Too Much Crypto

Jean-Philippe Aumasson

Taurus, Switzerland

Abstract. We show that many symmetric cryptography primitives would not be less safe with significantly fewer rounds. To support this claim, we review the cryptanalysis progress in the last 20 years, examine the reasons behind the current number of rounds, and analyze the risk of doing fewer rounds. Advocating a rational and scientific approach to round numbers selection, we propose revised number of rounds for AES, BLAKE2, ChaCha, and SHA-3, which offer more consistent security margins across primitives and make them much faster, without increasing the security risk.

Version 20210524090500.

1 Introduction

Designed in the 1970's, neither DES nor GOST are practically broken by cryptanalysis. DES is limited by its 56-bit key length and both are limited by their 64-bit block length. Designed in the late 1990's, AES will likely be unbroken when the human species goes extinct—or so we believe. SHA-3 will likely never be broken either, nor will MD5's preimage resistance. Designing safe symmetric primitives is now well understood, which is why almost never has an information system been compromised through cryptanalysis of a modern primitive, and in particular because of too few rounds[1]

(We restrict this reassuring outlook to symmetric primitives, and acknowledge that spectacular failures can happen for more sophisticated constructions. An example is characteristic-2 supersingular curves' fall from 128-bit to 59-bit security [32].)

The speed of symmetric primitives being inversely proportional to their number of rounds, a natural yet understudied question is whether fewer rounds would be sufficient assurance against cryptanalysis' progress. This article tackles this overdue question, observing that the current number of rounds is often way more than enough for long-term security. We conclude by proposing reduced-round versions of AES, BLAKE2, ChaCha, and SHA-3 that are significantly faster yet as safe as their full-round versions. After decades of cryptanalysis research and consistently diminishing returns, our understanding of and confidence in these primitives indeed calls for a revision of their initial conservative number of rounds.

Our pragmatic, risk-based suggestions may nonetheless not be unanimously endorsed. For example, in 2009 Bruce Schneier wrote this [51]:

> Cryptography is all about safety margins. If you can break n rounds of a cipher, you design it with $2n$ or $3n$ rounds. What we're learning is that the safety margin of AES is much less than previously believed. And while there is no reason to scrap AES in favor of another algorithm, NIST should increase the number of rounds of all three AES variants. At this point, I suggest AES-128 at 16 rounds, AES-192 at 20 rounds, and AES-256 at 28 rounds. Or maybe even more; we don't want to be revising the standard again and again.

Although it is a safe position to recommend even more rounds and always more prudence—for the same reasons that policymakers would apply a strong precautionary principle to avoid criticism—the reality of cryptanalysis research and progress thereof is not aligned with the above conservative guidelines, as this article demonstrates.

2 Security bits

The fewer rounds a symmetric primitive does, the less complex the relation between the input and output tends to be, and the more likely patterns are to exist between inputs and outputs. Such patterns may be exploited by differential cryptanalysis or variants thereof in order to craft an "attack", which is any method that violates a security assumption of the primitive. For example, any method to find a preimage of some 256-bit hash function by doing fewer than 2^{256} evaluations of the function—or equivalent operations—is labelled an attack. This definition of an attack is only about the operations count, not about practical efficiency. For example, a preimage attack requiring 2^{230} operations cannot physically be executed. An algorithm may therefore be broken in theory yet not breakable practically, as further discussed in §§4.4.

[1] Counterexamples include: the Flame malware, which among others exploited an MD5 collision; SHA-1, whose collisions could have been exploited (note that SHA-0 was fixed [sic] by changing constants/operations, rather than adding more rounds); RC4 in WEP, although it was more about a misuse than cryptanalysis, let alone too few rounds. We of course ignore weak custom algorithms, as well as "XOR encryption"—still used in some security products—and cryptographic curiosities such as Time AI™.

128-bit security is often acknowledged as sufficient for most applications, and means that N processors running in parallel would compromise the primitive's security with latency $2^{128}/N$, or the time required to perform of the order of 2^{128} operations by perfectly distributing the workload. If the goal of the attacker is to break at least one 128-bit key out of M instances, then the expected number of operations is not 2^{128} but $2^{128}/(NM)$. For example, if both N and M are equal to 2^{20} (around one million), then $2^{128}/(NM) = 2^{88}$, approximately the age of the universe in nanoseconds (note that testing one candidate key will take much more time than one nanosecond).

Although 2^{88} might sound huge from the above example, relatively close amounts of computations are realistic. Consider Bitcoin for example: at the time of writing mining a Bitcoin block requires approximately 2^{74} evaluations of SHA-256. If one evaluation of SHA-256 took one nanosecond (it actually takes much more), this would be about half a million core-years, however blocks are mined every ten minutes because there are many miners. Extrapolating from the current hashing power, it would therefore take Bitcoin miners 2^{38} years to perform the $\approx 2^{128}$ operations expected to find a SHA-256 collision, for example using parallel birthday collision search [52][2].

Some attacks not only require computation but also large storage capacity. For example, an attack on 7-round AES-128 require 2^{100} bytes of storage [26]. In comparison, the total hard drive capacity shipped in 2016 was around 2^{72} (around four yebibytes). From a physical perspective, Lloyd [46] estimated that "[the] Universe could currently register 10^{90} [or 2^{299}] bits. To register this amount of information requires every degree of freedom of every particle in the Universe.". Applying a more general bound and the holographic principle, Lloyd further calculates that the observable Universe could register approximately 2^{399} bits by using all the information capacity matter, energy, and gravity. To restrict this bound to the volume of Earth, it needs to be multiplied by the ratio between the Earth's volume ($\approx 2^{70}\,\mathrm{m}^3$) and that of the observable universe ($c^2t^2 \approx 2^{268}\,\mathrm{m}^3$), yielding an estimate of 2^{201} bits.

It's hard to draw a line between "practical" and "impractical"—especially with many non-technical factors coming into play—but it's safe to say that comparing time and space complexities over (say) 2^{150} is meaningless from a risk perspective, for all represent an impossible effort. As John Kelsey from NIST once put it,

> The difference between 80 bits and 128 bits of key search is like the difference between a mission to Mars and a mission to Alpha Centauri. As far as I can see, there is no meaningful difference between 192-bit and 256-bit keys in terms of practical bruteforce attacks; impossible is impossible.

Although one could argue that Alpha Centauri is relatively close (only 4.367 light years away), the energy required to accelerate to the speed of light makes this bound unreachable by a human-built spaceship. Because of similar physical limits, a 2^{128}-operations attack must remain fantasy.

3 A history of attacks

3.1 AES

The block cipher AES does 10, 12, and 14 rounds respectively for key lengths—and thus security levels—128, 192, and 256 bits. All three have the same 128-bit block size.

2018 attacks "with practical data and memory complexities" [5] break *5 rounds* with $2^{21.5}$ evaluations of a block encryption if the attacker can query for the ciphertexts of three million plaintext blocks and can store and R/W-access 46 mebibytes in memory. That paper extends

[2]Note that some coins only have 64-bit security, but partially dismiss the risk because the computation of 2^{64} scalar multiplications required to hijack an account is viewed as impractical and unworthwhile.

the attack to *6 rounds*, in which case the complexity grows to 2^{80} with 2^{26} chosen-plaintext queries and 512 gibibytes of storage. It further extends the attack to *7 rounds*, in which case the complexity grows to an impractical $2^{146.3}$ with 2^{30} chosen-plaintext queries and 16 tebibytes of storage (the R/W access cost of which is not considered in the cost evaluation), for a 95% success rate. Of course this attack can only qualify as an attack against the 192- and 256-bit versions of AES.

The best known attack [26] on 7-round AES-128 is expected to require 2^{99} encryption operations, using 2^{97} chosen plaintexts and 2^{100} bytes of storage. From these surrealist figures it is obvious that such an attack is only a cryptanalysis exercise and does not have much to do with the real security of AES.

Arguably even less realistic are key-recovery attacks on the full-round AES, using the "biclique" technique: for AES-128 [17], with time $2^{126.18}$ (against 2^{128} for the generic attack), using 2^{88} plaintext–ciphertext pairs. For AES-256 the figures are respectively $2^{254.42}$ and 2^{40}. Even if $126.16 < 128$ and $254.42 < 256$, the effective cost of these attacks is higher than that of generic bruteforce search[3].

Related-key attacks on AES-256 and -192 discovered in 2009 [12–14] attack require 4 related keys[4], with time $2^{99.5}$ (2^{176}), data $2^{99.5}$ (2^{123}), and memory 2^{77} (2^{152}) for AES-256 (AES-192).

Finally, a "known-key distinguisher" (see §§4.1 for an explanation of this model) for up to 12 rounds was described [33], with complexity around 2^{82} "computations", including 2^{76} table look-ups. (Such distinguishers are not directly applicable to threaten AES' security.)

3.2 BLAKE2

The hash function BLAKE2 has two main versions: 512-bit BLAKE2b does 12 rounds and 256-bit BLAKE2s does 10 rounds.

There are [28] "pseudo-preimage" attacks on *7.5 rounds* of BLAKE2b with $2^{510.3}$ operations and on *6.75 rounds* of BLAKE2s with $2^{253.8}$—against respectively exponents 512 and 256 for the generic attacks. Needless too say, these attacks are not practical. The "pseudo" prefix means that these aren't preimage attacks that conform to the hash function specification, as they do not use the initial value defined in the specification. On the proper hash function, the best preimage attack applies to no more than 2.75 rounds.

"Boomerang distinguishers" are ways to construct sets of input and output values that satisfy some relation by doing fewer operations than expected for an ideal function. Such distinguishers were found for *8.5 rounds* of BLAKE2b's keyed permutation (core component of the compression function) with complexity 2^{474}, and for *7.5 rounds* of BLAKE2s with complexity 2^{184} [39]. Needless to say, such distinguishers don't have much to do with the security of the hash function.

Compared to the results on BLAKE2b's predecessor BLAKE, not much progress was done on BLAKE2's cryptanalysis after its definition in 2012. Guo et al. [36] argued that some of the changes between BLAKE and BLAKE2 strengthened the latter against some classes of attack.

3.3 ChaCha

The stream cipher ChaCha does 20 rounds, for example when used in the ChaCha20-Poly1305 authenticated encryption scheme in TLS and SSH. In the attacks discussed below, ChaCha uses a 256-bit key.

The best result on ChaCha is a key recovery [24] attack on its *7-round* version, with $2^{237.7}$ time complexity (the exact unit is unclear) using output data from 2^{96} instances of ChaCha,

[3] Bogdanov et al. [16] decrease the data requirements of AES bicliques. Huang and Lai [40] generalize the attack for every iterated cipher.

[4] The relation between keys here was quite unusual; while usually one considered xor differences between keys, [13] considered *related subkeys*.

that is, 2^{105} bytes of data. On *6 rounds*, a similar attack can run in time & data 2^{116} & 2^{116}, or $2^{127.5}$ & $2^{37.5}$. On *5 rounds*, the attack becomes practical due to the lower diffusion, and runs in 2^{16} time and data.

Published in 2016, these attacks are a refinement of 2008 results [3] that claimed 2^{248} complexity on 7 rounds of ChaCha. (As a co-author of that paper, let us just say that the 247 exponent was an approximation.)

3.4 SHA-3

The hash functions from the SHA-3 family differ by their hash length and capacity but all do 24 rounds of the internal permutation.

Practical collision attacks exist for *5-round* SHA-3, as successfully implemented on GPUs [37], with a running time of 40 hours for 5-round SHA3-224 and 473 hours (20 days) for 5-round SHA3-256. For non-standard variants, a practical (complexity $\approx 2^{50}$) collision attack of 6-round Keccak with capacity 160 and output length 160 is documented in [37, §6].

Preimage attacks exist [45] for *4-round* SHA3-224 (in 2^{207}) and SHA3-256 (in 2^{239}). For non-standard variants, a practical (complexity $\approx 2^{54}$) preimage attack of 4-round Keccak with capacity 160 and output length 80 is documented in [38, §6.5].

One of the members of the Keccak family, "KangarooTwelve" [11], reduces the number of rounds from 24 to 12, commenting that "clearly, 12 rounds provide less safety margin than the full 24 rounds (...). Still, the safety margin provided by 12 rounds is comfortable as, e.g., the best published collision attacks at time of writing break Keccak only up to 6 rounds."

A variant of "KangarooTwelve" called, "MarsupilamiFourteen" does 14 rounds instead of 12. The two extra rounds are justified as follows: "if one wishes to keep the same safety margin [after modifying other security parameters], an increase of the attack complexity must be compensated by adding rounds". However, the choice of the number of rounds seems arbitrary and based on relative values.

Another member of the Keccak family, "Kravatte" [10]—an instance of the "Farfalle" PRF construction—was initially proposed with internal permutations doing 6 or 4 rounds, depending on where they were called within the construction. This pragmatic choice of number of rounds, with 4 rounds in the final calls to the permutation however turned out to allow for allegedly more efficient attacks than expected: whereas Kravatte was designed to achieve 128-bit security, cryptanalysts [22] found attacks requiring time & space & data $2^{27.8}/2^{76.9}/2^{115.3}$, $2^{51.2}/2^{51.2}/2^{65.1}$, or $2^{29.9}/2^{62.3}/2^{87}$. Whether these attacks actually "break" Kravatte is not clear though, as we'll discuss in §§4.2 and §§4.4.

4 The meaning of attacks

When AES was announced as Rijndael in October 2000, Bruce Schneier wrote the following [50]:

> There is a significant difference between an academic break of a cipher and a break that will allow someone to read encrypted traffic. (Imagine an attack against Rijndael that requires 2^{100} steps. That is an academic break of the cipher, even though it is a completely useless result to anyone trying to read encrypted traffic.) I believe that within the next five years someone will discover an academic attack against Rijndael. I do not believe that anyone will ever discover an attack that will allow someone to read Rijndael traffic. So while I have serious academic reservations about Rijndael, I do not have any engineering reservations about Rijndael.

Although the prophecy was not fulfilled, this summarizes well the difference between attacks in an academic context and in a real context. Note the implicit statement that 100-bit security will forever be enough for practical security.

Imagine indeed the following attack on AES: After 2^{100} chosen-plaintext queries to a black box and 2^{80} "operations" plus 2^{70} "memory", the attack outputs one bit that is a guess whether the box is AES (bit 1) or something else (bit 0). If that algorithm had a 50.1 % chance of making the right guess against a random permutation, the consensus among academic cryptographers is that this would mean that "AES is broken". This parable illustrates how "attacks" can be all but actual attacks and in turn may be misinterpreted by the public. This section therefore attempts to review the notion of attack, concluding with a fact-checking of "Schneier's law".

4.1 Attacks as negative results

The 2019 CFAIL conference[5] was about researchers sharing their failed approaches, and aimed to be "a place for papers that describe instructive failures or not-yet-successes, as they may prefer to be called". Negative results are indeed typically not submitted for publication let alone accepted, yet they're often worth sharing at least to prevent other teams from wasting their time on similar approaches. (This is of course not specific to cryptography.)

The tunable number of rounds offers cryptanalysts a way to present negative results as positive ones, by saying "we still failed to attack the full version, but succeeded in breaking one more round than last time!". Attacks on reduced-round primitives are indeed fundamentally negative results, by demonstrating an inability to violate the security claims of the designers.

Most attacks on reduced-round versions are actually good news for designers, for they mean that cryptanalysts invested significant time in attempting to break the algorithm, yet only succeeded in breaking a handful of rounds. Of course there are exceptions to this rule when full versions get broken, but these are usually experimental designs described in a paper with no real ambition of practical use, or homemade designs created by amateurs. Sometimes such breaks are even an opportunity for designers to publish a second paper with a "fixed" version. The worst scenario is when no attack is published on an algorithm. Undercryptanalysis is for example one of the reasons behind the disqualification of certain algorithms in the SHA-3 and CAESAR contests.

Reducing the number of rounds is the simplest, but not the only way to weaken a primitive in order to claim an attack on a variant of some primitive. Other ways to make a primitive easier to attack (and thus to make it easier to publish a negative result as a positive one) include:

- **Modified internals**, for example when the four 32-bit constants of SHA-1 [1] were modified to find collisions (and making the point that custom constants could serve as backdoors), or when SHA-0's partially linearized compression function "SHI1" was analyzed [21] to understand SHA-0's differential characteristics.

- **Weaker models**, for example when instead of the traditional and realistic fixed-key model researchers craft attacks in the "known-key" or "related-key" models. In the latter the attacker can perform queries to an instance whose key (or round key) has a specified difference with the initial one[6].

- **Weaker attack goals**, typically "distinguishers", or ways to exhibit property that wouldn't hold if the primitive behaved as a random one. For example, "zero-sum distinguishers" [4], arguably the most contrived and meaningless type of attack ever, are about finding a set of input values whose output values sum to zero.

These variants can of course be combined, for example to create "known-key distinguishers" [15, 43]. "Known-key" attacks might sound absurd, but they should not always be discarded, as

[5] Conference for Failed Approaches and Insightful Losses (CFAIL), https://www.cfail2019.com.

[6] Remember that the established model for block ciphers is the chosen-ciphertext, where the attacker gets to query the cipher for plaintext of given ciphertexts, or vice-versa. Of course this rarely captures reality but again if the cipher is safe in this model then it's at least as safe in more reasonable models.

they become relevant when the block cipher is used as component of a hash function. As noted for reduced-round versions, attacks created in unrealistic scenarios can be seen as negative results that speak in favor of the primitive's security. If the best cryptanalysts could find was a distinguisher in the chosen-ciphertext related-key model, then even less likely are practical key recovery attack in the chosen-plaintext model. The publication of such results is therefore a win-win as 1) the cryptanalyst increases their publication count, which is, alas, the primary way to quantify a researcher's value and 2) the community, primitive designers and potential users learn about a negative result or class thereof, and gets a better understanding of what makes an algorithm's strength.

4.2 Attacks on paper vs. in reality

When skimming through a cryptanalysis paper, a reader will typically look for a table in the first three pages summarizing the new results and comparing them to previous results. Such a table will aim to convince a reader that the new attack outperforms previous ones, because the exponents of 2 are all smaller numbers than in the previous best attack. Readers will carry on reading the paper and will be impressed by the ingenuity displayed by the cryptanalysts in crafting new techniques such as triclique partitions and Bojangle projections [48]. The description of the attack will conclude with a complexity analysis where "data", "memory", and "time" figures are more or less rigorously determined.

Unlike complexity theoretic estimates that use asymptotic notations such as $O(n \log n)$ where n is the problem size, cryptanalysts work with fixed-length values and can't work with asymptotics. Indeed, a notation such as $O(2^{80})$ is meaningless and equal to $O(1)$, that is, a constant. Since asymptotics can't work, cryptanalysts have to use another type of approximation, which includes a rough "time" estimate, for example counting the number of iterations of some internal loops, and a "memory" estimate that approximate the amount of storage needed by the attack.

Such estimates are good enough approximations to get an order of magnitude of the attack's efficiency, but they are not (nor do they claim to be) realistic estimates of the attack's actual cost. For example, complexities in cryptanalysis papers ignore the fact that a memory access at a random address is typically orders of magnitude slower than simple arithmetic operations. They also tend to ignore "constant factors," as well as the cost of operations such as searching in or sorting a list.

Another shortcoming of cryptanalysis complexity estimates is how they are compared with generic attacks. A 2^{100}-time attack with memory requirement 2^{99} might sound faster than a 2^{128} recovery attack for a 128-bit key. However when memory is just seen as a physical resource that could be used to host silicon circuits rather than storage units, the comparison is no longer relevant: if physical resources proportional to S are available, then a bruteforce search for an n-bit key will run in effective time $2^n/S$. In our example of a 128-bit key, it would only take 2^{28} cores to reach 2^{100} latency, which is less unrealistic than 2^{103} bytes of storage. Such a 2^{100}-time attack is therefore not more efficient than a generic one.

This example stresses that the area-time (AT) metric model, where the attack cost is viewed as the product between area and time requirements, is more realistic than the model where only time is considered (see also [8,53]). One should nonetheless take into account the fact that silicon is costlier than RAM, thus a (say) 2^{60} memory requirement comes at a lower (energy) cost than 2^{60} computing units.

The upshot is that complexity estimates that look better (that is, smaller) than the generic attack do not necessarily translate in a more effective attack, would the attack be practical (it's often not).

4.3 Attacks and risk

> Risk means more things can happen than will happen. —Elroy Dimson

There's a number R such that after R rounds, and unless some design flaw independent of the number of rounds, a primitive becomes secure and adding more rounds won't make it more secure for any reasonable definition of secure[7]. The question is how to find a good approximation of R? Doing (say) 10000 rounds might be sufficient but is inefficient, while doing just one more round than the number of rounds practically broken may be too risky. To discuss this, we therefore need to know what "broken" means in this context, and to think in terms of risk rather than possibilities.

4.3.1 Perfection is imperfection

A symmetric primitive is essentially a compact description of a function (or family thereof) that behaves like one that would have been chosen uniformly at random. We say that the primitive should be "indistinguishable" from such a function. This means that the primitive, as a black box, should have no exploitable structure. In particular, the algebraic description of the function as a system of equation (e.g. over GF(2)) should have density and coefficient distribution similar to that of a random function. This is the limitation that "zero-sum distinguisher" demonstrated, by exploiting the suboptimal maximal degree over GF(2) of certain primitives. From a theoretical perspective, the "right" number of rounds may therefore be the number of rounds that achieves such a structure-less, strongly indistinguishable property.

A reasoning can then be that unexpected observations such as "known-key distinguishers" reveal some imperfection in the primitive, and therefore the precautionary principle recommends to consider such instances as too risky. This model seems too strong though, since there is generally a large gap between the number of rounds meaningfully broken and the number of rounds for which a distinguisher exists. This is why nobody seems concerned by the impact of 12-round distinguishers on AES, for example.

4.3.2 Cryptographic numerology

The most common model is the one wherein designers make security claims (such as "256-bit preimage resistance") and cryptanalysts' goal is to find an attack running in time 2^n where $n < 256$. When this happen, the primitive is called "broken".

Saying that an attack requiring 2^{234} operations breaks a 256-bit cipher is just a statement about numbers though, not about security. The job of cryptanalysts is to solve puzzles and compare numbers, not to think about what those numbers actually mean. But as noted in §2, numbers such as 2^{192}, 2^{234}, 2^{256} can all be read as $\aleph_0$ (infinite) from a practical perspective—doing 2^{200} operations is not less impossible than doing 2^{256} operations. Of course, reading "256-bit security" rather than "200-bit security" will make us feel better because 256 is a larger number than 200; but this is about psychology, not about security and risk.

We will even argue that there's not much meaningful difference between 128 and 256 as far as practical risk is concerned. Not all cryptographers will share this view, however. For example, in 2007 Daniel J. Bernstein wrote [9] the following:

> I predict that future cryptographers will settle on 256-bit keys as providing a comfortable security level. They will regard 80-bit keys as a silly historical mistake, and 128-bit keys as uncomfortably risky.

[7] Some readers will object by arguing that yes, more rounds are safer because the same algorithm with kN rounds will be k times slower to break through bruteforce than a version with N rounds. But this argument is only meaningful when bruteforce is practical, for example when searching for low-entropy secrets such as passwords or PINs.

He likely meant "risky" not only as in "128-bit bruteforce", but also against improved attacks (again, cryptographers' strong precautionary principle). How risky is this really?

Suppose that against all expectations, someone finds a 2^{100} attack on AES-128 that is embarrassingly parallel and uses no additional memory[8]. Say you have one million cores, each running at 10 GHz, trying one key per cycle, and you're aiming for a one-in-a-thousand success rate: it'd still take you on average about four thousand years of computation to do the $\approx 2^{90}$ required (for only a 1/1000 chance of finding the right key).

A similar scenario actually happened in at least two cases:

- The 64-bit block cipher MISTY1, an ISO standard, can be broken after 2^{70} operations by using the whole 2^{64} codebook [6]. The paper comments that "our attack is clearly impractical due to its large data complexity".
- When security estimates for the pairing-friendly curve BN256 were revised from 128 to around 100 bits [7]. This prompted Cloudflare and Zcash to move to a curve with a higher security level, although they acknowledge the absence of practical risk[9]

We therefore believe that assessing security by comparing numbers without rooting these numbers in reality—what Grigg and Gutmann called "cryptographic numerology" [35]—is not a rational nor effective approach to risk. The next paragraph elaborates on the relative risk of cryptographic failure compared to other risks.

4.3.3 Cryptography is not an island

Let's directly cite Grigg and Gutmann [35]:

> The problem of cryptographic numerology has plagued modern cryptography throughout most of its life. The basic concept is that as long as your encryption keys are at least "this big," you're fine, even if none of the surrounding infrastructure benefits from that size or even works at all. The application of cryptographic numerology conveniently directs attention from the difficult to the trivial, because choosing a key size is fantastically easy, whereas making the crypto work effectively is really hard. In this sense, cryptographic numerology is a prime example of what psychologists call zero-risk bias.

Indeed cryptography is always a component of a broader information system, where the concern is more about securely managing keys[10], rather than picking a primitive that will withstand numerologic attacks for 10 million years.

Grigg and Gutmann even argue that "[the] encryption doesn't even have to be very strong to be useful, it just must be stronger than the other weak links in the system. Using any standard commercial risk management model, cryptosystem failure is orders of magnitude below any other risk." Any cryptographer who has worked with real systems will endorse this thought. For example, the greatest risks with e-voting systems are not the cryptographic protocols and key lengths, but the operational and information security concerns.

Cryptanalysts who restrict their perspective to cryptanalysis might see as alarming an attack with complexity 2^{90} and success rate 1/10, however if we were to quantify the probability of the existence of a remotely exploitable vulnerability in the device you use to read this PDF, it would be close to 1. Such an exploit would let the attack read all your PGP and SSH keys,

[8] Note that the popular use of AES-128-GCM with random 96-bit nonces does not technically meet the 128-bit security criteria.

[9] Ironically, Zcash moved to the curve BLS12-381, which also turned out to have an estimated security level lower than the advertized 128 bits, as noted in https://www.nccgroup.trust/globalassets/our-research/us/public-reports/2019/ncc_group_zcash2018_public_report_2019-01-30_v1.3.pdf (p8).

[10] Generate, provision to cryptographic modules, back-up, restrict access, etc.

regardless of their length. The cost of acquiring or burning an 0-day exploit is clearly less than that of running a 2^{90} complexity attack, plus you don't have to wait.

Companies bragging about picking 256-bit rather than 128-bit security often have systems penetrable with something like 2^5 complexity (and negligible memory, in the chosen-exploit model). For example, HSMs used by leading tech companies have had embarrassing software bugs [31], NATO-certified encryption devices have been found vulnerable to simple attacks, Infineon chips had that embarrassing ROCA bug, SGX' cryptography was proved useless against shared-resources attacks, and so on. In reality, using Triple-DES instead of AES is not a meaningful risk.

"But what if your adversary is NSA or Mossad? Won't they have the computing capabilities to run a 2^{80} attack?" Such a question is irrelevant. If your problem is to protect against such adversaries, the answer is probably not cryptography. As noted by Ian Beer in a great series of Project Zero posts[11], "The reality remains that security protections will never eliminate the risk of attack if you're being targeted."

4.4 Attacks taxonomy

As we've seen, not all attacks are equally threatening and meaningful. The main characteristics of an attack are its adversarial model, its goal, and its real cost (not limited to abstract complexity figures).

The severity of software vulnerabilities is often estimated using similar characteristics that define its exploitability and impact. For example, CVSS takes into account the attack vector (local, remote, etc.), the attack complexity, privileges required, and user interaction to define the exploitability of a vulnerability, while the impact is defined in terms of scope, confidentiality, integrity, and availability[12]

We're not going to set a formal set of criteria to create scores for cryptographic attacks (might be future work). Instead we'll informally define four statuses in which a primitive can be, given the description of an attack. Using this terminology aims to clarify the impact of an attack and reduce miscommunication with the general public.

Our proposed categories are:

- **Analyzed**: When an attack is less efficient than a generic attack both numerically and in practice, and therefore does not violate security claims, yet is publishable because it leverages some internal structure. Example: A key-recovery in time 2^{100} and memory 2^{80} for a 128-bit key cipher.

- **Attacked**: When an attack is numerically more efficient than any generic attack yet remains in the realm of practically impossible tasks. Example: A key-recovery in time 2^{220} for a 256-bit cipher.

- **Wounded**: When an attack is numerically more efficient than any generic attack and whose cost is in a "danger zone" where further improvements could make the attack practical now or in a near future. Example: A key-recovery in time 2^{100}.

- **Broken**: When an attack could realistically be carried out now or in the near future. Example: A key-recovery in time 2^{80}.

Like any model or simplification, these categories are not perfect nor always relevant. A limitation is their poor coverage of extreme cases: For example if an encryption scheme only requires

[11] https://googleprojectzero.blogspot.com/2019/08/a-very-deep-dive-into-ios-exploit.html.

[12] CVSS is not perfect though. As an anonymous security expert noted, "CVSS turned from a desperate attempt at a simple taxonomy to a tool for companies to hide vulnerabilities under a rug by manipulating the variables which make up CVSS scoring."

80-bit security or less, as sufficient for applications such as efficient pointer authentication, then our model defines it as broken by design whereas it's not.

This model also implicitly makes assumptions on the future of technology and scientific progress. It of course assumes that P $\neq$ NP, that attacks and defenders are in the same space-time region, and dismisses the risk of quantum search (see our discussion in §§5.2).

According to this proposed terminology, "Analyzed" and "Attacked" don't require any action, whereas "Wounded" may justify a change of algorithm to mitigate the risk (depending on the application, assets protected, required protection time, and so on), and "Broken" obviously does.

4.5 Attacks always get better

Schneier's law is the tautological-sounding statement "Attacks always get better, they never get worse", which Bruce Schneier popularized, and that (he heard) comes from inside the NSA. The idea of this adage is that we shouldn't underestimate the potential improvement between current and future attacks. While we can't disagree on the last part of the statement—although some attacks did get worse when proven incorrect—the first part deserves closer attention.

What does undoubtedly get better, first of all, is the potential *cost* of an attack, for the cost-efficiency of technology (CPU, storage, RAM, connectivity) keeps improving. In other words, doing a given amount of computation in a given amount of time today costs fewer dollars/euros/bitcoins than it cost yesterday, and it will be even cheaper tomorrow. So we can agree that "attacks always get more cost-efficient".

Furthermore, regardless of the financial cost and of the underlying technology, *implementations* of a given attack likely get better over time, thanks to optimizations of the software or internal algorithms and their parameters.

What about the attacks themselves? More precisely, to what extent do they get better? Is it more about incremental improvements (from impossible to "less impossible") or major improvements (from impossible to possibly practical)? Strictly speaking, what we tend to observe is that a given attack or attack technique may benefit from incremental improvements, whereas major improvements (in terms of complexity and efficiency) come from new types of attacks being discovered, or new attack goals being considered to "break" even more rounds.

Let's briefly see how attacks have gotten better on major symmetric primitives (see also details in §2):

- **AES**: Between 1998 and 2019 attacks got better, but not considerably: in 1998 the initial AES/Rijndael submission document described a generic attack on 6 rounds with 2^{72} complexity and 2^{34} data. In 2000, [29] described attacks on 6 rounds with complexity 2^{44} (for any key size), and on 7 rounds with complexity respectively 2^{155} and 2^{172} for 192- and 256-bit keys. Today 7 rounds can be attacked in time 2^{146}, which remains highly impractical. On 5 rounds, the complexity $2^{21.5}$ from 2018 was improved in 2019 [27] to $2^{16.5}$, using a variant of the boomerang attack (a way to exploit differential characteristics).

- **BLAKE2**: There was no meaningful progress since the differential analysis of BLAKE in 2011 (BLAKE's and BLAKE2's internals are based on ChaCha's permutation).

- **ChaCha**: Between 2008 and 2019, the estimated complexity of an attack went from 2^{248} to 2^{235}, using the same technique but refined analysis.

- **SHA-1**: The collision found in 2017 uses a refined version of the 2005 collision attack, and ran with a complexity close to the one estimated in 2005. The 2019 chosen-prefix attack [44] is a notable improvement and is estimated to run with similar complexity.

- **SHA-256**: Cryptanalysts seem to have given up on looking for collision attacks. The much higher non-linearity, compared to SHA-1, is indeed intimidating and unlikely (in our opinion) to be worked around.

Not being based on reduction to hard problems, we can't prove that attacks on symmetric primitives won't get substantially better, However, given the information and understanding we have today, the discovery of a practical attack on AES, ChaCha, BLAKE2, or Keccak is highly unlikely. Unlike entropy that is bound to increase indefinitely, it looks like attacks on modern primitives are bound to plateau and hit diminishing returns once non-trivial differential analysis has been done. "Schneier's law" should therefore not be blindly followed.

5 Choosing round numbers

Where we examine the reasons behind the number of rounds, comment on the risk posed by quantum computers, and finally propose new primitives for a future where less energy is wasted on computing superfluous rounds.

5.1 The security margin

When you design a new cipher and submit it to a competition (be it NIST's AES, SHA-3, post-quantum, and lightweight crypto, or non-NIST's NESSIE, eSTREAM, and CAESAR), you don't want to wake up the next day and read an email telling you that your cipher is broken. Even after decades of experience, a cryptographer knows all too well that their inability to break their own primitive is a poor indicator of its security. It takes years before gaining sufficient trust in an algorithm's security. It's therefore natural to initially set the number of rounds to a relatively high value to prevent an attack that would eliminate the algorithm from the competition. The difference between the number of rounds known to be insufficient and the number of rounds specified is then known as the *security margin*.

The number of rounds is therefore all but a rational choice; it reflects the the designers' risk appetite and confidence in their own design, and is also influenced by the process that established the primitive as a formal or de facto standard. Arguably, Snowden's 2013 revelations and the ensuing panic contributed to SHA-3's conservativeness. Regarding ChaCha, the eSTREAM actually recommended Salsa20/12, or ChaCha's predecessor with 12 rounds instead of 20, but ChaCha was de facto standardized with 20 rounds.

Look at Keccak/SHA-3, for example: Initially submitted with 18 rounds, the designers increased Keccak's rounds to 24 after the zero-sum observation that "broke" 18 rounds. In hindsight, this decision was probably justified by the fact that the impact of these distinguishers was not clear at the time. The round increase also eliminated the risk of other contenders (or NIST) claiming that Keccak wasn't perfectly secure. However, by the end of the competition, it was clear that 24 rounds was a very high security margin, yet standardizing a version with fewer rounds might not have been accepted by everyone.

Rarely have number of rounds been challenged as too high. A possible reason (simplifying) is that people competent to constructively question the number of rounds have no incentive to promote faster cryptography, and that those who don't have the expertise to confidently suggest fewer rounds. Thus the status quo, which this article questions.

5.2 Quantum security margins

At this point some readers will have thought about the Grover's quadratic speed-up and alleged reduction from (say) 2^{128} to 2^{64} key search complexity. Even though Grover's search (and other quantum cryptanalysis techniques) is mostly independent of the number of rounds, we

would like to briefly discuss the potential impact of scalable quantum computers on symmetric primitives.

Quantum algorithms' impact on symmetric cryptanalysis can be summarized with two categories[13]:

5.2.1 Black-box Grover

Although Grover reduces key search from $O(2^n)$ to $O(2^{n/2})$, one shouldn't ignore the constant factors hiding in the $O()$. Translating this asymptotic speed-up into a square-root of the actual cost is a gross oversimplification; between constant factors, the size and cost of a quantum circuit implementing the attacked primitive, the lack of parallelism [30], and the latency of the circuit, it's actually unclear, given today's quantum computing engineering knowledge, whether Grover would actually be more cost-efficient than classical computers [2, 34, 42, 47, 49]. It's nonetheless a safe bet to assume that it would be.

5.2.2 Quantum cryptanalysis

The most interesting research related to quantum algorithms and symmetric cryptography is arguably a series of papers investigating potential quantum speedups by exploiting a primitive's internal structure [19, 20, 23, 41] (non-exhaustive list). These notably include exponential speed-ups leveraging Simon's algorithm. A major caveat of these results, however, is often their requirement of superposition queries (though recent results attempt to work around this limitation [18]). It's worth noting that most of these results are mostly independent of the number of rounds, and include results that improve the originally non-quantum and round-independent slide attack [41].

5.2.3 Should we care?

When will a quantum computer be available to Grover-search keys for symmetric primitives? Ask 10 cryptographers and you'll get 10 different predictions (for different definitions of "quantum computer"), plus cryptographers are usually not the better informed on this question, due to their limited knowledge of quantum physics and engineering, and their natural incentive to exaggerate the risk. The past scientific and technological progress, as well as the current pace of research and engineering, do not support a prediction of a quantum computer able (say) to run Grover on AES anytime soon, or so we believe.

Dismissing the risk as negligible is however as easy as it is to naively apply the precautionary principle—for every complex question there is an answer that is clear, simple, and wrong, to quote H.L. Mencken. Anyway, the number of rounds would not matter much would AES be Groverable, the answer to that question is therefore not important in choosing a number of rounds.

5.3 How many rounds?

Our goal is to propose numbers of rounds for which we have strong confidence that the algorithm will never be wounded, let alone broken, using the terminology defined in §4.4. Based on the previous research and our cryptanalysis experience, in particular as related to these algorithm, we propose the following:

- **AES**: 9 rounds instead of 10 for AES-128, 10 instead of 12 for AES-192, 11 instead of 14 for AES-256, yielding respectively a 1.1×, 1.2×, and 1.3× speed-up.

[13] A third one might be Shor's exponential speed-up, but it won't apply to symmetric primitives as they rarely rely on Shorable problems—an exception is the VSH hash function [25].

- **BLAKE2**: 8 rounds instead of 12 for BLAKE2b, 7 rounds instead of 10 for BLAKE2s (we'll call these versions BLAKE2bf and BLAKE2sf), yielding respectively a 1.5× and 1.4× speed-up.
- **ChaCha**: 8 rounds instead of 20 (that is, ChaCha8), yielding a 2.5× speed-up.
- **SHA-3**: 10 rounds instead of 24 (we'll call this version *KitTen*, inspired by Keccak family member KangarooTwelve), yielding a 2.4× speed-up.

These suggested numbers of rounds are probably imperfect choices in terms of consistency, yet the security margins of these four families of algorithms are more consistent with these new rounds numbers than with their original ones. Note that AES' initial security margin was the thinnest of all, not because it's the older design, but because of the initial not-overconservative choice of rounds vs. best known attack. AES is thus the primitive for which we propose the smallest change.

6 Conclusion

We proposed to adjust the number of rounds of widely used symmetric primitives in order to 1) revise the security margin from its initial one according to the cryptanalysis performed over more than ten years, and 2) have more consistent security margins across primitives.

A main point from this article piece is that, if performance matters, the number of rounds of a primitive shouldn't be picked randomly as whatever value is sufficiently high. We've also challenged the fact that numbers of rounds are never revised (decreased or increased) based on new information and research. We believe that picking a number of rounds should ideally not only be left to cryptographers, but also involve data scientists and risk management experts.

We don't expect our proposal to receive unanimous support, but hope that it will initiate a discussion and more research related to risk assessment of cryptographic primitives. Standards and libraries of the primitives discussed are not going to change after this paper, but we are confident that our observations will be proven valid over time.

Edit (May 24, 2021): 17 months after the original publication of this paper, its most concrete impact seems to be the adoption of ChaCha8 in a number of projects, notably thanks to its addition to the `chacha20` crate of the RustCrypto project[14]. We are not aware of new research results contradicting the paper's claims, nor of results that would justify (from our point of view) a revision of the proposed reduced-round versions in §§5.3.

Acknowledgments

Thanks to Samuel Neves for numerous insights, references, and reviews of early drafts of this article. Thanks also to Dominique Bongard, Pascal Junod, Nadim Kobeissi, Alfred Menezes, Andrew Miller, Thomas Pornin, Antony Vennard, and Zooko for their feedback and improvement suggestions. Thanks to @laughinghan for reporting a calculus error in the original version.

References

[1] Ange Albertini, Jean-Philippe Aumasson, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Malicious hashing: Eve's variant of SHA-1. In Antoine Joux and Amr M. Youssef, editors, *SAC 2014*, volume 8781 of *LNCS*, pages 1–19. Springer, Heidelberg, August 2014. doi:10.1007/978-3-319-13051-4_1.

[14] https://github.com/RustCrypto/stream-ciphers/commit/b014bfa8f67099ebf42b0f4e158e52c0f45a2824.

[2] Matthew Amy, Olivia Di Matteo, Vlad Gheorghiu, Michele Mosca, Alex Parent, and John M. Schanck. Estimating the cost of generic quantum pre-image attacks on SHA-2 and SHA-3. In Roberto Avanzi and Howard M. Heys, editors, *SAC 2016*, volume 10532 of *LNCS*, pages 317–337. Springer, Heidelberg, August 2016. doi:10.1007/978-3-319-69453-5_18.

[3] Jean-Philippe Aumasson, Simon Fischer, Shahram Khazaei, Willi Meier, and Christian Rechberger. New features of latin dances: Analysis of Salsa, ChaCha, and Rumba. In Kaisa Nyberg, editor, *FSE 2008*, volume 5086 of *LNCS*, pages 470–488. Springer, Heidelberg, February 2008. doi:10.1007/978-3-540-71039-4_30.

[4] Jean-Philippe Aumasson and Willi Meier. Zero-sum distinguishers for reduced Keccak-f and for the core functions of Luffa and Hamsi. Rump session of CHES 2009, 2009.

[5] Achiya Bar-On, Orr Dunkelman, Nathan Keller, Eyal Ronen, and Adi Shamir. Improved key recovery attacks on reduced-round AES with practical data and memory complexities. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 185–212. Springer, Heidelberg, August 2018. doi:10.1007/978-3-319-96881-0_7.

[6] Achiya Bar-On and Nathan Keller. A 2^{70} attack on the full MISTY1. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 435–456. Springer, Heidelberg, August 2016. doi:10.1007/978-3-662-53018-4_16.

[7] Razvan Barbulescu and Sylvain Duquesne. Updating key size estimations for pairings. Cryptology ePrint Archive, Report 2017/334, 2017. https://eprint.iacr.org/2017/334.

[8] Daniel J. Bernstein. Understanding brute force, 2005. URL: http://cr.yp.to/papers.html#bruteforce.

[9] Daniel J. Bernstein. The Salsa20 family of stream ciphers. In Matthew J. B. Robshaw and Olivier Billet, editors, *New Stream Cipher Designs - The eSTREAM Finalists*, volume 4986 of *LNCS*, pages 84–97. Springer, 2008. doi:10.1007/978-3-540-68351-3_8.

[10] Guido Bertoni, Joan Daemen, Seth Hoffert, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. Farfalle: parallel permutation-based cryptography. *IACR Trans. Symm. Cryptol.*, 2017(4):1–38, 2017. doi:10.13154/tosc.v2017.i4.1-38.

[11] Guido Bertoni, Joan Daemen, Michaël Peeters, Gilles Van Assche, Ronny Van Keer, and Benoît Viguier. KangarooTwelve: Fast hashing based on Keccak-p. In Bart Preneel and Frederik Vercauteren, editors, *ACNS 18*, volume 10892 of *LNCS*, pages 400–418. Springer, Heidelberg, July 2018. doi:10.1007/978-3-319-93387-0_21.

[12] Alex Biryukov, Orr Dunkelman, Nathan Keller, Dmitry Khovratovich, and Adi Shamir. Key recovery attacks of practical complexity on AES-256 variants with up to 10 rounds. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 299–319. Springer, Heidelberg, May / June 2010. doi:10.1007/978-3-642-13190-5_15.

[13] Alex Biryukov and Dmitry Khovratovich. Related-key cryptanalysis of the full AES-192 and AES-256. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 1–18. Springer, Heidelberg, December 2009. doi:10.1007/978-3-642-10366-7_1.

[14] Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolic. Distinguisher and related-key attack on the full AES-256. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 231–249. Springer, Heidelberg, August 2009. doi:10.1007/978-3-642-03356-8_14.

[15] Céline Blondeau, Thomas Peyrin, and Lei Wang. Known-key distinguisher on full PRESENT. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 455–474. Springer, Heidelberg, August 2015. doi:10.1007/978-3-662-47989-6_22.

[16] Andrey Bogdanov, Donghoon Chang, Mohona Ghosh, and Somitra Kumar Sanadhya. Bicliques with minimal data and time complexity for AES. In Jooyoung Lee and Jongsung Kim, editors, *ICISC 14*, volume 8949 of *LNCS*, pages 160–174. Springer, Heidelberg, December 2015. doi:10.1007/978-3-319-15943-0_10.

[17] Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biclique cryptanalysis of the full AES. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 344–371. Springer, Heidelberg, December 2011. doi:10.1007/978-3-642-25385-0_19.

[18] Xavier Bonnetain, Akinori Hosoyamada, María Naya-Plasencia, Yu Sasaki, and André Schrottenloher. Quantum attacks without superposition queries: the offline simon algorithm. Cryptology ePrint Archive, Report 2019/614, 2019. https://eprint.iacr.org/2019/614.

[19] Xavier Bonnetain and María Naya-Plasencia. Hidden shift quantum cryptanalysis and implications. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part I*, volume 11272 of *LNCS*, pages 560–592. Springer, Heidelberg, December 2018. doi:10.1007/978-3-030-03326-2_19.

[20] Xavier Bonnetain, María Naya-Plasencia, and André Schrottenloher. Quantum security analysis of AES. *IACR Trans. Symm. Cryptol.*, 2019(2):55–93, 2019. doi:10.13154/tosc.v2019.i2.55-93.

[21] Florent Chabaud and Antoine Joux. Differential collisions in SHA-0. In Hugo Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 56–71. Springer, Heidelberg, August 1998. doi:10.1007/BFb0055720.

[22] Colin Chaigneau, Thomas Fuhr, Henri Gilbert, Jian Guo, Jérémy Jean, Jean-René Reinhard, and Ling Song. Key-recovery attacks on full Kravatte. *IACR Trans. Symm. Cryptol.*, 2018(1):5–28, 2018. doi:10.13154/tosc.v2018.i1.5-28.

[23] André Chailloux, María Naya-Plasencia, and André Schrottenloher. An efficient quantum collision search algorithm and implications on symmetric cryptography. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 211–240. Springer, Heidelberg, December 2017. doi:10.1007/978-3-319-70697-9_8.

[24] Arka Rai Choudhuri and Subhamoy Maitra. Significantly improved multi-bit differentials for reduced round Salsa and ChaCha. *IACR Trans. Symm. Cryptol.*, 2016(2):261–287, 2016. URL: https://tosc.iacr.org/index.php/ToSC/article/view/574.

[25] Scott Contini, Arjen K. Lenstra, and Ron Steinfeld. VSH, an efficient and provable collision resistant hash function. Cryptology ePrint Archive, Report 2005/193, 2005. https://eprint.iacr.org/2005/193.

[26] Patrick Derbez, Pierre-Alain Fouque, and Jérémy Jean. Improved key recovery attacks on reduced-round AES in the single-key setting. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 371–387. Springer, Heidelberg, May 2013. doi:10.1007/978-3-642-38348-9_23.

[27] Orr Dunkelman, Nathan Keller, Eyal Ronen, and Adi Shamir. The retracing boomerang attack. Cryptology ePrint Archive, Report 2019/1154, 2019. https://eprint.iacr.org/2019/1154.

[28] Thomas Espitau, Pierre-Alain Fouque, and Pierre Karpman. Higher-order differential meet-in-the-middle preimage attacks on SHA-1 and BLAKE. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 683–701. Springer, Heidelberg, August 2015. doi:10.1007/978-3-662-47989-6_33.

[29] Niels Ferguson, John Kelsey, Stefan Lucks, Bruce Schneier, Michael Stay, David Wagner, and Doug Whiting. Improved cryptanalysis of Rijndael. In Bruce Schneier, editor, *FSE 2000*, volume 1978 of *LNCS*, pages 213–230. Springer, Heidelberg, April 2001. doi:10.1007/3-540-44706-7_15.

[30] Scott Fluhrer. Reassessing Grover's algorithm. Cryptology ePrint Archive, Report 2017/811, 2017. http://eprint.iacr.org/2017/811.

[31] Jean-Baptiste Bédrune Gabriel Campana. Everybody be cool, this is a robbery! Black Hat briefings, 2019.

[32] Robert Granger, Thorsten Kleinjung, and Jens Zumbrägel. Breaking '128-bit secure' supersingular binary curves - (or how to solve discrete logarithms in $\mathbb{F}_{2^{4\cdot 1223}}$ and $\mathbb{F}_{2^{12\cdot 367}}$). In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 126–145. Springer, Heidelberg, August 2014. doi:10.1007/978-3-662-44381-1_8.

[33] Lorenzo Grassi and Christian Rechberger. New and old limits for AES known-key distinguishers. Cryptology ePrint Archive, Report 2017/255, 2017. https://eprint.iacr.org/2017/255.

[34] Markus Grassl, Brandon Langenberg, Martin Roetteler, and Rainer Steinwandt. Applying Grover's algorithm to AES: quantum resource estimates. In *PQCrypto*, 2016. https://arxiv.org/abs/1512.04965.

[35] Ian Grigg and Peter Gutmann. The curse of cryptographic numerology. *IEEE Security & Privacy*, 9, 2011. URL: https://www.computer.org/csdl/magazine/sp/2011/03/msp2011030070/13rRUxBa54s.

[36] Jian Guo, Pierre Karpman, Ivica Nikolic, Lei Wang, and Shuang Wu. Analysis of BLAKE2. In Josh Benaloh, editor, *CT-RSA 2014*, volume 8366 of *LNCS*, pages 402–423. Springer, Heidelberg, February 2014. doi:10.1007/978-3-319-04852-9_21.

[37] Jian Guo, Guohong Liao, Guozhen Liu, Meicheng Liu, Kexin Qiao, and Ling Song. Practical collision attacks against round-reduced SHA-3. *IACR Cryptology ePrint Archive*, 2019:147, 2019. URL: https://eprint.iacr.org/2019/147.

[38] Jian Guo, Meicheng Liu, and Ling Song. Linear structures: Applications to cryptanalysis of round-reduced Keccak. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 249–274. Springer, Heidelberg, December 2016. doi:10.1007/978-3-662-53887-6_9.

[39] Yonglin Hao. The boomerang attacks on BLAKE and BLAKE2. In *Inscrypt*, volume 8957 of *LNCS*, pages 286–310. Springer, 2014.

[40] Jialin Huang and Xuejia Lai. What is the effective key length for a block cipher: an attack on every block cipher. Cryptology ePrint Archive, Report 2012/677, 2012. http://eprint.iacr.org/2012/677.

[41] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Breaking symmetric cryptosystems using quantum period finding. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 207–237. Springer, Heidelberg, August 2016. doi:10.1007/978-3-662-53008-5_8.

[42] Panjin Kim, Daewan Han, and Kyung Chul Jeong. Time-space complexity of quantum search algorithms in symmetric cryptanalysis: applying to AES and SHA-2. *Quantum Information Processing*, 17(12):339, 2018. URL: https://doi.org/10.1007/s11128-018-2107-3.

[43] Lars R. Knudsen and Vincent Rijmen. Known-key distinguishers for some block ciphers. In Kaoru Kurosawa, editor, *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 315–324. Springer, Heidelberg, December 2007. doi:10.1007/978-3-540-76900-2_19.

[44] Gaëtan Leurent and Thomas Peyrin. From collisions to chosen-prefix collisions application to full SHA-1. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 527–555. Springer, Heidelberg, May 2019. doi:10.1007/978-3-030-17659-4_18.

[45] Ting Li and Yao Sun. Preimage attacks on round-reduced keccak-224/256 via an allocating approach. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 556–584. Springer, Heidelberg, May 2019. doi:10.1007/978-3-030-17659-4_19.

[46] Seth Lloyd. Computational capacity of the universe. *Phys. Rev. Lett.*, 88:237901, May 2002. URL: https://arxiv.org/abs/quant-ph/0110141.

[47] Alfred Menezes. The computational supersingular isogeny problem. NutMIC invited talk, 2019. http://nutmic2019.imj-prg.fr/slides/Menezes-NutMic.pdf.

[48] James Mickens. This world of ours. *USENIX ;login: logout*, January 2014.

[49] Ray Perlner and Yi-Kai Liu. Thermodynamic analysis of classical and quantum search algorithms, 2017. URL: https://arxiv.org/abs/1709.10510.

[50] Bruce Schneier. AES announced. https://www.schneier.com/crypto-gram/archives/2000/1015.html#8, 2000.

[51] Bruce Schneier. Another new AES attack. https://www.schneier.com/blog/archives/2009/07/another_new_aes.html, 2009.

[52] Paul C. van Oorschot and Michael J. Wiener. Parallel collision search with cryptanalytic applications. *Journal of Cryptology*, 12(1):1–28, January 1999. doi:10.1007/PL00003816.

[53] Michael J. Wiener. The full cost of cryptanalytic attacks. *Journal of Cryptology*, 17(2):105–124, March 2004. doi:10.1007/s00145-003-0213-5.