with any such guess, the second subset allows plaintexts to be stepped forward, and the third allows ciphertexts to be stepped backward. The forward-stepped plaintext and the backward-stepped ciphertext then meet on some common information in the middle to allow the assumptions to be checked. An improved "matchbox meet-in-the-middle" technique effectively allows matching across several rounds in the middle.

Given the fairly large numbers of rounds that Simon and Speck take, and the number of times each word of key effectively gets used (very roughly: look at number of rounds divided by number of words of key), such attacks are unlikely. The best reduced-round attacks of this sort are not competitive with the linear and differential attacks. In [SHMS15] a matchbox meet-in-the-middle attack on 19 (of 44) rounds of Simon 64/128 is described; the attack makes it through a smaller proportion of the rounds for the larger block sizes. We are not aware of any meet-in-middle results on Speck reported in the literature.

Biclique attacks. Biclique attacks are exhaustive attacks which reduce the inner loop key search work below the work to do a single encryption. As such they are not of the *greatest* concern. They are extensions of the meet-in-the-middle approach, and because that approach appears so far from working, given the numbers of rounds, it is unlikely that this approach would lead to anything but a tiny reduction in the work to exhaustively search for a Simon or Speck key.

We note that it is generally possible to organize an exhaustive key search so that the inner-loop work is somewhat less (maybe by 10-30%) than the work to do a full encryption. This can be done for Simon and Speck and virtually every other block cipher. It seems to be generally recognized that the existence of an optimized exhaustive search is not a weakness—if it were considered a weakness, all block ciphers would be broken.

Boomerang/Rectangle attacks. Boomerang attacks are viable when there are strong difference paths that extend about halfway through a block cipher. But boomerang properties tend to decay more rapidly than ordinary differential properties, basically as p^4, where p is the halfway-through path probability.

Again because of the large numbers of rounds, and in view of the known probabilities of optimal difference paths, it appears that these sorts of attacks are inferior to ordinary difference attacks for Simon and Speck. The best reduced round attacks found to date are in [ALLW13b] (a rectangle attack on 20 of 32 rounds of Simon 32/64) and [ALLW13a] (in particular, a rectangle attack on 14 of 27 rounds of Speck 64/128). And as with the impossible differential/linear attacks, the results are worse (with respect to the best reduced-round attacks and as a percentage of the total number of rounds) as the block size increases.

Cube attacks. This attack method seems best suited for the very smallest sizes of Simon and Speck, and even there do not yield results that are competitive with other approaches. The state of the art for attacks of this type are a 17-round attack on Simon 32 [ARSA15] and an 8-round attack on Speck 32 [WW16]. Simon 32 and Speck 32 step 32 and 22 times, respectively.

Algebraic/equation-solving attacks. Attacks using SAT/SMT solvers, Gröbner basis methods, etc., are hard to characterize, for any block cipher. For Simon and Speck these techniques have only been used successfully to break a very small number of rounds, and so they don't appear to be of concern.

For academic research see [Zaj17], where the author is able to attack a handful of rounds (≤ 10) of Simon and Speck. Raddum [Rad15] attacks 16 rounds of Simon 128 (which steps 68, 69, or 72 times, depending on the key size) by equation solving methods, and Courtois [CMS+14] considers the same sorts of attacks on Simon, but does not appear to extend Raddum's result.

Partitions. Simon and Speck are not SPNs and there is no obvious block structure to exploit. It's not clear how to reasonably attempt such an attack, and there is nothing in the literature that we are aware of along these lines.

Slide/rotational attacks. Both Simon and Speck employ round counters to block slide and rotational properties. (To be precise, Speck uses a 1-up counter, because this is easiest in software. Simon saves a small amount in gate area by instead using a 5-bit shift register to produce a sequence of bits.)

We note that, as with many block ciphers, the counters are essential elements of the designs; without them there are rotational attacks. In fact a very early analysis paper described a rotational attack on Speck, but it only worked because the authors of that paper mistakenly omitted the counter (see [ALLW13a] (20130909 version)). Also see [AL16].

Boolean degrees/Integral cryptanalysis. Simon has a degree 2 round function, and because of this the growth in Boolean degree as a function of the number of rounds is relatively slow. But the number of rounds is sufficiently large for Simon that these attacks are not the limiting attacks. For Simon 64/128, reduced-round attacks of this sort get through 18 of 44 rounds; see [KSI16], [TM16], [WLV+14], [XZBL16], [XZL16]. The best reduced-round attack on Simon 64/128 applies to 31 of the 44 rounds. And the attacks are worse, as a percentage of the total rounds, for the larger block sizes.

SPECK's modular addition has high algebraic degree, and for SPECK this is certainly not the attack method of choice. We are aware of no significant results of this type for SPECK.

5 SECURITY MARGINS

A somewhat contentious issue is that of security margins. What constitutes an appropriate security margin? There does not seem to be a consensus here, and there are several reasonable approaches that designers take. Some are quite conservative, and set their stepping to be more-or-less double or triple the number of rounds of the best attack. Some propose a range of possible values for the number of rounds, and then allow a choice to be made based on future cryptanalysis.

We have worked hard to get the stepping right. In our view, an algorithm must be secure, with security margins adequate to protect against future analytic improvements, but at the same time efficiency should not be disregarded, because there is a real-world cost to unnecessarily large security margins.

So we believe that if you have done the necessary work, you can set the stepping of an algorithm without requiring a two- or three-fold margin. Thus, the design team set the stepping with the aim of having security margins comparable to those of existing and trusted algorithms, like AES-128. After four years of concerted effort by academic researchers, the various versions of SIMON and SPECK retain a margin averaging around 30%, and in every case over 25%. The design team's analysis when making stepping decisions was consistent with these numbers.

6 BLOCK SIZE ISSUES

Questions have been raised about the small block size versions of SIMON and SPECK. While there are understandable concerns regarding small block sizes, we believe that such block ciphers could potentially enable security on low-end devices where 128-bit ciphers are not feasible. We focus here on the 64-bit sizes, because that seems to be a sweet spot for lightweight cryptography. Indeed, both before and after the publication of SIMON and SPECK, leading cryptographers have proposed 64-bit block ciphers for constrained devices.

Basically, we believe that for applications where large amounts of data might be encrypted using a single key, block ciphers with 128-bit blocks (or larger) should always be used. In addition, where 128-bit block ciphers can be used, they should be used. For highly constrained applications where only small amounts of data will be encrypted using a single key, and where a 128-block cipher poses a cost or performance barrier to providing security, smaller block sizes may be appropriate.

In the standard modes of operation, like cipher-block chaining mode (CBC), cipher feedback mode (CFB), counter mode (CTR), etc., a block cipher with an n-bit block begins to leak plaintext information as the amount of data encrypted using a single key approaches $2^{n/2}$ blocks. See, for example, [BL16]. For a block cipher with a 128-bit block (like AES, CLEFIA, LEA, SIMON-128 and SPECK-128), this will typically not be an issue, because for most applications the number of blocks encrypted with a single key will stay well below 2^{64}.

Underlying these data leakage issues is the "birthday problem": a collection of $2^{n/2-k}$ randomly chosen n-bit values, for $k \geq 0$, will exhibit a collision—i.e., a repeated value—with probability about 2^{-2k-1}. (In fact 2^{-2k-1} is an upper bound on the collision probability.) $2^{n/2}$ is referred to as the "birthday bound," and given this amount of data, a collision will happen with probability about $1 - e^{-1/2} \approx 0.39$, which is much higher than might naively be guessed. Attacks on modes typically exploit the fact that information can be leaked if a block is repeated (this is the case for CBC mode), or that with enough data it's possible to distinguish a *permutation*, which has distinct outputs given distinct inputs, from a *function*, which can have the same output for two different inputs. (This is an issue for CTR mode.) For these attacks, the advantage of an adversary who makes at most $2^{n/2-k}$ queries to the block cipher is bounded above by the collision probability, which is $\leq 2^{-2k-1}$. For any value of n, for the standard modes (excluding ECB mode, where collisions in inputs can easily be constructed and detected), if the amount of data encrypted using a single key stays *well below* $2^{n/2}$ blocks—i.e., if k is not too small—then these attacks are not a concern.

It is important to note that the idea that there are no security issues until the number of blocks encrypted using a single key reaches $2^{n/2}$ is incorrect. Rather, the security degrades as the number of blocks approaches this number. Thus, the number of blocks encrypted using a single key for an n-bit block cipher should be kept well under $2^{n/2}$. This is discussed further below.

For block sizes n with n smaller than 128, it is important in practice to ensure that there are limits on the amount of data encrypted using a single key. For $n = 64$, for example, $2^{n/2}$ is just 2^{32}, which corresponds to 32 gigabytes of data. This means that the key for a 64-bit block cipher operating in a standard mode should be changed well before the birthday bound, i.e., well before 2^{32} blocks are encrypted. Ensuring that the key is changed after 2^{32-k} blocks are encrypted means that collisions occur, along with the corresponding potential compromise to security, with probability about 2^{-2k-1}. The user's assessment of the risk associated with a particular application should dictate how large a value of k is required.

64-bit block ciphers (in standard modes) are only appropriate for applications where small amounts of data will be encrypted using a single key, and where 128-bit block ciphers are not viable. They should never be used (in standard modes) for applications where the amount of data available to an adversary cannot be tightly controlled.

A possible example of an appropriate use for a 64-bit block cipher is RFID item tracking for items of low to moderate value, where each item has a unique key, and where any particular tag is expected to be queried only a handful of times over its lifetime. Inappropriate uses would include, for example, file encryption on a desktop machine, TLS applications, etc.

References

[AAA+14] Mohamed Ahmed Abdelraheem, Javad Alizadeh, Hoda A. Alkhzaimi, Mohammad Reza Aref, Nasour Bagheri, Praveen Gauravaram, and Martin M. Lauridsen. Improved Linear Cryptanalysis of Reduced-round Simon. Cryptology ePrint Archive, Report 2014/681, 2014. eprint.iacr.org/2014/681.pdf.

[AAA+15] Mohamed Ahmed Abdelraheem, Javad Alizadeh, Hoda A. Alkhzaimi, Mohammad Reza Aref, Nasour Bagheri, and Praveen Gauravaram. Improved Linear Cryptanalysis of reduced-round Simon 32 and Simon 48. Cryptology ePrint Archive, Report 2015/988, 2015. eprint.iacr.org/2015/988.pdf.

[Age16] National Security Agency. Algorithms to Support the Evolution of Information Assurance Needs, 2016. iadgov.github.io/simon-speck/papers/Algorithms-to-Support-the-Evolution-of-Information-Assurance-Needs.pdf.

[AL13] Hoda A. Alkhzaimi and Martin M. Lauridsen. Cryptanalysis of the SIMON Family of Block Ciphers. Cryptology ePrint Archive, Report 2013/543, 2013. eprint.iacr.org/2013/543.pdf.

[AL16] T. Ashur and Y. Liu. Rotational Cryptanalysis in the Presence of Constants. Cryptology ePrint Archive, Report 2016/826, 2016. eprint.iacr.org/2016/826.pdf.

[ALLW13a] F. Abed, E. List, S. Lucks, and J. Wenzel. Cryptanalysis of the SPECK Family of Block Ciphers. Cryptology ePrint Archive, Report 2013/568, 2013. eprint.iacr.org/2013/568.pdf.

[ALLW13b] F. Abed, E. List, S. Lucks, and J. Wenzel. Differential and Linear Cryptanalysis of Reduced-Round SIMON. Cryptology ePrint Archive, Report 2013/526, 2013. eprint.iacr.org/2013/526.pdf.

[ARSA15] Z. Ahmadian, S. Rasoolzadeh, M. Salmasizadeh, and M. Reza Aref. Automated Dynamic Cube Attack on Block Ciphers: Cryptanalysis of SIMON and KATAN. Cryptology ePrint Archive, Report 2015/040, 2015. eprint.iacr.org/2015/040.pdf.

[Ber] Daniel J. Bernstein. ChaCha, a variant of Salsa20.

[BL16] Karthikeyan Bhargavan and Gaëtan Leurent. On the Practical (In-)Security of 64-bit Block Ciphers: Collision Attacks on HTTP over TLS and OpenVPN. Cryptology ePrint Archive, Report 2016/798, 2016. http://eprint.iacr.org/2016/798.

[BSS+13] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. The SIMON and SPECK Families of Lightweight Block Ciphers. Cryptology ePrint Archive, Report 2013/404, 2013. eprint.iacr.org/2013/404.pdf.

[BSS+14] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers. Implementation and Performance of the SIMON and SPECK Lightweight Block Ciphers on ASICs. GitHub, 2014. https://iadgov.github.io/simon-speck/papers/simon-speck-asic-2014.pdf.

[BSS+15] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. Simon and Speck: Block Ciphers for the Internet of Things. NIST Lightweight Cryptography Workshop, July 2015. csrc.nist.gov/groups/ST/lwc-workshop2015/papers/session1-shors-paper.pdf.

[BTCS+15] Ray Beaulieu, Stefan Treatman-Clark, Douglas Shors, Bryan Weeks, Jason Smith, and Louis Wingers. The Simon and Speck Lightweight Block Ciphers. In *Design Automation Conference, DAC 2015*, pages 175:1–175:6. ACM, 2015.

[CMS+14] N. Courtois, T. Mourouzis, G. Song, P. Sepehrdad, and P. Susil. Combined algebraic and truncated differential cryptanalysis on reduced-round SIMON. In *2014 11th International Conference on Security and Cryptography (SECRYPT)*, pages 1–6, 2014.

[CW15] H. Chen and X. Wang. Improved Linear Hull Attack on Round-Reduced Simon with Dynamic Key-guessing Techniques. Cryptology ePrint Archive, Report 2015/666, 2015. eprint.iacr.org/2015/666.pdf.

[CWW15] Zhan Chen, Ning Wang, and Xiaoyun Wang. Impossible Differential Cryptanalysis of Reduced Round Simon. Cryptology ePrint Archive, Report 2015/286, 2015. eprint.iacr.org/2015/286.pdf.

[Din14] Itai Dinur. Improved Differential Cryptanalysis of Round-Reduced Speck. Cryptology ePrint Archive, Report 2014/320, 2014. eprint.iacr.org/2014/320.pdf.

[DPU+16] D. Dinu, L. Perrin, A. Udovenko, V. Velichkov, J. Großschädl, and A. Biryukov. Design Strategies for ARX with Provable Bounds: Sparx and LAX (full version). Cryptology ePrint Archive, Report 2016/984, 2016. eprint.iacr.org/2016/984.pdf.

[FWG+16] Kai Fu, Meiqin Wang, Yinghua Guo, Siwei Sun, and Lei Hu. *MILP-Based Automatic Search Algorithms for Differential and Linear Trails for* Speck, pages 268–288. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[JNP14] Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. Tweaks and Keys for Block Ciphers: the TWEAKEY Framework. Cryptology ePrint

Archive, Report 2014/831, 2014. http://eprint.iacr.org/2014/831.

[KLT15] Stefan Kölbl, Gregor Leander, and Tyge Tiessen. Observations on the SIMON block cipher family. Cryptology ePrint Archive, Report 2015/145, 2015. eprint.iacr.org/2015/145.pdf.

[KSI16] K. Kondo, Y. Sasaki, and T. Iwata. On the Design Rationale of SIMON Block Cipher: Integral Attacks and Impossible Differential Attacks against SIMON Variants. Cryptology ePrint Archive, Report 2016/625, 2016. eprint.iacr.org/2016/625.pdf.

[Liu16] Zhengbin Liu. Automatic search algorithm for differential characteristics in ARX ciphers. In *Chinese Journal of Network and Information, 2016, 2 (5)*, volume 2, pages 56–63, 2016.

[LKH+16] H. Lee, H. Kang, D. Hong, J. Sung, and S. Hong. New Impossible Differential Characteristic of SPECK 64 using MILP. Cryptology ePrint Archive, Report 2016/1137, 2016. eprint.iacr.org/2016/1137.pdf.

[LLW17] Z. Liu, Y. Li, and M. Wang. Optimal Differential Trails in SIMON-like Ciphers. Cryptology ePrint Archive, Report 2017/178, 2017. eprint.iacr.org/2017/178.pdf.

[LWR16] Yuhwen Liu, Qingju Wang, and Vincent Rijmen. *Automatic Search of Linear Trails in ARX with Applications to* SPECK *and Chaskey*, pages 485–499. Springer International Publishing, 2016.

[MMH+14] N. Mouha, B. Mennink, A. Van Herrewege, D. Watanabe, B. Preneel, and I. Verbauwhede. Chaskey: An efficient MAC algorithm for 32-bit microcontrollers. Cryptology ePrint Archive, Report 2014/386, 2014. eprint.iacr.org/2014/386.pdf.

[QHS15] K. Qiao, L. Hu, and S. Sun. Differential Analysis on Simeck and SIMON with Dynamic Key-guessing Techniques. Cryptology ePrint Archive, Report 2015/902, 2015. eprint.iacr.org/2015/902.pdf.

[Rad15] Håvard Raddum. *Algebraic Analysis of the* SIMON *Block Cipher Family*, pages 157–169. Springer International Publishing, 2015.

[SFW16] Ling Sun, Kai Fu, and Meiqin Wang. *Improved Zero-Correlation Cryptanalysis on* SIMON, pages 125–143. Springer International Publishing, 2016.

[SHMS15] Ling Song, Lei Hu, Bingke Ma, and Danping Shi. *Match Box Meet-in-the-Middle Attacks on the* SIMON *Family of Block Ciphers*, pages 140–151. Springer, 2015.

[SHS+14] Danping Shi, Lei Hu, Siwei Sun, Ling Song, Kexin Qiao, and Xiaoshuang Ma. Improved Linear (hull) Cryptanalysis of Round-reduced Versions of SIMON. Cryptology ePrint Archive, Report 2014/973, 2014. eprint.iacr.org/2014/973.pd.

[SHY16] Ling Song, Zhangjie Huang, and Qianqian Yang. *Automatic Differential Analysis of ARX Block Ciphers with Application to* SPECK *and LEA*, pages 379–394. Springer International Publishing, 2016.

[TM16] Yosuke Todo and Masakatu Morii. Bit-Based Division Property and Application to SIMON Family. In *Fast Software Encryption, FSE 2016*, LNCS. Springer, 2016.

[WLV+14] Qingju Wang, Zhiqiang Liu, Kerem Varıcı, Yu Sasaki, Vincent Rijmen, and Yosuke Todo. *Cryptanalysis of Reduced-Round* SIMON *32 and* SIMON *48*, pages 143–160. Springer International Publishing, 2014.

[WW16] L. Wan and Y. Wei. Cube Test and Analysis of SPECK Block Cipher Algorithm. Journal of Computer Engineering, 2016. www.ecice06.com/EN/10.3969/j.issn.1000-3428.2016.11.025.

[XZBL16] Z. Xiang, W. Zhang, Z. Bao, and D. Lin. Applying MILP Method to Searching Integral Distinguishers Based on Division Property for 6 Lightweight Block Ciphers. Cryptology ePrint Archive, Report 2016/857, 2016. eprint.iacr.org/2016/857.pdf.

[XZL16] Z. Xiang, W. Zhang, and D. Lin. On the Division Property of SIMON 48 and SIMON 64. Cryptology ePrint Archive, Report 2016/839, 2016. eprint.iacr.org/2016/839.pdf.

[YZS+15] Gangqiang Yang, Bo Zhu, Valentin Suder, Mark D. Aagaard, and Guang Gong. *The Simeck Family of Lightweight Block Ciphers*, pages 307–329. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[Zaj17] Pavol Zajac. Upper bounds on the complexity of algebraic cryptanalysis of ciphers with a low multiplicative complexity. *Designs, Codes and Cryptography*, 82(1):43–56, 2017.

[ZW16] Huiling Zhang and Wenling Wu. *Structural Evaluation for* SIMON-*like Designs Against Integral Attack*, pages 194–208. Springer International Publishing, 2016.

[ZWW16] Huiling Zhang, Wenling Wu, and Yanfeng Wang. *Integral Attack Against Bit-Oriented Block Ciphers*, pages 102–118. Springer International Publishing, 2016.