Notes on the design and analysis of Simon and Speck

19 January 2018

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 Information Assurance Research Laboratory of NSA's Research Directorate, with the goal of designing foundational lightweight cryptographic block ciphers. SIMON and SPECK emerged from that research effort in 2013. See [Age16].

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 area—a fraction of a penny per device may make the difference between whether a cryptographic solution is 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 the ISO national body members 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 NIST Lightweight Cryptography paper [BSS+15] for some previous discussion of the reasons for our parameter choices. This paper expands significantly on those discussions.

This revision (19 Jan 2018) clarifies some points made in the original version of this paper.

2. Design goals

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

In addition, we sought to provide resistance against related-key attacks. This was seen as less crucial, given the uses we had in mind, and the fact that they were not intended to be used as hashes, and the key was not meant to be used as tweakey. But we understood that 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.

For Simon and Speck our primary aim 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 aim 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 have 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. For this, one needs to understand what the *limiting* attack methods are, i.e., the ones that yield attacks that make 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 believe 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 reduction for the three versions of Simon with four words of key

(Simon32/64, Simon48/96, Simon64/128) 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).* But the limiting cryptanalytic features are affected by the new parameters: for example, the best single difference path through Simon 64 falls to probablity 2^{-64} after 19 rounds, whereas 25 rounds (31% more) are required to achieve this for Simeck 64 [LLW17].

In addition to the Simeck work, there have been various claims in the literature that the Simon parameter choices are not optimal [KLT15], [KSI16]. Typically this means they are not optimal with respect to some non-limiting cryptanalytic feature of the design† 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 Simon64 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%,‡ but each

*This may not be wise; 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.

†An example of a nonlimiting cryptanalytic feature 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.)

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

round costs 50% more, is that a win? Certainly not in terms of throughput on this device, since $0.9 \cdot 1.5 \approx 1.35$, 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 some 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 nearly four years of academic analyis, 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 our understanding of it, and this 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 nearly 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. While this alone does not guarantee that an ARX design will have compact implementations, achieving this was a design goal, and such implementations of SPECK can in fact be realized [BSS+14]. While computation of the addition carry chain means that latency can be relatively high, this is not an issue for many low-end platforms, 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-look 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.

The Feistel-like map $(x, y) \mapsto (y, (x + y) \oplus k)$ was then the starting point for the SPECK round function. On its own, it's 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 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 amount of round key that would have to be generated. This would have negatively affected 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, 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 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 the linear and differential attacks, and also had good performance (see previous discussion of 8-bit microcontroller performance). Later, we changed the parameters to $(8, 3)$, except in the case of Speck 32: This version appeared to have comparable security, comparable performance on 8-bit microcontrollers, but better performance on x86 processors, because a SIMD byte-shuffle operation can be done to effect the 8-bit rotation. Speck32 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).

The Sparx design paper [DPU+16] puts forward the Speck 32 parameter options $(9, 2)$, $(9, 5)$, $(11, 7)$, and $(7, 11)$ as having slightly improved linear and differential properties over Speck's $(7, 2)$, with optimal 9-round differential and linear paths a factor of two lower in both probability and correlation.

We note that a small improvement of this sort does not allow us to reduce the number of rounds the design would require, and it doesn't change security margins; at best it lowers the data requirement for a reduced round attack (on 14 out of 22 rounds) by a factor of two, e.g., from 2^{29} to 2^{28}. 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 leaves $(9, 2)$ as an interesting, and viable, choice, as 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 $(7, 2)$. For SPECK 32, the $(7, 2)$ and $(9, 2)$ choices lead to block ciphers which are virtually tied in terms of both security and performance. We chose $(7, 2)$, but an argument can also be made for $(9, 2)$. Both are secure.

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 in furtherance of 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. Therefore, we would like to address that here. We will begin by addressing how we as the design team considered the standard block cipher attacks and their applicability to the security of the SIMON and SPECK design.

Differential and linear attacks. As the limiting attacks on Simon and Speck have been observed to be differential and linear 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 $-\log_2$ 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 Simon64, Simon96, and Simon128 beyond about 18 steps 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, as those sizes were considered for standardization by ISO. *At the time of this revision, only the 128-bit size is under consideration.*)

The results we obtain for SIMON agree with those found in [LLW17]. In particular, the designer team determined that the single path probabilities (and linear correlations) dip below $2^{-\text{block size}}$ for 12, 16, 20, 29, and 38* 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 estimate that the number of rounds admitting detectable linear correlations (12, 16, 20, 29, and 38) increases by a quarter or so.† 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 the table below.

*The 38 was reported to be 37 in the original version of this paper. We mixed up $\leq$ and $<$ here: for 37 rounds, the probability is equal to 2^{-128}, not $< 2^{-128}$, as we mistakenly claimed.

†The original version of this paper said 50% here, but noted that this was "very conservative." This led to confusion by some, who interpreted 50% as an exact value, rather than the very conservative upper bound we intended it to be. This is supported by the literature (see, e.g., [CW15]) and by our internal analysis. Indeed 50% is a significant overestimate; 25% appears to be a more accurate estimate. We apologize for the lack of clarity here, and note that even if future advances increased the 25% to 50% SIMON would still be secure.

blocksize	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

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 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 in some cases an additional round can be gained or the data requirement can be reduced, as noted by Song et al. [SHY16]. Additional rounds were added to obtain a security buffer similar to that of AES-128, which is ample at 30%. (See, for example, [DFJ12]; the best current attack on AES-128 in the standard attack model is on 7 of its 10 rounds.)

For example, consider SPECK 128/128. As noted above, difference paths extend through 20 rounds; the probability drops below 2^{-128} at 21 rounds. Using the results of Dinur, the 20-round path results in an attack on $r + m = 20 + 2 = 22$ rounds of SPECK128/128. To get a 30% margin with respect to this attack would require 31 rounds, and the stepping for SPECK128/128 was set at 32 rounds. (And because SPECK128/192 and SPECK128/256 have one and two more words of key, respectively, we set the stepping at $32 + 1 = 33$ for

Speck128/192 and 32 + 2 = 34 for Speck128/256.) Dinur's idea was extended by [SHY16], where it shown that $(r + m + 1)$-round attacks are possible. The result is an attack on 23 out of the 32 rounds, which is currently the best attack on Speck128/128. This leaves Speck128/128 with a 28% security margin, similar to that of AES-128.

The best linear paths 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.* This agrees with what was found (through non-exhaustive searches) in [FWG+16]. In [LWR16], it's proven that for Speck32, Speck48, and Speck64 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.

Here are the numbers of rounds for Speck:

blocksize	**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

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.

*Except in the case of Speck32, where there's a tie in the number of rounds required.

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 deterministic properties that persist (nearly) halfway through the full number of rounds can be susceptible to these methods.

Simon and Speck have a 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 Simon64/128 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 Speck64, 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 the other attacks: For example, 24 of 44 rounds of Simon64/128 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 such a 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 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 SIMON64/128 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 virtually always 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 features, 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 SIMON32/64) 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 Simon32 [ARSA15] and an 8-round attack on Speck 32 [WW16]. Simon32 and Speck32 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 Simon128 (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, the 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 threefold 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 almost 4 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.

As a final point, we have presented security margins throughout as percentages. That seems reasonable to us, but it's not entirely clear that it's the

best way of thinking about security margins. One might also compute how many bits of round key would need to be guessed in order to get to a number of rounds for which there's an attack. We could normalize this by dividing by the number of bits of key. For AES-128, this number is $3 \cdot 128/128 = 3$, so the key is effectively included 3 times, beyond the point at which attacks go away. For Speck 128/128, this number is $8 \cdot 64/128 = 4$. For Simon 128/128, it's $19 \cdot 64/128 = 9.5$. This sort of analysis favors block ciphers with simple round functions and lots of rounds, but perhaps doesn't properly weigh the complexity of the round function. The best approach is probably somewhere between these two extremes.

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. 11

[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. 11

[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. 1

[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. 11

[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. 16

[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. 15, 16

[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. 15

[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. 16

[Ber] Daniel J. Bernstein. ChaCha, a variant of Salsa20. 7

[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. 7, 9

[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. 1

[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. 1

[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. 16

[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. 5, 11, 14

[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. 14

[DFJ12] Patrick Derbez, Pierre-Alain Fouque, and Jérémy Jean. Improved key recovery attacks on reduced-round aes in the single-key setting. Cryptology ePrint Archive, Report 2012/477, 2012. eprint.iacr.org/2012/477.pdf. 12

[Din14] Itai Dinur. Improved differential cryptanalysis of round-reduced speck. Cryptology ePrint Archive, Report 2014/320, 2014. eprint.iacr.org/2014/320.pdf. 12

[DPU+16] D. Dinu, L. Perrin, A. Udovenko, V. Velichkov, J. Großschäld, 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. 8

[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. 12, 13

[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. 5

[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. 5, 17

[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. 12

[LKH+16] H. Lee, H. Kang, D. Hong, J. Sung, and S. Hong. New impossible differential characteristic of SPECK64 using MILP. Cryptology ePrint Archive, Report 2016/1137, 2016. eprint.iacr.org/2016/1137.pdf. 14

[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. 5, 10, 11

[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. 13

[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. 7

[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. 11

[Rad15] Håvard Raddum. *Algebraic Analysis of the Simon Block Cipher Family*, pages 157–169. Springer International Publishing, 2015. 16

[SFW16] Ling Sun, Kai Fu, and Meiqin Wang. *Improved Zero-Correlation Cryptanalysis on SIMON*, pages 125–143. Springer International Publishing, 2016. 14

[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. 2015. 15

[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. 11

[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. 12, 13, 14

[TM16] Yosuke Todo and Masakatu Morii. Bit-based division property and application to SIMON family. In *Fast Software Encryption, FSE 2016*, LNCS. Springer, 2016. 17

[WLV+14] Qingju Wang, Zhiqiang Liu, Kerem Varıcı, Yu Sasaki, Vincent Rijmen, and Yosuke Todo. *Cryptanalysis of Reduced-Round SIMON32 and SIMON48*, pages 143–160. Springer International Publishing, 2014. 14, 17

[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. 16

[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. 17

[XZL16] Z. Xiang, W. Zhang, and D. Lin. On the division property of SIMON48 and SIMON64. Cryptology ePrint Archive, Report 2016/839, 2016. eprint.iacr.org/2016/839.pdf. 5, 17

[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. 4

[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. 16

[ZW16] Huiling Zhang and Wenling Wu. *Structural Evaluation for Simon-like Designs Against Integral Attack*, pages 194–208. Springer International Publishing, 2016. 5, 9

[ZWW16] Huiling Zhang, Wenling Wu, and Yanfeng Wang. *Integral Attack Against Bit-Oriented Block Ciphers*, pages 102–118. Springer International Publishing, 2016. 5